

Installation of Hadoop on Ubuntu

Various software and settings are required for Hadoop. This section is mainly developed based on “rsqrl.com” tutorial.

1- Install Java

Software	Java
Version*	Openjdk version “9-internal”
Download link(s)	Use the provided command
File size	-
Install size	-
Requirements	-

* This version is used in this tutorial

Now that we have one Ubuntu node, we can proceed with installation of Hadoop. The first step for installing Hadoop is Java installation.

```
~$ sudo apt-get install openjdk-9-jre
~$ sudo apt-get install openjdk-9-jdk
```

These commands install the specified version of java on your VM. The “sudo” command enables installation as an administrator. When you use the sudo command, the system asks for your password (you created for Ubuntu, Fig 30 in “Preparation of a Cluster Node with Ubuntu” manual). During the installation, the installer identifies the amount of disc space that is required and asks for your permission to continue. Input “y” to continue (Fig 1).

```
Building dependency tree
Reading state information... Done
openjdk-8-jre is already the newest version (8u111-b14-2ubuntu0.16.04.2).
0 upgraded, 0 newly installed, 0 to remove and 51 not upgraded.
daneshva@moe-VM:~$ sudo apt-get install openjdk-9-jre
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libatk-wrapper-java libatk-wrapper-java-jni
Suggested packages:
  icedtea-8-plugin
The following NEW packages will be installed:
  libatk-wrapper-java libatk-wrapper-java-jni openjdk-9-jre
0 upgraded, 3 newly installed, 0 to remove and 51 not upgraded.
Need to get 113 kB of archives.
After this operation, 338 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Fig 1. Installing Java to prepare the system for Hadoop

To check the installation of Java, you can check the version of the installed Java with the following command. You should see the results similar to Fig 2.

```
~$ java -version
```

```
daneshva@MoeVB:~$ java -version
openjdk version "9-internal"
OpenJDK Runtime Environment (build 9-internal+0-2016-04-14-195246.buildd.src)
OpenJDK 64-Bit Server VM (build 9-internal+0-2016-04-14-195246.buildd.src, mixed
mode)
```

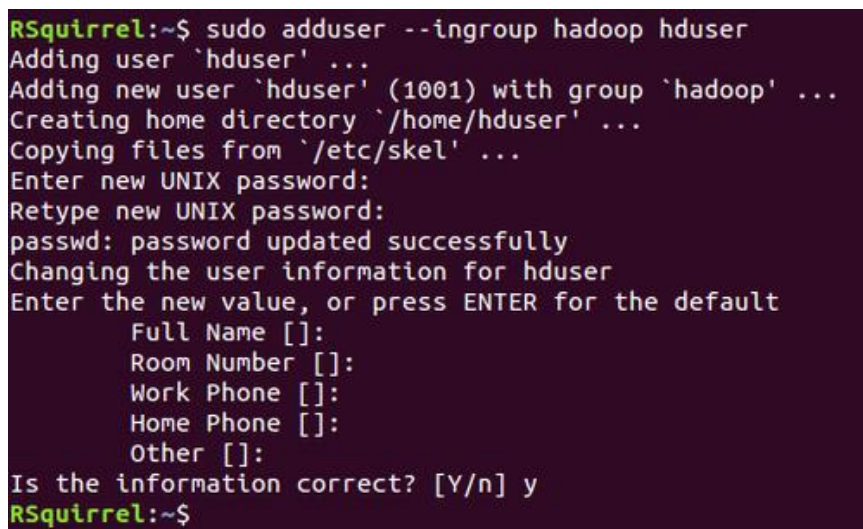
Fig 2. Test the version of your installed Java

2- Define Users and Identify Permission

The most important part in this installation is the management of permissions. Therefore, we create a new user and give read-write permission to her to manage Hadoop related directories. Although this user has enough permission to manage Hadoop, later, we need to use the “root” user, which has enough access, to install packages on R.

To identify user permission, first identify a user group (hadoop). This user group enables to manage several users and assign them to hadoop. Then, add a user to the group (hduser). Again, we use sudo command to run our code with administrator privileges. You can use a different group name and user name based on your preferences. Once the user is identified, the system asks for her password and other information including name, last name, etc. Since this is a trial installation, you can enter a simple password (just for training purposes) and press enter to skip these question (Fig 3).

