Practical no-3

Aim: Working with HBase

a) Set up an HBase cluster in a lab environment.
b) Create an HBase table and define column families.
c) Insert sample data into the table.
d) Perform CRUD operations and retrieval of data in HBase.

Set up an HBase cluster in a lab environment

Hadoop Installation in Ubuntu

Introduction

Apache Hadoop is an open-source software framework used to store, manage and process large datasets for various big data computing applications running under clustered systems. It is Java-based and uses Hadoop Distributed File System (HDFS) to store its data and process data using MapReduce. In this article, you will learn how to install and configure Apache Hadoop on Ubuntu

Install Java

```
sudo apt install openjdk-8-jdk-headless
```

Verify the installed version of Java.

```
java -version
```

Create Hadoop User and Configure Password-less SSH

Add a new user hadoop.

```
sudo adduser hadoop
```

Add the hadoop user to the sudo group.

```
sudo usermod -aG sudo hadoop
```

Install the OpenSSH server and client.

```
sudo apt install openssh-server openssh-client -y
```

When you get a prompt, respond with: keep the local version currently installed
Log in with hadoop user.

```
sudo su - hadoop
```

Generate public and private key pairs.

```
ssh-keygen -t rsa
```

(don’t add password just press enter as many times to reach normal prompt)

Add the generated public key from id_rsa.pub to authorized_keys.

```
sudo cat  ~/.ssh/id_rsa.pub  >>  ~/.ssh/authorized_keys
```

Change the permissions of the authorized_keys file.

```
sudo chmod 640  ~/.ssh/authorized_keys
```

Verify if the password-less SSH is functional.

```
ssh localhost
```

Install Apache Hadoop

Download the latest stable version of Hadoop. To get the latest version, go to Apache Hadoop official download page.

```
wget https://archive.apache.org/dist/hadoop/core/hadoop-3.1.1/hadoop-3.1.1.tar.gz
```

Extract the downloaded file.

```
tar -xvzf hadoop-3.1.1.tar.gz
```

Move the extracted directory to the /usr/local/ directory.

```
sudo mv hadoop-3.1.1 /usr/local/hadoop
```

Create directory to store system logs.

```
sudo mkdir  /usr/local/hadoop/logs
```

Change the ownership of the hadoop directory.

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

Configure Hadoop

Edit file ~/.bashrc to configure the Hadoop environment variables.

```
sudo nano ~/.bashrc
```

Add the following lines to the file. Save and close the file.

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$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 PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
```

Activate the environment variables.

```
source ~/.bashrc
```

Configure Java Environment Variables

Hadoop has a lot of components that enable it to perform its core functions. To configure these components such as YARN, HDFS, MapReduce, and Hadoop-related project settings, you need to define Java environment variables in hadoop-env.sh configuration file.

Find the Java path.

```
which javac
```

Find the OpenJDK directory.

```
readlink -f /usr/bin/javac
```

Edit the hadoop-env.sh file.

```
sudo nano $HADOOP_HOME/etc/hadoop/hadoop-env.sh
```

Add the following lines to the file. Then, close and save the file.

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_CLASSPATH+=" $HADOOP_HOME/lib/*.jar"
```

Browse to the hadoop lib directory.

```
cd /usr/local/hadoop/lib
```

Download the Javax activation file.

```
sudo wget https://jcenter.bintray.com/javax/activation/javax.activation-api/1.2.0/javax.activation-api-1.2.0.jar
```

Come back to home folder of Hadoop user

```
cd /home/hadoop
```

Verify the Hadoop version.

```
hadoop version
```

Edit the core-site.xml configuration file to specify the URL for your NameNode.

```
sudo nano $HADOOP_HOME/etc/hadoop/core-site.xml
```

Add the following lines. Save and close the file.

```
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://0.0.0.0:9000</value>
    <description>The default file system URI</description>
  </property>
</configuration>
```

Create a directory for storing node metadata and change the ownership to hadoop.

```
sudo mkdir -p /home/hadoop/hdfs/{namenode,datanode}

sudo  chown  -R  hadoop:hadoop  /home/hadoop/hdfs
```

Edit hdfs-site.xml configuration file to define the location for storing node metadata, fs-image file.

```
sudo nano $HADOOP_HOME/etc/hadoop/hdfs-site.xml
```

Add the following lines. Close and save the file.

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.name.dir</name>
<value>file:///home/hadoop/hdfs/namenode</value>
</property>
<property>
<name>dfs.data.dir</name>
```

```
<value>file:///home/hadoop/hdfs/datanode</value>
</property>
<property>
<name>dfs.permissions.enabled</name>
<value>false</value>
</property>
</configuration>
```

Edit mapred-site.xml configuration file to define MapReduce values.

