

## **Exp.1 Downloading and installing Hadoop, Understanding different Hadoop modes, Startup scripts, Configuration files.**

### **AIM:**

To Download and install Hadoop, Understanding different Hadoop modes, Startup scripts, Configuration files.

### **Procedure:**

#### **Step 1 : Install Java Development Kit**

The default Ubuntu repositories contain Java 8 and Java 11 both. But, Install Java 8 because hive only works on this version. Use the following command to install it.

```
$sudo apt update&&sudo apt install openjdk-8-jdk
```

#### **Verify the Java version**

Once installed, verify the installed version of Java with the following command:

```
$ java -version
```

#### **Step 3: Install SSH**

SSH (Secure Shell) installation is vital for Hadoop as it enables secure communication between nodes in the Hadoop cluster. This ensures data integrity, confidentiality, and allows for efficient distributed processing of data across the cluster.

```
$sudo apt install ssh
```

#### **Step 4 : Create the hadoop user :**

All the Hadoop components will run as the user that you create for Apache Hadoop, and the user will also be used for logging in to Hadoop's web interface.

Run the command to create user and set password:

```
$ sudo adduser hadoop
```

#### **Step 5 : Switch user**

Switch to the newly created hadoop user:

```
$ su - hadoop
```

#### **Step 6 : Configure SSH**

Now configure password-less SSH access for the newly created hadoop user, so didn't enter the key to save file and passphrase. Generate an SSH keypair (generate Public and Private Key Pairs)first

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

#### **Step 7 : Set permissions :**

Next, append the generated public keys from id\_rsa.pub to authorized\_keys and set proper permission:

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

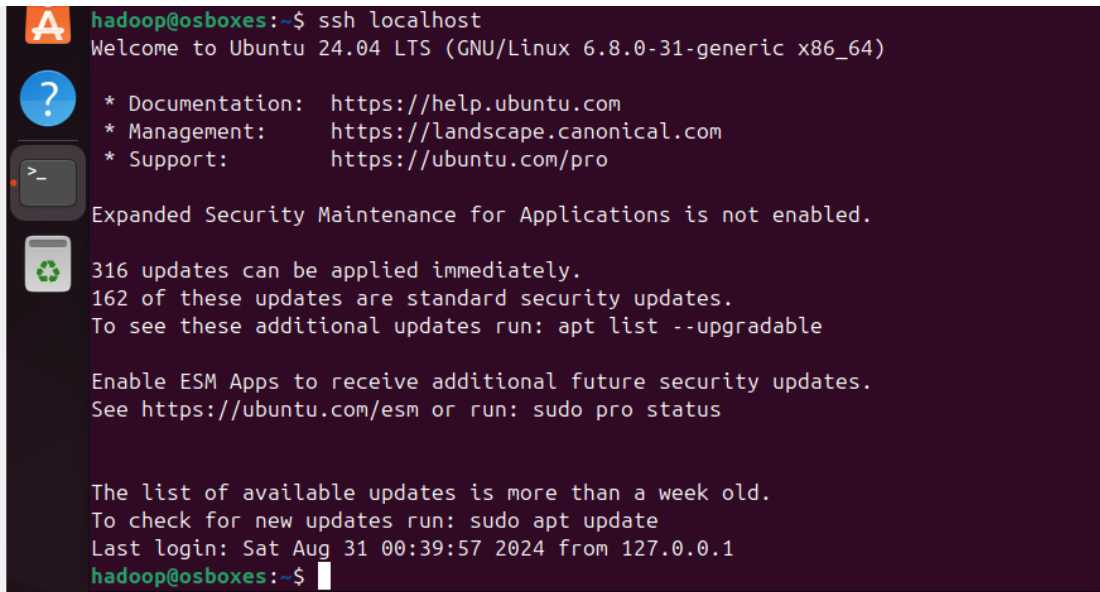
```
$ chmod 640 ~/.ssh/authorized_keys
```

**Step 8 : SSH to the localhost**

Next, verify the password less SSH authentication with the following command:

**\$ ssh localhost**

You will be asked to authenticate hosts by adding RSA keys to known hosts. Type yes and hit Enter to authenticate the localhost:



```
hadoop@osboxes:~$ ssh localhost
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-31-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

316 updates can be applied immediately.
162 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Last login: Sat Aug 31 00:39:57 2024 from 127.0.0.1
hadoop@osboxes:~$
```

**Step 9 : Switch user**

Again switch to hadoop. So, First, change the user to hadoop with the following command:

**\$ su-hadoop**

**Step 10 : Install hadoop**

Next, download the latest version of Hadoop using the wget command:

**\$ wget <https://downloads.apache.org/hadoop/common/hadoop-3.3.6/hadoop-3.3.6.tar.gz>**

Once downloaded, extract the downloaded file:

**\$ tar -xvzf hadoop-3.3.6.tar.gz**

Next, rename the extracted directory to hadoop:

empty before retry.