```
~$ sudo addgroup hadoop
~$ sudo adduser --ingroup hadoop hduser
```



```
RSquirrel:~$ sudo adduser --ingroup hadoop hduser
Adding user `hduser' ...
Adding new user `hduser' (1001) with group `hadoop' ...
Creating home directory `/home/hduser' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for hduser
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
RSquirrel:~$
```

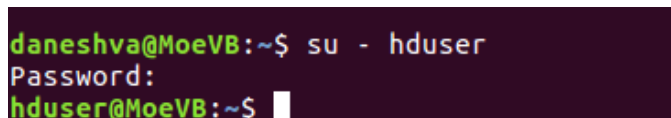
Fig 3. Add a user (hduser) to a user group (hadoop)

It is important to consider that the terminal is case sensitive and considers Hadoop and hadoop to be two different words.

The new user needs to have sudo privileges to be able to run the programs as administrator. Once the sudo privilege is set, we can continue with the new user logged in.

```
~$ sudo usermod -a -G sudo hduser
~$ su - hduser
```

To login with hduser, the system asks for its password which we identified in previous step (Fig 4).



```
daneshva@MoeVB:~$ su - hduser
Password:
hduser@MoeVB:~$
```

Fig 4. Log in with the newly defined user

During the installation process and later, when we use the system, we will need to execute and access files. To create a short hand for these files, go to their directory and run the related command. For example, we need access to the jdk file. Therefore, we use the following command. Later, we create other short hands for starting the Hadoop’s services.

```
~$ cd /usr/lib/jvm
~$ sudo ln -s java-8-openjdk-amd64 jdk
```

Now that the short hand is created, we can go back to our previous directory with the following command.

```
~$ cd ~
```

In this tutorial, you can install Hadoop without knowing the commands in the Terminal. However, if you learn them ([here](#)), it will be much easier for you to comprehend the commands as you go through them.

The next step is installation of ssh server which is needed for management of the communication with localhost. In multiple node Hadoop installation, this server manages the communication of nodes. The following commands download and install the related files for the ssh server.

```
~$ sudo apt-get install openssh-client  
~$ sudo apt-get install openssh-server
```

The first step is configuration of the ssh client by assigning a key to it. Use the following code in Terminal to generate the key. Once asked for address, press enter. The key will be generated and stored.

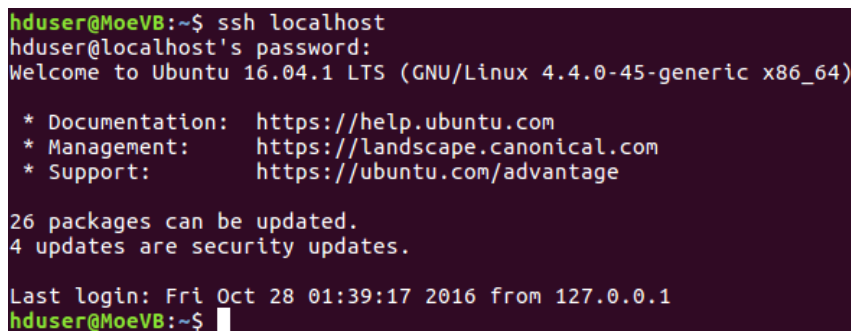
```
~$ ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
```

This key should be authorized so that you can access ssh on your machine.

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

You can test the successful configuration of the ssh on your machine with the following command. If successful, you will see your results similar to Fig 5.

```
~$ ssh localhost
```



```
hduser@MoeVB:~$ ssh localhost  
hduser@localhost's password:  
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-45-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
  
26 packages can be updated.  
4 updates are security updates.  
  
