Ex.No:2 **Hadoop Installation**

Hadoop 3.4.1 Installing on Ubuntu 24.04 (Single-Node Cluster)

Step 1: Update the OS

```
npprakash@npprakashhp:~$ sudo apt-get update
```

//*apt-get update* updates the package lists for upgrades for packages that need upgrading, as well as new packages that have just come to the repositories.

Step 2: Installing Java

```
npprakash@npprakashhp:~$ sudo apt-get install default-jdk
```

Step 2.1 Check the version

```
npprakash@npprakashhp:~$ java -version
```

```
openjdk version "21.0.5" 2024-10-15
OpenJDK Runtime Environment (build 21.0.5+11-Ubuntu-1ubuntu124.04)
OpenJDK 64-Bit Server VM (build 21.0.5+11-Ubuntu-1ubuntu124.04, mixed mode,
sharing)
```

Step 3: Adding a dedicated Hadoop user

The next step is to create a dedicated user and group for our Hadoop installation. This allows all of the installation to be insulated from the rest of the environment, as well as enable tighter security measures to be enforced (in case you have a production environment). We will create a user hduser and a group hadoop, and add the user to the group. This can be done using the following commands.

```
npprakash@npprakashhp:~$ sudo addgroup hadoop
```

```
Adding group `hadoop' (GID 1001) ...
Done.
```

```
npprakash@npprakashhp:~$ sudo adduser --ingroup hadoop hduser
```

```
      Adding user `hduser' ...
      Adding new user `hduser' (1001) with group `hadoop' ...
      Creating home directory `/home/hduser' ...
      Copying files from `/etc/skel' ...
      Enter new UNIX password:
      Retype new UNIX password:
      passwd: password updated successfully
      Changing the user information for hduser
      Enter the new value, or press ENTER for the default
            Full Name []:
            Room Number []:
            Work Phone []:
            Home Phone []:
            Other []:
      Is the information correct? [Y/n] Y
```

Step 3.1 We can check if we create the **hadoop** group and **hduser** user:

```
nppprakash@nppprakashhp:~$ groups hduser

      hduser : hadoop users
```

Step 4 : Installing SSH

The hadoop control scripts rely on SSH to peform cluster-wide operations. For example, there is a script for stopping and starting all the daemons in the clusters. To work seamlessly, SSH needs to be setup to allow password-less login for the hadoop user from machines in the cluster. The simplest way to achive this is to generate a public/private key pair, and it will be shared across the cluster.

Hadoop requires SSH access to manage its nodes, i.e. remote machines plus your local machine. For our single-node setup of Hadoop, we therefore need to configure SSH access to localhost for the hduser user we created in the earlier.

We have to generate an SSH key for the hduser user.

```
nppprakash@nppprakashhp:~$ sudo apt-get install ssh
```

This will install ssh on our machine. If we get something similar to the following, we can think it is setup properly:

```
nppprakash@nppprakashhp:~$ which ssh

      /usr/bin/ssh

nppprakash@nppprakashhp:~$ which sshd

      /usr/sbin/sshd
```

Step 4.1 :

Hadoop uses SSH (to access its nodes) which would normally require the user to enter a password. However, this requirement can be eliminated by creating and setting up SSH certificates using the following commands. If asked for a filename just leave it blank and press the enter key to continue.

```
nppprakash@nppprakashhp:~$ su hduser
```

```
hduser@nppprakashhp:/home/nppprakash$ ssh-keygen -t rsa -P ""
Geneyrating public/private rsa key pair.
Enter file in which to save the key (/home/hduser/.ssh/id_rsa):
Created directory '/home/hduser/.ssh'.
Your identification has been saved in /home/hduser/.ssh/id_rsa.
Your public key has been saved in /home/hduser/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:/M18Dv+ku5js8npZvYi45Fr4F84SzoqXBUO5xAfo+/8 hduser@nppprakashhp
The key's randomart image is:
+---[RSA 2048]----+
|      o.o        |
|     . = .       |
|    . o o        |
|     . =         |
|      . S       .|
|     .   .+ +    o|
|      ..=o* * .oo|
|      .+== *.B++ |
|     ..o+==EB*B+.|
+----[SHA256]-----+
```

Step 4.2 The following command adds the newly created key to the list of authorized keys so that Hadoop can use ssh without prompting for a password.

```
hduser@nppprakashhp:/home/nppprakash$ cat $HOME/.ssh/id_rsa.pub >>
$HOME/.ssh/authorized_keys
```

