Setting Up Hadoop Cluster

Goal:

We are making a HDFS cluster of one namenode and two datanodes. Our aim is to distribute the nodes between 2 physical machines, such that each machine contains at least 1 datanode.One of the machines will be hosting the namenode currently which will be later augmented with another standby namenode once the cluster starts getting heavy in data processing.

Given Configuration

Two laptops with ubuntu as base OS.

Ubuntu comes with a virtual machine named as KVM.

Kernel-based Virtual Machine (**KVM**) is an open source virtualization technology built into Linux®. Specifically, **KVM** lets you turn Linux into a hypervisor that allows a host machine to run multiple, isolated virtual environments called guests or virtual machines (VMs). **KVM** is part of Linux.

Plan

1. Create VM - one per each node
2. Host OS is chosen to be RedHat
3. Setup cluster
4. Setup Hadoop

Get going….

Prerequisite For both Machines:

Software

- Virtual Machine(KVM)

```
$ sudo apt-get install virt-manager ssh-askpass-gnome --no-install-recommends
```

- Redhat(7.5)

https://developers.redhat.com/products/rhel/download

- Java(JDK)

http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

- Hadoop(v2.7.3)

https://archive.apache.org/dist/hadoop/core/

STEPS NEEDED FOR HDFS SET-UP IN REDHAT: (will performe in both laptops)

- **Setup the Virtual Machines so that they are connected with the same network(LAN)**
- **In the Virtual Machine:** Setup the Network in eth/ens(network card) and also make this connection Bridged.(Try to do at the time of installation of OS)

- **Setting up the hostname:**

  Step 1: Set the HOSTNAME=namenode.cluster1.com.

  To be done on all the namenode and datanode.

  Step 2: Make your local DNS on every system this file will look like this:

  /etc/hosts

  ```
  192.168.0.254      namenode.cluster1.com
  192.168.0.201      datanode1.cluster1.com
  192.168.0.202      datanode2.cluster1.com
  192.168.0.203      datanode3.cluster1.com
  ```

  Note: IP's are to be changed according to the respective system.
  Make Sure that all the VM would ping each other.

  The changes in the 'hosts' file is for the system to be aware of the other nodes which are trying to connect to it.

- Make IP Address Static

  (For making hdfs cluster permanent we need some static ip to all our nodes so that if we will change our local network still hadoop cluster will not be affeced.)

  Navigate to the file:

  ```
  []# vim /etc/sysconfig/network-scripts/ifcfg-ens3
  ```

  Change the bootproto file from
  BOOTPROTO="dhcp" → "static"

  And then add the IP of the system
  IPADDR=192.168.###.###

  ONBOOT="yes" (check it once otherwise changes will not become permanent)

  Set the IP Address Save & Quit

  (**Pro Tip!** When you don't bridge the connection, and when you make the IP Static, the VM's will stop interacting with each other and you won't be able to ping each other)

Step 1: Setting up the Java Path:

Download the jdk.rpm file from the given java link and then install it by migrating it to the Downloads folder and open terminal in the downloads folder and then install it by []# rpm -ivh jdk*

[]# /root/.bashrc (This entry will make java path permanent)

#It is the hidden and sensitive file of redhat. Open it and do only changes i.e required otherwise Redhat OS will corrupt.

Add to the last line in the bashrc file:

```
JAVA_HOME=/usr/java/jdk1.8.0_121
```

IMPORTANT: The Java version which we have used here Jdk - 221.

Please make the necessary changes in according to your version of Java where ever the Java Path is specified. And check with

[]# java -version

If the newest Java version is not shown, then remove the other version by:

[]# yum remove java*   → It will automatically remove the older version of Java.

and check again the Java Version.

Step 2: Installing apache hadoop from tar:

In the downloads folder:

1. []# `tar  -xvzf  hadoop-2.7.3.tar.gz`
   a. This command is used for installation of the Hadoop 2.7.3 .
2. []# `mv hadoop-2.7.3 /hadoop2`

   b. Moving file into the root folder for security reasons.
3. For Hadoop Path setup, make changes in bashrc file:

   Add the following code at the end of the file:

```
HADOOP_HOME=/hadoop2
PATH=$JAVA_HOME/bin:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH
export PATH
```

Step 3: Setting up the NameNode Apache Hadoop:

1. Move to the directory of extracted Hadoop file.

   []# `cd /hadoop2/etc/hadoop/`

   Use #ls command here to check whether hadoop is installed correctly or not.
2. Setup the Hadoop Environment in hadoop-env.sh

```
# The only required environment variable is JAVA_HOME.  All others are
# set JAVA_HOME in this file, so that it is correctly defined on
export JAVA_HOME=/usr/java/jdk1.8.0_221
```

   Note: Comment all the other lines except the two JAVA Path
3. Make changes in the file hdfs-site.xml

   By default hdfs-site.xml remains vacant.For clustering, Property TAG needs to be added with Name and Value pair-tags.

```
<configuration>
      <property>
            <name>dfs.namenode.name.dir</name>
            <value>/name</value>
      </property>
```

```
</configuration>
```

Exit and run ([]# mkdir /name) for the storage of metadata of all datanode in hdfs.