**\$ nano ~/.bashrc**

Append the below lines to file.

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/home/hadoop/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 HADOOP_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"
```

Save and close the file. Then, activate the environment variables with the following command:

**s\$ source ~/.bashrc**

Next, open the Hadoop environment variable file:

**\$ nano \$HADOOP\_HOME/etc/hadoop/hadoop-env.sh**

Search for the “export JAVA\_HOME” and configure it.

**JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64**

```
File Edit View Search Terminal Help
GNU nano 7.2 /home/hadoop/hadoop/etc/hadoop/hadoop-env.sh *
##
## Precedence rules:
##
## {yarn-env.sh|hdfs-env.sh} > hadoop-env.sh > hard-coded defaults
##
## {YARN_xyz|HDFS_xyz} > HADOOP_xyz > hard-coded defaults
##
# Many of the options here are built from the perspective that users
# may want to provide OVERWRITING values on the command line.
# For example:
#
# JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
#
# Therefore, the vast majority (BUT NOT ALL!) of these defaults
# are configured for substitution and not append. If append
# is preferable, modify this file accordingly.
###
# Generic settings for HADOOP
###
# Technically, the only required environment variable is JAVA_HOME.
# All others are optional. However, the defaults are probably not
# preferred. Many sites configure these options outside of Hadoop,
# such as in /etc/profile.d
#
# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
File Name to Write: /home/hadoop/hadoop/etc/hadoop/hadoop-env.sh
^C Help ^M-D DOS Format ^M-A Append ^M-B Backup File
^C Cancel ^M-W Mac Format ^M-B Prepend ^M-T Browse
```

Save and close the file when you are finished.

**Step 11 : Configuring Hadoop :**

First, you will need to create the namenode and datanode directories inside the Hadoop user home directory. Run the following command to create both directories:

```
$ cd hadoop/
```

```
$mkdir -p ~/hadoopdata/hdfs/{namenode,datanode}
```

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

Change the following name as per your system hostname:

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

Save and close the file.

Then, edit the hdfs-site.xml file:

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

- Change the NameNode and DataNode directory paths as shown below:

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

  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:///home/hadoop/hadoopdata/hdfs/namenode</value>
  </property>

  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:///home/hadoop/hadoopdata/hdfs/datanode</value>
  </property>
</configuration>
```

- Then, edit the mapred-site.xml file:  

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

- Make the following changes:

```
<configuration>
  <property>
    <name>yarn.app.mapreduce.am.env</name>
    <value>HADOOP_MAPRED_HOME=$HADOOP_HOME/home/hadoop/hadoop/bin/hadoop</value>
  </property>
  <property>
    <name>mapreduce.map.env</name>
    <value>HADOOP_MAPRED_HOME=$HADOOP_HOME/home/hadoop/hadoop/bin/hadoop</value>
  </property>
  <property>
    <name>mapreduce.reduce.env</name>
    <value>HADOOP_MAPRED_HOME=$HADOOP_HOME/home/hadoop/hadoop/bin/hadoop</value>
  </property>
</configuration>
```

- Then, edit the yarn-site.xml file:  
**\$nano \$HADOOP\_HOME/etc/hadoop/yarn-site.xml**
- Make the following changes:

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

Save the file and close it .

## Step 12 – Start Hadoop Cluster

Before starting the Hadoop cluster. You will need to format the Namenode as a hadoop user. Run the following command to format the Hadoop Namenode:

```
$hdfs namenode -format
```

Once the namenode directory is successfully formatted with hdfs file system, you will see the message “Storage directory /home/hadoop/hadoopdata/hdfs/namenode has been successfully formatted “

Then start the Hadoop cluster with the following command.

```
$ start-all.sh
```

```
hadoop@osboxes:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
localhost: namenode is running as process 8126. Stop it first and ensure /tmp/hadoop-hadoop-namenode.pid file is empty before retry.
Starting datanodes
localhost: datanode is running as process 8271. Stop it first and ensure /tmp/hadoop-hadoop-datanode.pid file is empty before retry.
Starting secondary namenodes [osboxes]
osboxes: secondarynamenode is running as process 8462. Stop it first and ensure /tmp/hadoop-hadoop-secondarynamenode.pid file is empty before retry.
Starting resource manager
resource manager is running as process 8728. Stop it first and ensure /tmp/hadoop-hadoop-resource manager.pid file is empty before retry.
Starting nodemanagers
localhost: nodemanager is running as process 8868. Stop it first and ensure /tmp/hadoop-hadoop-nodemanager.pid file is empty before retry.
```

You can now check the status of all Hadoop services using the jps command:

\$ jps

```
hadoop@osboxes:~$ jps
10898 Jps
8868 NodeManager
8728 ResourceManager
9212 RunJar
8126 NameNode
8462 SecondaryNameNode
8271 DataNode
```