Step 4.3 We can check if ssh works:

```
hduser@nppprakashhp:/home/nppprakash$  ssh localhost

hduser@localhost's password:

Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-28-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
232 packages can be updated.
117 updates are security updates.
Last login: Wed Nov  1 19:38:55 2017 from 127.0.0.1
```

Step 5: Install Hadoop

```
hduser@nppprakashhp:~$                                                  wget
https://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-3.4.1/hadoop-
3.4.1.tar.gz
```

Step 5.1 : move the Hadoop installation to the **/usr/local/hadoop** directory. So, we should create the directory first:

```
hduser@nppprakashhp:~$ sudo mkdir -p /usr/local/hadoop
```

Step 5.2 We can check again if **hduser** is not in **sudo** group:

```
hduser@npprakashhp:~$  sudo -v
Sorry, user hduser may not run sudo on laptop.
```

This can be resolved by logging in as a root user, and then add **hduser** to **sudo** group:

```
hduser@npprakashhp:~$ su npprakash
Password:
npprakash@npprakashhp:/home/hduser$
```

Now, the **hduser** has root priviledge, we can move the Hadoop installation to the **/usr/local/hadoop** directory without any problem:

```
npprakash@npprakashhp:/home/hduser$ sudo su hduser

hduser@npprakashhp:~/hadoop-3.4.1$ sudo mv * /usr/local/hadoop

hduser@npprakashhp:~/hadoop-3.4.1$ sudo chown -R hduser:hadoop
/usr/local/hadoop
```

Step 6: Setup Configuration Files

Step 6.1 : Edit ~/.bashrc file

Before editing the **.bashrc** file in **hduser**'s home directory, we need to find the path where Java has been installed to set the **JAVA_HOME** environment variable using the following command:

```
hduser@npprakashhp:~$ update-alternatives --config java
There is only one alternative in link group java (providing /usr/bin/java): /usr/lib/jvm/java-21-openjdk-amd64/jre/bin/java
Nothing to configure.
```

Now we can append the following to the end of **~/.bashrc**:

```
hduser@npprakashhp:~$  vim ~/.bashrc

#HADOOP VARIABLES START
export JAVA_HOME=/ usr/lib/jvm/java-21-openjdk-amd64
export HADOOP_INSTALL=/usr/local/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
#HADOOP VARIABLES END

hduser@laptop:~$ source ~/.bashrc
```

Step 6.2 : **2. /usr/local/hadoop/etc/hadoop/hadoop-env.sh**
Adding the belwo statement in the **hadoop-env.sh** file ensures that the value of JAVA_HOME variable will be available to Hadoop whenever it is started up.

```
hduser@npprakashhp:~$  vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh
```

```
export JAVA_HOME=/usr/lib/jvm/java-21-openjdk-amd64
```

Step 6.3 : **3. /usr/local/hadoop/etc/hadoop/core-site.xml**:

The **/usr/local/hadoop/etc/hadoop/core-site.xml** file contains configuration properties that Hadoop uses when starting up.

This file can be used to override the default settings that Hadoop starts with.

```
hduser@npprakashhp:~$ sudo mkdir -p /app/hadoop/tmp
```

```
hduser@npprakashhp:~$ sudo chown hduser:hadoop /app/hadoop/tmp
```

Open the file and enter the following in between the <configuration></configuration> tag:

```
hduser@npprakashhp:~$ vi /usr/local/hadoop/etc/hadoop/core-site.xml
```

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

Step 6.4 : **/usr/local/hadoop/etc/hadoop/mapred-site.xml**

By default, the /usr/local/hadoop/etc/hadoop/ folder contains
/usr/local/hadoop/etc/hadoop/mapred-site.xml.template
file which has to be renamed/copied with the name mapred-site.xml:

```
hduser@npprakashhp:~$  cp /usr/local/hadoop/etc/hadoop/mapred-
site.xml.template /usr/local/hadoop/etc/hadoop/mapred-site.xml
```

The /usr/local/hadoop/etc/hadoop/mapred-site.xml file is used to specify which framework is being used for MapReduce.

```
We need to enter the following content in between the
<configuration></configuration> tag:
```

```
hduser@npprakashhp:~$  vim usr/local/hadoop/etc/hadoop/mapred-site.xml
```

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

Step 6.5 : /usr/local/hadoop/etc/hadoop/hdfs-site.xml

The /usr/local/hadoop/etc/hadoop/hdfs-site.xml file needs to be configured for each host in the cluster that is being used.

It specifies the directories which will be used as the namenode and the datanode on that host. Before editing this file, we need to create two directories which will contain the namenode and the datanode for this Hadoop installation.

