Configuring a Hadoop single-node test cluster

1. First download the latest version of Hadoop 3 from Apache's webpage.
2. We will do single nodcoe set up first and later extend it to a multi node one.
3. Download the GA (Generally Available) release of Hadoop, that is hadoop-3.0.0 (released in December 2017, the latest production-ready release by Apache) from the following web page:
   https://archive.apache.org/dist/hadoop/common/
4. Check whether the file download is correct by following the steps in the page:
   https://www.apache.org/info/verification.html
5. Log in to the master node on Oracle cloud, that was created earlier.
6. Run upgrade and update on terminal.

```
ubuntu@hadoopmaster: ~$ sudo apt-get update
ubuntu@hadoopmaster: ~$ sudo apt-get upgrade
```

7. Hadoop 3.0.0 requires Java 8. (See https://cwiki.apache.org/confluence/display/HADOOP/Hadoop+Java+Versions)

```
ubuntu@hadoopmaster: ~$ sudo apt-get update
ubuntu@hadoopmaster: ~$ sudo apt-get install openjdk-8-jdk
```

8. Check java version

```
ubuntu@hadoopmaster: ~$ java -version
```

   I got the following output:

```
openjdk version "1.8.0_252"
OpenJDK Runtime Environment (build 1.8.0_252-8u252-b09-1~18.04-b09)
OpenJDK 64-Bit Server VM (build 25.252-b09, mixed mode)
```

9. If there are multiple java installations, the version to be used can be configured with the steps below:

```
ubuntu@hadoopmaster: ~$ sudo update-alternatives --config java
```

I got the output:

```
There is only one alternative in link group java (providing
/usr/bin/java): /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java
Nothing to configure.
```

If your machine has multiple java versions, press enter highlighting openjdk8.

10. Set the JAVA_HOME environment variable to enable Hadoop to determine the Java installation location as below:

```
ubuntu@hadoopmaster: ~$ vi /etc/environment
```

Add
JAVA_HOME="/usr/lib/jvm/java-8-openjdk-amd64"
to the end of the file.

To test if it is set correctly, run the following command:

```
ubuntu@hadoopmaster: ~$ echo $JAVA_HOME
```

11. Next, we need to create a Hadoop user and set its privileges to be able to access HDFS and MapReduce.

```
# Add a new user group named Hadoop
ubuntu@hadoopmaster: ~$ sudo addgroup hadoop
# Add a new user named hduser to this group. set password and other
optional info when prompted
ubuntu@hadoopmaster: ~$ sudo adduser --ingroup hadoop hduser
# Log in to root user
ubuntu@hadoopmaster: ~$ sudo su root
# Open and edit the /etc/sudoers file to specify privileges for hduser
root@hadoopmaster: ~$  sudo vi /etc/sudoers
# Exit root and log in as hduser
root@hadoopmaster: ~$ sudo su hduser
hduser@hadoopmaster : ~$
```

Add this
User privilege specification for hadoop user

hduser ALL=(ALL:ALL) ALL
to the end of this file.

12. Hadoop uses Secure Shell (SSH) protocol to communicate between nodes. For single-node setup, SSH needs to be configured to localhost.
For multi-node set up, password less SSH login is to be set up from master to all slave nodes.

```
# Install an OpenSSH server
hduser@hadoopmaster : ~$ sudo apt-get install openssh-server
# Generate SSH key for hduser
hduser@hadoopmaster : ~$ ssh-keygen -t rsa -P ""
# Add the key to the list of authorized keys
hduser@hadoopmaster : ~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

Please note: The file *authorized_key*s would have to be created if not present in ~/.ssh.

13. Hadoop works only on IPv4, so we need to disable IPv6.

```
# Open /etc/sysctl.conf
hduser@hadoopmaster : ~$ sudo vi /etc/sysctl.conf
# After update, reboot the machine
hduser@hadoopmaster : ~$  sudo reboot
```

Update the file by adding the following to its end:

```
# disable ipv6
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

14. Then we need to copy the downloaded Hadoop project on our local computer to the remote instance on Oracle cloud.

I used the *pscp* (Putty Secure Copy) utility to copy Hadoop from my windows machine to the remote hadoopmaster machine being set-up on Oracle cloud.

```
C:\Users\lakshmijm\Desktop> pscp -scp -i masterkey.ppk hadoop-3.0.0.tar.gz ubuntu@168.138.4.55:/home/ubuntu
```

masterkey.ppk is the private key of the secure connection to hadoopmaster machine whose public IP address is 168.138.4.55.