Last login: Fri Oct 28 01:39:17 2016 from 127.0.0.1  
hduser@MoeVB:~$
```

Fig 5. Control the appropriate configuration of the ssh server

3- Install Hadoop

Software	Hadoop
Version*	2.7.1
Download link(s)	Use the provided command in the tutorial link(s)
File size	210 MB
Install size	Variable
Requirements	<ul style="list-style-type: none">• Virtualbox• Ubuntu 10.04 LTS or higher
* This version is used in this tutorial	

The tutorial in this section is developed based on [this](#) and [this](#) sources which I found the most comprehensive and reliable reference.

First, you need to download Hadoop. Navigate to your download folder.

```
~$ cd /  
~$ cd /home/daneshva/Downloads
```

You need to replace daneshva (my username) with your own username. Attention: the username is different from hduser and is the user that you identified in installation of the Ubuntu (Fig 30 in “Preparation of a Cluster Node with Ubuntu” manual).

Now that you are in the folder, download Hadoop and extract it into its installation locatoin (second line of code).

```
~$ wget http://mirror.nohup.it/apache/hadoop/common/hadoop-2.7.1/hadoop-2.7.1.tar.gz  
~$ sudo tar vxzf hadoop-2.7.1.tar.gz -C /usr/local
```

Now, move to the folder of Hadoop and setup the ownership and permissoins.

```
~$ cd /usr/local  
~$ sudo mv hadoop-2.7.1 hadoop  
~$ sudo chown -R hduser:hadoop hadoop
```

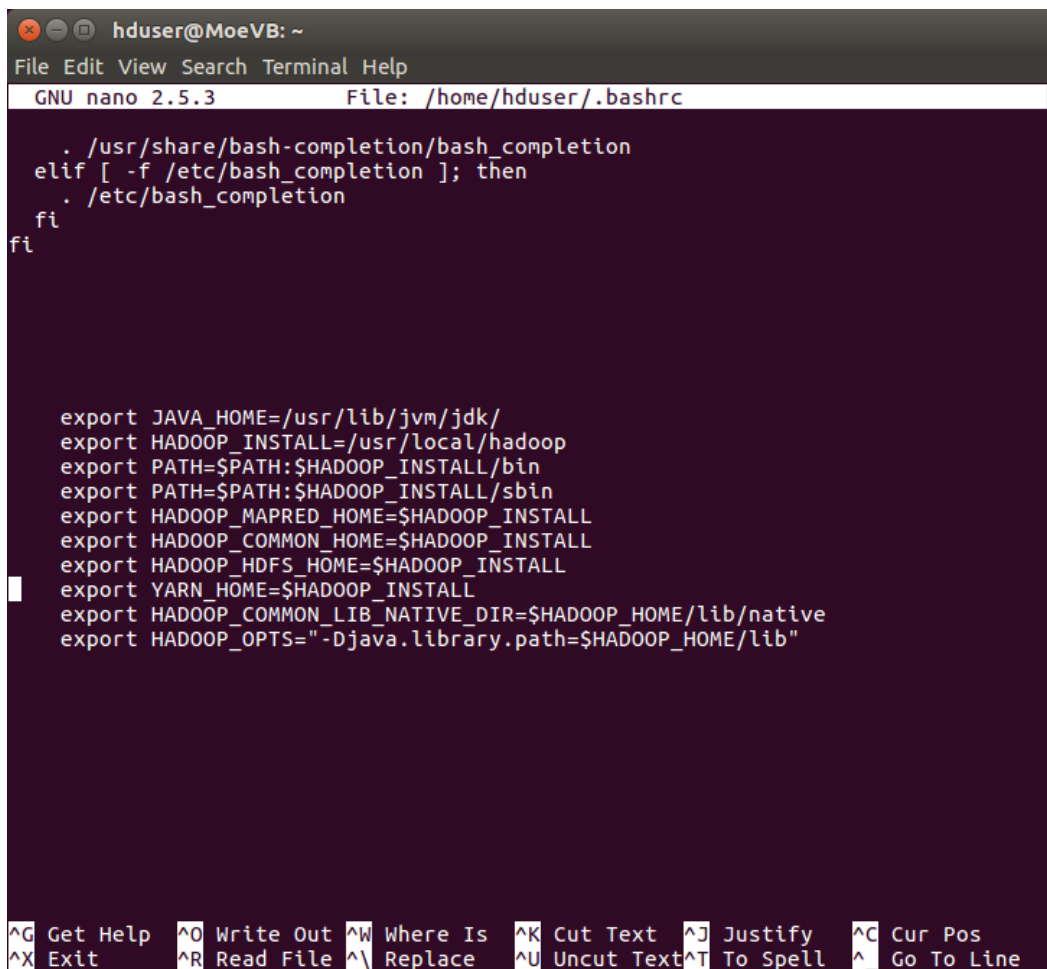
We need to setup parameters in Hadoop so that the program is introduced to important locations that are required for different services. For this purpose, we start with editing `.bashrc`.

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

This command opens a window. Navigate to the end of the window and paste the following lines to it. Then hold CTRL+x to exit. Type “y” and press enter to save the file.