This can be done using the following commands:

```
hduser@npprakashhp:~$  sudo mkdir -p /usr/local/hadoop_store/hdfs/namenode

hduser@npprakashhp:~$   sudo mkdir -p /usr/local/hadoop_store/hdfs/datanode

hduser@npprakashhp:~$  sudo chown -R hduser:hadoop /usr/local/hadoop_store
```

Open the file and enter the following content in between the <configuration></configuration> tag:

```
hduser@npprakashhp:~$  vim /usr/local/hadoop/etc/hadoop/hdfs-site.xml
```

```
<configuration>
 <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>
 <property>
   <name>dfs.namenode.name.dir</name>
   <value>file:/usr/local/hadoop_store/hdfs/namenode</value>
 </property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/usr/local/hadoop_store/hdfs/datanode</value>
 </property>
</configuration>
```

Step 7. Format the New Hadoop Filesystem

Now, the Hadoop file system needs to be formatted so that we can start to use it. The **format** command should be issued with write permission since it creates **current** directory under **/usr/local/hadoop_store/hdfs/namenode** folder:

```
hduser@nnprakashhp:~$  hadoop namenode -format
```

Important Note :

➔ **Note that hadoop namenode -format command should be executed once before we start using Hadoop.**

➔ **If this command is executed again after Hadoop has been used, it'll destroy all the data on the Hadoop file system.**

Step 8 : Starting Hadoop

Start NameNode daemon and DataNode daemon:

```
hduser@nnprakashhp:/usr/local/hadoop/sbin$ start-dfs.sh
```

Start ResourceManager daemon and NodeManager daemon:

```
hduser@nnprakashhp:/usr/local/hadoop/sbin$ start-yarn.sh
```

We can check if it's really up and running:

```
hduser@nnprakashhp:/usr/local/hadoop/sbin$ jps

7040 NameNode
7956 Jps
7156 DataNode
7525 ResourceManager
7367 SecondaryNameNode
7834 NodeManager
```

Step 9: Stopping Hadoop

```
hduser@nnprakashhp:/usr/local/hadoop/sbin$ stop-dfs.sh

hduser@nnprakashhp:/usr/local/hadoop/sbin$ stop-yarn.sh
```

Step 10 : http://localhost:50070/ to check the hadoop User interface

Errors and Other Commands

Note : Sudoers Issue ::

edit /etc/sudoers

```
change the user privilege specficiation to hduser

hduser ALL=(ALL:ALL) ALL
```

or use:

```
at root the following command:

sudo usermod -a -G sudo hduser
```

Some important commands

To remove hduser:

deluser hduser

To remove hadoop group

deluser –-group hadoop

To uninstall hadoop

sudo rm -r -f location (/usr/local/hadoop)

To un install java

Remove all the Java related packages (Sun, Oracle, OpenJDK, IcedTea plugins, GIJ):

```
dpkg-query -W -f='${binary:Package}\n' | grep -E -e '^(ia32-)?(sun|oracle)-java' -e '^openjdk-' -e '^icedtea' -e '^(default|gcj)-j(re|dk)' -e '^gcj-(.*)-j(re|dk)' -e '^java-common' | xargs sudo apt-get -y remove
sudo apt-get -y autoremove
```

- **Purge config files:**

```
dpkg -l | grep ^rc | awk '{print($2)}' | xargs sudo apt-get -y purge
```

- **Remove Java config and cache directory:**

```
sudo bash -c 'ls -d /home/*/.java' | xargs sudo rm -rf
```

- **Remove manually installed JVMs:**

```
sudo rm -rf /usr/lib/jvm/*
```

- **Remove Java entries, if there is still any, from the *alternatives*:**

```
for g in ControlPanel java java_vm javaws jcontrol jexec keytool mozilla-javaplugin.so orbd pack200 policytool rmid rmiregistry servertool tnameserv
```

```
unpack200 appletviewer apt extcheck HtmlConverter idlj jar jarsigner javac
javadoc javah javap jconsole jdb jhat jinfo jmap jps jrunscript jsadebugd jstack
jstat jstatd native2ascii rmic schemagen serialver wsgen wsimport xjc
xulrunner-1.9-javaplugin.so; do sudo update-alternatives --remove-all $g;
done
```

- **Search for possible remaining Java directories:**

```
sudo updatedb
sudo locate -b '\pack200'
```

If the command above produces any output like /path/to/jre1.6.0_34/bin/pack200 remove the directory that is parent of **bin**, like this: sudo rm -rf /path/to/jre1.6.0_34.

to check for Installations of Java

```
update-alternatives --config java
```