(This changes are done to tell namenode where namenode will store information of all datanode .Here /name is the folder where namenode will place metadata of all datanode)

4. Make changes in the file core-site.xml

   By default core-site.xml remains vacant.For clustering, Property TAG needs to be added with Name and Value pair-tags.

```
<configuration>
            <property>
                  <name>fs.defaultFS</name>
                  <value>hdfs://192.168.10.120:10003</value>
            </property>
</configuration>
```

   NOTE: The IP Address mentioned above in the name tag is the IP Address of the Name Node of HDFS.

   (IP address mentioned here will tell our hdfs cluster that you are the namenode.)

5. Format namenode and start service (As we all konw that formatting is required before using any data storage or file system)

```
[]# hdfs  namenode -format
```

   NOTE: The IP Address mentioned above in the name tag is the IP Address of the Name Node of HDFS.

6. Start the service of namenode

```
[]# hadoop-daemon.sh  start  namenode
```

   Used to start the Namenode and make it running.We can check its status by: []# jps:
   {JPS (Java Virtual Machine Process Status Tool) is a command is used to check all the Hadoop daemons like NameNode, DataNode, ResourceManager, NodeMa   nager etc. which are running on the machine}

   The Output of the JPS is:

```
      5540 DataNode/NameNode
      5710 Jps
```

   This shows that our namenode is up and running.

5540 is a process number(it will be different for different cluster)

sometimes it might happen that when you start namenode it will show that some process is running but the namenode would not start. So, use command #kill 5540 to kill the process and then stop the node and then restart the node.

   We can also check the status by:

```
[]# hadoop fsck /
```

Step 4: Setting up the DataNode Apache Hadoop:

Steps 1,2 & 3 are the same as that of the Namenode

1. Setting up hdfs-site.xml

   By default hdfs-site.xml remains vacant.For clustering, Property TAG needs to be added with Name and Value pair-tags.

```
<configuration>
            <property>
                  <name>dfs.datanode.data.dir</name>
                  <value>/data</value>
            </property>
</configuration>
```

   Exit and run ([]# mkdir /data) for the storage of actual data in hdfs.

2. Setting up core-site.xml

   By default core-site.xml remains vacant.For clustering, Property TAG needs to be added with Name and Value pair-tags.

```
<configuration>
            <property>
                  <name>fs.defaultFS</name>
                  <value>hdfs://192.168.10.120:10003</value>
            </property>
</configuration>
```

   Note:The IP Address mentioned is the IP Address of the NameNode

3. Starting the Datanode

```
[]# hadoop-daemon.sh  start  datanode
```

Step 5: Starting the HDFS Cluster:

1. Run to command to test the working of the nodes

```
#[] jps
```

   The node is up and running when it shows some value like:

```
5540 DataNode/NameNode
5710 Jps
```

2. 2. With this, Our namenode and datanode are ready and we can see our cluster running by entering the following address in the browser of the namenode :

   Localhost:50070

   Or use the command

```
[]# hdfs dfsadmin -report
```

SUMMARY

step1	Laptop 1 with base OS ubuntu	Laptop 2 with base OS ubuntu
2	Install kVM and inside kvm install 2 redhat OS(planning to make 1 OS as namenode and another as datanode)	Install KVM and inside KVM install 1 redhat OS(Planning to make it as datanode)

3	1 OS as namenode	1 OS as Datanode 1	1 OS asDatanode 2
4	Make ip as static.	Make IP as static.	Make IP as static.
5	Do changes in /etc/hosts	Do changes in /etc/hosts	Do changes in /etc/hosts
6	Install jdk and hadoop also make it permanent by giving entry in /root/.bashrc	Install jdk and hadoop also make it permanent by giving entry in /root/.bashrc	Install jdk and hadoop also make it permanent by giving entry in /root/.bashrc
7	Do changes in hadoop-env.sh file . All previous entries will commented out and write java-path here .	Do changes in hadoop-env.sh file . All previous entries will commented out and write java-path here .	Do changes in hadoop-env.sh file . All previous entries will commented out and write java-path here .
8	Do changes in hdfs-site.xml Name-dfs.namenode.name.dir Value-/name [make a folder also # mkdir /name]	Do changes in hdfs-site.xml Name-dfs.datanode.data.dir Value-/data [make a folder also # mkdir /data]	Do changes in hdfs-site.xml Name- dfs.datanode.data.dir Value-/data [make a folder also # mkdir /data]

9	Do changes in core-site.xml Name-fs.defaultFS Value-hdfs://IP address of namenode:10003	Do changes in core-site.xml Name-fs.defaultFS Value-hdfs://IP address of namenode:10003	Do changes in core-site.xml Name-fs.defaultFS Value-hdfs://IP address of namenode:10003
10	#iptables -F #setenforce 0 For removing restriction of firewall .	#iptables -F #setenforce 0 For removing restriction of firewall .	#iptables -F #setenforce 0 For removing restriction of firewall .
11	`#hdfs namenode -format`	-	-
12	#hadoop-daemon.sh start namenode	#hadoop-daemon.sh start datanode	#hadoop-daemon.sh start datanode
13	#jps To check namenode is up or not.	#jps To check datanode is up or not.	#jps To check datanode is up or not.
14	#hdfs dfsadmin -report Or open any browser and check for **`Localhost:50070`** This is the url of apche hadoop where we can see how many datanode are connected and whether our namenode is active or not.	-	-