# HADOOP SINGLE SYSTEM INSTALLATION

1. **Login as Root**

   **$sudo su**

2. **Adding a dedicated Hadoop system user called hduser**

   Its better if we have dedicated Hadoop user for running hadoop.It is recommended because we can have entire hadoop framework seperated from other software applications and have a seperate environment.

3. **Create a Group called hadoop**

   **#sudo addgroup hadoop**

4. **Create an User hduser**

   **#sudo adduser hduser**
   Give password as "hadoop" and conform it.Press enter and give yes.

5. **Add hduser to hadoop group**

   **#sudo adduser hduser hadoop**

6. **Add the 'hduser' to sudoers list so that hduser can do admin tasks.**

   **$sudo visudo**

   Add given line under
   ##Allow member of group sudo.
   **hduser ALL=(ALL)ALL**
   Press ctrl+x, Y enter
   This will add the user hduser and the group hadoop to your local machine.
7. **Logout Your System and login again as hduser.**

8. **Configuring SSH**

   Hadoop requires SSH access to manage its nodes, i.e. remote machines plus your local machine if you want to use Hadoop on it (which is what we want to do in this short tutorial). For our single-node setup of Hadoop, we therefore need to configure SSH access to localhost for the hduser user we created in the previous section.
   I assume that you have SSH up and running on your machine and configured it to allow SSH public key authentication. If not, there are several guides available.
   First, we have to generate an SSH key for the hduser user.

   #Install ssh server on your computer
   **hduser@ubuntu:~$ sudo apt-get install openssh-server**

REFERENCE :KAUSHIK SHAKKARI
EMAIL : kaushik.shakkari@gmail.com

Enter Password(hadoop) and Y to continue.

If this did not work, then install openssh-server using Ubuntu Software center by searching for openssh-server.

## 9. Generate SSH for communication

**hduser@ubuntu:~$ ssh-keygen**

**Just press Enter for what ever is asked.**
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hduser/.ssh/id_rsa):
Created directory '/home/hduser/.ssh'.
Your identification has been saved in /home/hduser/.ssh/id_rsa.
Your public key has been saved in /home/hduser/.ssh/id_rsa.pub.
The key fingerprint is: 9b:82:ea:58:b4:e0:35:d7:ff:19:66:a6:ef:ae:0e:d2hduser@localhost
The key's randomart image is:
[...snipp...]
**hduser@ubuntu:~$**

The final step is to test the SSH setup by connecting to your local machine with the hduser user. The step is also needed to save your local machine's host key fingerprint to the hduser user's known_hosts file. If you have any special SSH configuration for your local machine like a non-standard SSH port, you can define host-specific SSH options in $HOME/.ssh/config (see man ssh_config for more information).

### Copy Public Key to Authorized_key file & edit the permission

#now copy the public key to the authorized_keys file, so that ssh should not require passwords every time

**hduser@ubuntu:~$cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys**

#Change permissions of the authorized_keys fie to have all permissions for hduser

**hduser@ubuntu:~$chmod 700 ~/.ssh/authorized_keys**

### Start SSH

If ssh is not running, then run it by giving the below command

**hduser@ubuntu:~$ sudo /etc/init.d/ssh restart**

**Enter your Password(hadoop)**

### Test Your SSH Connectivity
**hduser@ubuntu:~$ ssh localhost**

Type '**Yes**', when asked for. You should be able to connect without password. If you are asked to enter password here, then something went wrong. Please check your steps.

### 11. Disable IPV6

Hadoop and IPV6 do not agree on the meaning of 0.0.0.0 address, thus it is advisable to disable IPV6 adding the following lines at the end of /etc/sysctl.conf

**hduser@ubuntu:~$ sudo vim /etc/sysctl.conf**

**Enter Your Password: hadoop**

**# disable ipv6**

**net.ipv6.conf.all.disable_ipv6 = 1**

**net.ipv6.conf.default.disable_ipv6 = 1**

**net.ipv6.conf.lo.disable_ipv6 = 1**

**Check if IPv6 is disabled.**

After a system reboot the output of

**hduser@ubuntu:~$ cat /proc/sys/net/ipv6/conf/all/disable_ipv6**

should be 1, meaning that IPV6 is actually disabled. Without reboot it would be showing you 0.

## Hadoop Installation

### 1. Download Hadoop

For this tutorial, I am using hadoop- 2.7.3.tar.gz, but it should work with any other version.

Download hadoop-2.7.3.tar.gz and save it to hduser/Desktop.