```
sudo nano $HADOOP_HOME/etc/hadoop/mapred-site.xml
```

Add the following lines. Save and close the file.

```
<configuration>
   <property>
      <name>mapreduce.framework.name</name>
      <value>yarn</value>
   </property>
 <property>
    <name>yarn.app.mapreduce.am.env</name>
    <value>HADOOP_MAPRED_HOME=/usr/local/hadoop</value>
    <description>Change this to your hadoop location.</description>
  </property>
  <property>
    <name>mapreduce.map.env</name>
    <value>HADOOP_MAPRED_HOME=/usr/local/hadoop</value>
    <description>Change this to your hadoop location.</description>
 </property>
  <property>
    <name>mapreduce.reduce.env</name>
    <value>HADOOP_MAPRED_HOME=/usr/local/hadoop</value>
    <description>Change this to your hadoop location.</description>
  </property>
</configuration>
```

Edit the yarn-site.xml configuration file and define YARN-related settings.

```
sudo nano $HADOOP_HOME/etc/hadoop/yarn-site.xml
```

Add the following lines. Save and close the file.

```
<configuration>
   <property>
      <name>yarn.nodemanager.aux-services</name>
      <value>mapreduce_shuffle</value>
   </property>
</configuration>
```

Log in with hadoop user.(if you are not. This will be case once you have restarted you have computer)

```
sudo su - hadoop
```

Validate the Hadoop configuration and format the HDFS NameNode.

```
hdfs namenode -format
```

Start the Apache Hadoop Cluster

Start the NameNode and DataNode.

```
start-dfs.sh
```

Start the YARN resource and node managers.

```
start-yarn.sh
```

Verify all the running components.

```
jps
```

Access Apache Hadoop Web Interface

You can access the Hadoop NameNode and DataNode on your browser via http://localhost:9870. For example:

```
http://localhost:9870
http://localhost:8088
```

Install HBase cluster using Hadoop

Download Hbase

```
wget https://archive.apache.org/dist/hbase/2.4.1/hbase-2.4.1-bin.tar.gz
```

Unzip it by executing command

```
tar -xvf hbase-2.4.1-bin.tar.gz
```

Rename directory to hbase

```
mv hbase-2.4.1 hbase
```

It will unzip the contents, and it will create hbase-2.4.15 directory in the location /home/Hadoop
Now rename the directory to hbase

Open hbase-env.sh in hbase/conf and assign the JAVA_HOME path

```
sudo nano hbase/conf/hbase-env.sh
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

Edit the .bashrc file

```
sudo nano ~/.bashrc
export HBASE_HOME=/home/hadoop/hbase
export PATH=$PATH:$HBASE_HOME/bin
```

Read the edited bashrc file to the running memory

```
 source ~/.bashrc
```

Open HBase-site.xml and mention the below properties in the file.

```
sudo nano hbase/conf/hbase-site.xml
```

```
<property>
<name>hbase.rootdir</name>
<value>hdfs://localhost:9000/hbase</value>
</property>
<property>
<name>hbase.cluster.distributed</name>
<value>true</value>
</property>
<property>
<name>hbase.zookeeper.quorum</name>
```

```
<value>localhost</value>
</property>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>hbase.zookeeper.property.clientPort</name>
<value>2181</value>
</property>
<property>
<name>hbase.zookeeper.property.dataDir</name>
<value>/home/hadoop/hbase/zookeeper</value>
</property>
```

Properties Explanation

1. Setting up Hbase root directory in this property, for distributed set up we have to set this property
2. ZooKeeper quorum property should be set up here
3. Replication set up done in this property. By default we are placing replication as 1.In the fully distributed mode, multiple data nodes present so we can increase replication by placing more than 1 value in the dfs.replication property
4. Client port should be mentioned in this property
5. ZooKeeper data directory can be mentioned in this property

Start Hadoop daemons first and after that start HBase daemons as shown below:

```
start-dfs.sh
start-yarn.sh
start-hbase.sh
```

Create an HBase table and define column families

```
create 'emp', 'pri_data', 'pro_data'
```

Insert sample data into the table

```
put 'emp', '1', 'pri_data:name', 'Andy'

put 'emp', '1', 'pri_data:age', '22'

put 'emp' ,'1', 'pro_data:post', 'asst. manager'

put 'emp' ,'1', 'pro_data:salary', '40k'

put 'emp' ,'2', 'pri_data:name', 'Icarus'

put 'emp' ,'2', 'pri_data:age', '22'

put 'emp' ,'2', 'pro_data:post', 'manager'
```

Perform CRUD operations and retrieval of data in HBase.

```
get 'emp' ,'1'

get 'emp' ,'2'

delete 'emp', '1', 'pri_data:city'  (this will delete only city of employee 1)
```

```
deleteall ‘emp’,’1’ (this will delete the first employee)
scan emp
list
```