### Step 13 – Access Hadoop Namenode and Resource Manager

- First we need to know our ipaddress, In Ubuntu we need to install net-tools to run ipconfig command,

If you installing net-tools for the first time switch to default user:

**\$sudo apt install net-tools**

- Then run ifconfig command to know our ip address:

```
hadoop@osboxes:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::a00:27ff:fe56:6fa2 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:56:6f:a2 txqueuelen 1000 (Ethernet)
    RX packets 458730 bytes 616003488 (616.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 171067 bytes 18436556 (18.4 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 21342 bytes 2170633 (2.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 21342 bytes 2170633 (2.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

ifconfig

Here my ip address is 192.168.1.6.

- To access the Namenode, open your web browser and visit the URL <http://your-server-ip:9870>.
- You should see the following screen:  
<http://192.168.1.6:9870>

Overview 'localhost:9000' (active)

Started:	Sun Sep 10 13:08:22 +0530 2023
Version:	3.3.6, r1be78238728da9266a4f88195058f08fd012bf9c
Compiled:	Sun Jun 18 13:52:00 +0530 2023 by ubuntu from (HEAD detached at release-3.3.6-RC1)
Cluster ID:	CID-dc5a1253-b0cd-4686-a807-fd0ddfd4c5a9a
Block Pool ID:	BP-1272319295-127.0.1.1-1694331447796

### Summary

Security is off.  
Safemode is off.  
1 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 1 total filesystem object(s).  
Heap Memory used 69.85 MB of 107 MB Heap Memory. Max Heap Memory is 748 MB.  
Non Heap Memory used 51.89 MB of 55.44 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity: 24.44 GB

To access Resource Manage, open your web browser and visit the URL <http://your-server-ip:8088>. You should see the following screen:

<http://192.168.16:8088>

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running
0	0	0	0	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decom
1	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Min
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCore:

Show 20 entries

ID	User	Name	Application Type	Application Tags	Queue	Application Priority	StartTime	LaunchTime	Finis
Showing 0 to 0 of 0 entries									

## Step 14 – Verify the Hadoop Cluster

At this point, the Hadoop cluster is installed and configured. Next, we will create some directories in the HDFS filesystem to test the Hadoop.

Let's create some directories in the HDFS filesystem using the following command:

```
$ hdfsdfs -mkdir /test1
$ hdfsdfs -mkdir /logs
```



Next, run the following command to list the above directory:

```
$ hdfs dfs -ls /
```

You should get the following output:

```
hadoop@osboxes:~$ hdfs dfs -ls /
Found 10 items
drwxr-xr-x - hadoop supergroup      0 2024-09-18 03:06 /hive
drwxr-xr-x - hadoop supergroup      0 2024-08-31 01:08 /logs
drwxr-xr-x - hadoop supergroup      0 2024-09-02 03:27 /new_output
drwxr-xr-x - hadoop supergroup      0 2024-09-04 12:09 /pig_output_data
drwxr-xr-x - hadoop supergroup      0 2024-09-04 11:46 /piginput
drwxr-xr-x - hadoop supergroup      0 2024-09-18 05:23 /tmp
drwxr-xr-x - hadoop supergroup      0 2024-09-04 12:09 /udfs
drwxr-xr-x - hadoop supergroup      0 2024-09-18 03:10 /user
drwxr-xr-x - hadoop supergroup      0 2024-09-02 02:25 /weatherdata
drwxr-xr-x - hadoop supergroup      0 2024-09-01 10:27 /word_count_in_python
hadoop@osboxes:~$
```

Also, put some files to hadoop file system. For the example, putting log files from host machine to hadoop file system.

```
$ hdfs dfs -put /var/log/* /logs/
```

You can also verify the above files and directory in the Hadoop Namenode web interface.

Go to the web interface, click on the Utilities => Browse the file system. You should see your directories which you have created earlier in the following screen:

The screenshot shows the Hadoop web interface at 192.168.1.6:9870/explorer.html#/ . The 'Utilities' menu is expanded, and 'Browse the file system' is selected. The 'Browse Directory' page shows the root directory with a search bar and a table of files. The table lists two entries: 'logs' and 'test1', both with permissions 'drwxr-xr-x', owned by 'hadoop' and 'supergroup', with a size of '0 B' and last modified on 'Sep 10 13:19' and 'Sep 10 13:16' respectively. The 'Previous' and 'Next' buttons are visible at the bottom of the table.

## Step 15 – Stop Hadoop Cluster

To stop the Hadoop all services, run the following command:

```
$ stop-all.sh
```



```
hadoop@osboxes:~$ stop-all.sh
WARNING: Stopping all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: Use CTRL-C to abort.
Stopping namenodes on [localhost]
Stopping datanodes
Stopping secondary namenodes [osboxes]
Stopping nodemanagers
Stopping resourcemanager
```