Apache Hadoop3-Single Node

Apache Hadoop
Apache hadoop is a distributed processing framework which brings 100 percent data alive

Method to Deploy Hadoop
Stand-alone: Used to install on laptop for developer
Single-node: Pseudo (Not Real) Mode
Multi-node: Fully distributed mode

Launch Instance
Name-SingleNode | Ubuntu20| t2.medium | Storage-8GB
ami-0e6b0a1f7f29fa192

Go to Security group → Inbound Rule → Edit inbound rule → Add rule → All traffic → Any whare IPv4 and IPv6.

Check the key is available or not, also check permission of it.
```
Ls -l
chmod 600 key.pem
ssh -i key.pem ubuntu@<Public-IP>
sudo apt-get update -y
```

Install Java
Hadoop is built in java
OpenJDK is open source, while Oracle JDK is intended for enterprise use. These two are the main options, although other alternatives are also available.

JVM – Java Virtual Machine, where the bytecode executes. It provides platform independence.
JRE – Java Runtime Environment, which contains runtime libraries and the JVM.
JDK – Java Development Kit, which contains development libraries and the JRE.

```
sudo apt install openjdk-8-jdk openjdk-8-jre -y
java -version
```

If Java is not installed, Update ubuntu and install it again, then check.
```
Sudo apt-get update -y
sudo apt install openjdk-8-jdk openjdk-8-jre -y
java -version
```

Create a Hadoop user for accessing HDFS and MapReduce
Instead of using the default user (i.e., ubuntu), we will create a dedicated user named **hduser** and a group called **hadoop**.

```
sudo addgroup hadoop
sudo adduser hduser --ingroup hadoop
```
Type new password, Retype, Enter, Enter, Enter, Enter, Enter, Y.
Give sudo access to hduser.
```
sudo adduser hduser sudo
su hduser
```
Showing /home/ubuntu, Now move to home directory of hduser /home/hduser.
```
cd
pwd
```

Configure Password-less SSH for localhost
Key.pem works with public and private IPs, but not with loopback IP. Localhost runs on the loopback IP, so we need to create a key for loopback IP.

```
ssh localhost
```
This will show Permission denied.

```
ssh-keygen
```
Generating a public/private RSA key pair by default location /home/hduser/.ssh/id_rsa

```
Enter Enter Enter
ssh localhost
```
This will show 'Permission denied.' We need to make the key available.

```
cd .ssh
```
Do Authorisation

```
cat id_rsa.pub >> authorized_keys
ssh localhost
```
The connection has been established without changing the IP.

```
exit
```
Connection to local host closed.

Download

Search hadoop.apache.org in browser → Download → mirrorsite → backupsite → stable → copy the link of tar.gz file.

```
wget https://dlcdn.apache.org/hadoop/common/stable/hadoop-<3.3.6>.tar.gz
ls
```

Extract the tarball file.

```
tar -xzvf hadoop-<3.3.6>.tar.gz
ls
```

Moving to proper location

It will be used by multiple users, so we will move it to the /usr directory.

```
sudo mv hadoop-<3.3.6> /usr/local/hadoop
```
If we use the sudo command, ownership will be transferred to root. Since we are working from hduser and hduser does not have access to root, we need to change the ownership to hduser

```
sudo chown hduser:hadoop -R /usr/local/hadoop
```

Configure

Setting environemnt variable

Linux level – nano .bashrc

Hadoop level – /usr/local/hadoop/etc/hadoop/hadoop-env.sh

Example: If we run 'which ls,' it will show the location of ls (i.e., /usr/bin/ls). However, we can use the ls command directly because the environment variable is set.

```
which ls
echo $PATH
nano .bashrc
```
Go to last line and Paste below lines.