If using Linux or Mac, please use appropriate commands for securely copying the project. (*scp* can be used)

15. Move the file to /home/hduser and untar it.

```
# Move Hadoop tar to /home/hduser
ubuntu@hadoopmaster : ~$  sudo mv hadoop-3.0.0.tar.gz /home/hduser/
# Decompress Hadoop project
hduser@hadoopmaster : ~$ tar -xzvf hadoop-3.0.0.tar.gz
```

16. Now we need to configure settings for Hadoop. Please run the following commands:

```
# Move Hadoop to Hadoop directory (/usr/local/hadoop)
ubuntu@hadoopmaster : ~$  sudo mv hadoop-3.0.0 /usr/local/hadoop
# Make hduser as the owner of Hadoop directory
hduser@hadoopmaster : ~$ sudo chown hduser:hadoop -R
/usr/local/hadoop
# Create directory for NameNode
hduser@hadoopmaster : ~$ sudo mkdir -p
/usr/local/hadoop/hadoopData/hdfs/namenode
# Create directory for DataNode
hduser@hadoopmaster : ~$ sudo mkdir -p
/usr/local/hadoop/hadoopData/hdfs/datanode
# Make hduser as the owner of newly created Hadoop directories
hduser@hadoopmaster : ~$ sudo chown hduser:hadoop -R
/usr/local/hadoop
```

17. Now, the .bashrc file needs to be updated:

```
hduser@hadoopmaster : ~$ sudo vi .bashrc
```

Add the following to its end:

```
# -- HADOOP VARIABLES -- #
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 HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
export PATH=$PATH:/usr/local/hadoop/bin/
# -- HADOOP VARIABLES -- #
```

18. We need to configure Hadoop configuration files now (available at the following location).

```
hduser@hadoopmaster : ~$ sudo gedit /usr/local/hadoop/etc/hadoop/*filename*.
```

Files to be edited:

a. hadoop-env.sh:

```
#The Java implementation to use
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

b. core-site.xml (specify that namenode runs on localhost)

```
<property>
        <name>fs.default.name</name>
        <value>hdfs://localhost:9000</value>
</property>
```

c. hdfs-site.xml (specify replication, namenode and datanode directories)

```
<property>
        <name>dfs.replication</name>
        <value>1</value>
</property>
<property>
        <name>dfs.namenode.name.dir</name>
        <value>file:/usr/local/hadoop/hadoopData/hdfs/namenode</value>
</property>
<property>
        <name>dfs.datanode.data.dir</name>
        <value>file:/usr/local/hadoop/hadoopData/hdfs/datanode</value>
</property>
```

d. yarn-site.xml

```
<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>
```

e. mapred-site.xml

```
<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>
```

19. Now we are ready to format (to be performed only once) and start Hadoop. You may have to reboot the machine before doing these steps, to make environmental changes come into effect.

```
# Format Namenode
hduser@hadoopmaster:/$ hdfs namenode -format
# Start HDFS daemons/services
hduser@hadoopmaster:/$ start-dfs.sh
# Start MapReduce daemons/services
hduser@hadoopmaster:/$ start-yarn.sh
```

20. We can use the jps command to check whether Hadoop has started correctly. If all is well, the jps command should output six services (as shown below) that are required to successfully run Hadoop as a single-node cluster. The output that I got is pasted for reference.

```
# Check if Hadoop started correctly
hduser@hadoopmaster:/$ jps
30165 NameNode
30581 SecondaryNameNode
31289 Jps
30859 ResourceManager
31020 NodeManager
30333 DataNode
```

21. Hadoop cluster deamons can be stopped using the command:

```
# Stop HDFS daemons/services
hduser@hadoopmaster:/$ stop-dfs.sh
# Stop MapReduce daemons/services
hduser@hadoopmaster:/$ stop-yarn.sh
hduser@hadoopmaster:/$ jps
32440 Jps
```

22. Hadoop gives WebUI for namenode and resourcemanager deamons. Working of a Hadoop cluster can be checked on the browser using the following URLs:

    http://localhost:9870 for NameNode
    http://localhost:8088 for ResourceManager