### 2. move the zip file to /usr/local/

Use Terminal(Ctrl+Alt+T)

**$ sudo mv ~/Desktop/hadoop-2.7.3.tar.gz /usr/local/**
**Enter password: hadoop**
**$ cd /usr/local**
**sudo tar -xvf hadoop-2.7.3.tar.gz**
**sudo rm hadoop-2.7.3.tar.gz**
**sudo ln -s hadoop-2.7.3 hadoop**
**sudo chown -R hduser:hadoop hadoop-2.7.3**
**sudo chmod 777 hadoop-2.7.3**

3. **Edit hadoop-env.sh and configure Java.**
Add the following to /usr/local/hadoop/etc/hadoop/hadoop-env.sh by removing

export JAVA_HOME=${JAVA_HOME}

and add

**$ sudo vim /usr/local/hadoop/etc/hadoop/hadoop-env.sh**
**export HADOOP_OPTS=-Djava.net.preferIPv4Stack=true**
**export HADOOP_HOME_WARN_SUPPRESS="TRUE"**
**export JAVA_HOME=/usr/local/java/jdk1.8.0_91**

**First Export is to disable ipv6**

**Please Note:**
In hadoop 2.6,the location is /usr/local/hadoop/conf/hadoop-env.sh.
But in 2.7 there is no conf folder. In hadoop its is present in /etc/bin

4. **Update $HOME/.bashrc**

Add the following lines to the end of the $HOME/.bashrc file of user hduser. If you use a shell other than bash, you should of course update its appropriate configuration files instead of .bashrc

$ vim ~/.bashrc
#type :$ to go to the last line and then press I to switch to Insert mode

# Set Hadoop-related environment variables
**export HADOOP_HOME=/usr/local/hadoop**
**export HADOOP_PREFIX=/usr/local/hadoop**
**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_CONF_DIR=${HADOOP_HOME}/etc/hadoop**

**# Native Path**
**export HADOOP_COMMON_LIB_NATIVE_DIR=${HADOOP_PREFIX}/lib/native**

```
        export HADOOP_OPTS="-Djava.library.path=$HADOOP_PREFIX/lib"

# Set JAVA_HOME (we will also configure JAVA_HOME directly for Hadoop later on)
export JAVA_HOME=/usr/local/java/jdk1.8.0_91
# Some convenient aliases and functions for running Hadoop-related commands

unaliasfs&> /dev/null
aliasfs="hadoop fs"
unaliashls&> /dev/null
aliashls="fs -ls"


# If you have LZO compression enabled in your Hadoop cluster and
# compress job outputs with LZOP (not covered in this tutorial):
# Conveniently inspect an LZOP compressed file from the command
# line; run via:
#
# $ lzohead /hdfs/path/to/lzop/compressed/file.lzo
#
# Requires installed 'lzop' command.
# lzohead () { hadoopfs -cat $1 | lzop -dc | head -1000 | less }
# Add Hadoop bin/ directory to PATH
        export PATH=$PATH:$HADOOP_HOME/bin:$PATH:$JAVA_HOME/bin:
        $HADOOP_HOME/sbin
```

You need to close the terminal and open a new terminal to have the bash changes into effect. The shortcut to open the terminal is (Ctrl+Atl+t).

5. **Update yarn-site.xml**

   **sudo vim /usr/local/hadoop/etc/hadoop/yarn-site.xml**

   Add the following snippets between the <configuration> ... </configuration> tags

```
<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>
```

6. **Update core-site.xml file**

   **$ sudo vim /usr/local/hadoop/etc/hadoop/core-site.xml**

Add the following snippets between the <configuration> ... </configuration> tags

```
<property>

<name>hadoop.tmp.dir</name>
<value>/app/hadoop/tmp</value>
<description>A base for other temporary directories.</description>
</property>

<property>
 <name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
<description>The name of the default file system.
        A URI whose scheme and authority determine the FileSystem
        implementation.  The uri's scheme determines the config
        property (fs.SCHEME.impl) naming theFileSystem
        implementation class. The uri's authority is used to determine
        the host, port, etc. for a filesystem.
</description>

        </property>
```

**Note:** In hadoop 2.6 location is /usr/local/hadoop/etc/hadoop/yarn-site.xml

7. **Create the above temp folder and give appropriate permission**

```
sudo mkdir -p /app/hadoop/tmp
sudo chown hduser:hadoop -R /app/hadoop/tmp
sudo chmod 750 /app/hadoop/tmp
```