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export PATH=$PATH:/usr/local/hadoop/bin/
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=/usr/local/hadoop/etc/hadoop
```
Ctrl+O, Ctrl+x

```
source .bashrc
echo $PATH
```

```
cd /usr/local/hadoop/etc/hadoop/
nano hadoop-env.sh
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_LOG_DIR=/var/log/hadoop
sudo mkdir /var/log/hadoop
```

```
sudo chown hduser:hadoop -R /var/log/hadoop
```

Higher level architecture of Hadoop

We have two components in Hadoop: Storage and Processing.
For storage, we use HDFS (Hadoop Distributed File System).
The daemons of HDFS are the NameNode and Secondary NameNode, which are master nodes, and the DataNode, which is a slave node.
For processing, we use YARN and MapReduce.
The daemons of YARN include the ResourceManager, which is the master, and the NodeManager, which is the slave.

Core Configuration of Hadoop

```
nano core-site.xml
```

Paste between the configuration tag.

```
 <property>
  <name>fs.defaultFS</name>
  <value>hdfs://localhost:54310</value>
 </property>
```

The core-site.xml file is used for cluster-wide configuration.
The property **fs.defaultFS** indicates where the NameNode is running.
We specify that the NameNode is part of **HDFS** and is running on **localhost** at port number **54310**.
In Hadoop, we use ports above 5000 to avoid conflicts.

```
nano hdfs-site.xml
```

```
 <property>
  <name>dfs.replication</name>
  <value>1</value>
 </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>
```

```
sudo mkdir -p /usr/local/hadoop_store/hdfs/namenode
sudo mkdir -p /usr/local/hadoop_store/hdfs/datanode
sudo chown -R hduser:hadoop /usr/local/hadoop_store
```

In HDFS we get by default three copies of file available as we are working on single node we have to set it on 1.
We have specified the locations where data will be saved for the NameNode and DataNode. After specifying these locations, we created the directories for both the NameNode and DataNode.

dfs.namenode.name.dir → /usr/local/hadoop_store/hdfs/namenode
dfs.datanode.data.dir → /usr/local/hadoop_store/hdfs/datanode

In HDFS, three copies of a file are created by default. Since we are working on a single node, we need to set this replication factor to 1.
These two properties specify where the data for the NameNode and DataNode will be stored, and the locations must be available. After specifying these properties, we created the necessary directories.

```
nano yarn-site.xml
 <property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
 </property>
 <property>
  <name>yarn.nodemanager.env-whitelist</name>
<value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
 </property>
```

We have specified that YARN will run MapReduce.

```
nano mapred-site.xml
 <property>
  <name>mapreduce.jobtracker.address</name>
  <value>localhost:54311</value>
 </property>
 <property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
 </property>
 <property>
  <name>yarn.app.mapreduce.am.env</name>
  <value>HADOOP_MAPRED_HOME=$HADOOP_MAPRED_HOME</value>
 </property>
 <property>
  <name>mapreduce.map.env</name>
  <value>HADOOP_MAPRED_HOME=$HADOOP_MAPRED_HOME</value>
 </property>
 <property>
  <name>mapreduce.reduce.env</name>
  <value>HADOOP_MAPRED_HOME=$HADOOP_MAPRED_HOME</value>
 </property>
```

We have specified from where job tracking will be done, which will be handled by the ResourceManager running on localhost at port 54311.
We have specified the name of the MapReduce framework, indicating that YARN will run MapReduce.

```
cd
```

Formatting Namenode

The final configuration involves formatting the NameNode. Since the NameNode is part of the storage system and it is standard practice to format storage before use, we are formatting it because the NameNode is the primary master.

```
hdfs namenode -format
```

Deploy

Start NN, SNN, DN

```
start-dfs.sh
```

Start RM, NM

```
start-yarn.sh
```

These are daemon services running in the background, so we cannot see them directly. To check their status, we can use the jps command, as these are Java processes that run on the JVM.

```
jps
```

Put data on HDFS

```
hdfs dfs -mkdir /user
hdfs dfs -mkdir /user/hduser
hdfs dfs -put hadoop-<3.3.6>.tar.gz /user/hduser
```

Check on Browser
<Public-IP>:9870 For Namenode

```
yarn jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-*examples*.jar pi 5 10
```

WebUI
Pulic-IP:9870