8. **Create mapred-site.xml file from mapred-site.xml.template**

```
$ sudo cp  /usr/local/hadoop/etc/hadoop/mapred-site.xml.template
/usr/local/hadoop/etc/hadoop/mapred-site.xml
```

Add the following to /usr/local/hadoop/etc/hadoop/mapred-site.xml between<configuration> ... </configuration>

```
$ sudo vim  /usr/local/hadoop/etc/hadoop/mapred-site.xml
 <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
        </property>
 <property>
        <name>mapreduce.jobhistory.address</name>
```

```
        <value>localhost:10020</value>
        <description>Host and port for Job History Server (default
        0.0.0.0:10020)</description>
</property>
```

9. **Create a temporary directory which will be used as base location for DFS.**

Now we create the directory and set the required ownerships and permissions:\

```
sudo mkdir -p /usr/local/hadoop/tmp

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

sudo chmod 777 /usr/local/hadoop/tmp

sudo mkdir -p /usr/local/hadoop/yarn_data/hdfs/namenode

sudo mkdir -p /usr/local/hadoop/yarn_data/hdfs/datanode

sudo chmod 777 /usr/local/hadoop/yarn_data/hdfs/namenode

sudo chmod 777 /usr/local/hadoop/yarn_data/hdfs/datanode

sudo chown hduser:hadoop -R /usr/local/hadoop/yarn_data/hdfs/namenode

sudo chown hduser:hadoop -R /usr/local/yarn_data/hdfs/datanode
```

If you forget to set the required ownerships and permissions, you will see a java.io.IOException when you try to format the name node in the next section).

10. **Update hdfs-site.xml file**

```
$ sudo vim /usr/local/hadoop/etc/hadoop/hdfs-site.xml
```

Add the following to /usr/local/hadoop/conf/hdfs-site.xml between<configuration> ... </configuration>

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

### 11. Format your namenode

Open a new Terminal as the hadoop command will not work

Format hdfs cluster with below command

**$ hadoop namenode -format**

If the format is not working, double check your entries in .bashrc file. The .bashrc updating come into force only if you have opened a new terminal.

### 12. Starting your single-node cluster

Congratulations, your Hadoop single node cluster is ready to use. Test your cluster by running the following commands.

**$ start-dfs.sh --starts NN,SNN,DN --Type Yes if anything asked for**
**$ start-yarn.sh --starts NodeManager,ResourceManager**

**$ start-dfs.sh && start-yarn.sh --In a single line**

**Type yes if asked for**

### 13.Start your history-server. (not required)

Some of the component like pig heavily depends on history server

**$mr-jobhistory-daemon.sh start historyserver**
**$mr-jobhistory-daemon.sh stop historyserver --If you want to stop**

### 14.Check if all the necessary hadoop daemon is running or not

**$ jps**
**4912 NameNode**
**5361 ResourceManager**

REFERENCE :KAUSHIK SHAKKARI
EMAIL : kaushik.shakkari@gmail.com

**5780 Jps**
**5209 SecondaryNameNode**
**5485 NodeManager**
**5251 DataNode**
**3979 JobHistoryServer**

If you see any of the daemon not running, You can visit the log files to figure out the problem. The log files are located at **/usr/local/hadoop/logs**.
E.g; If you don't see data node running, then you should look into **/usr/local/hadoop/logs/hadoop-hduser-datanode-ubuntu.log** and it should help you to debug the problem.

## Check if home folder is created or not in hdfs

**$ hadoop fs -ls**
**16/06/23 13:47:12 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable**
**ls: `.': No such file or directory**

**If You get the above error:** that means Your hadoop home directory was not created successfully. Type the below command

**$ hadoop fs -mkdir -p /user/hduser (Deprecated)**
**$ hdfs dfs -mkdir -p /user/hduser (Use this)**

Now you should not get error with below command. For the first time you will not get any output as the hdfs home folder is empty.

**$ hdfs dfs -ls**

## Check if the hadoop is accessible through browser by hitting the below URLs.

| | |
|---|---|
| **NameNode** | http://*localhost:50070* |
| **ResourceManager** | http://*localhost:8088* |
| **MapReduce JobHistory Server** | http://*localhost:19888* |

19888 is the http port of JobHistoryServer, where as 10020 is the service port which we had configured in step-8
That is all for this tutorial, you may continue with next article in the series "Setup Multi Node Hadoop Cluster on Ubuntu".