```
export JAVA_HOME=/usr/lib/jvm/jdk/  
export HADOOP_INSTALL=/usr/local/hadoop  
export PATH=$PATH:$HADOOP_INSTALL/bin  
export PATH=$PATH:$HADOOP_INSTALL/sbin  
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL  
export HADOOP_COMMON_HOME=$HADOOP_INSTALL  
export HADOOP_HDFS_HOME=$HADOOP_INSTALL  
export YARN_HOME=$HADOOP_INSTALL  
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native  
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
```

Fig 6 shows the environment that you need to input the above text into it.



```
hduser@MoeVB: ~
File Edit View Search Terminal Help
GNU nano 2.5.3 File: /home/hduser/.bashrc

. /usr/share/bash-completion/bash_completion
elif [ -f /etc/bash_completion ]; then
. /etc/bash_completion
fi
fi

export JAVA_HOME=/usr/lib/jvm/jdk/
export HADOOP_INSTALL=/usr/local/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

Fig 6. Edit .bashrc to set the location of services for Hadoop

The next step is to edit hadoop-env.sh. Similar to .bashrc, open the file and apply the changes. To open the file:

```
~$ sudo nano /usr/local/hadoop/etc/hadoop/hadoop-env.sh
```

Replace the JAVA_HOME export value to the following:

```
~$ export JAVA_HOME=/usr/lib/jvm/jdk/
```

To set the changes in place, reboot the system.

```
~$ systemctl reboot -i
```

Once the system is reset, you should test the Hadoop installation with the following command.

```
~$ hadoop version
```

The results should be similar to Fig 7.

```

hduser@MoeVB:~$ hadoop version
Hadoop 2.7.1
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r 15ecc87ccf4a022
8f35af08fc56de536e6ce657a
Compiled by jenkins on 2015-06-29T06:04Z
Compiled with protoc 2.5.0
From source with checksum fc0a1a23fc1868e4d5ee7fa2b28a58a
This command was run using /usr/local/hadoop/share/hadoop/common/hadoop-common-2
.7.1.jar
hduser@MoeVB:~$

```

Fig 7. Control the version of the installed Hadoop to ensure that installation is flawless

Follow the next steps to configure internal parts of Hadoop. Open each file, paste the configuration settings, and press CTRL+x to exit. To save the settings, type “y” and press enter to save the file.

1- core-site.xml

```
~$ sudo nano /usr/local/hadoop/etc/hadoop/core-site.xml
```

Paste the following between the <configuration>...</configuration> tags.

```

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

```

2- yarn-site.xml

```
~$ sudo nano /usr/local/hadoop/etc/hadoop/yarn-site.xml
```

Paste the following 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>

```

3- mapred-site.xml.template

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

Paste the following between the <configuration>...</configuration> tags.

```

<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>

```

You can save this file without the template extension and copy the file with a new name.

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

4- hdfs-site.xml

To enable interaction with HDFS, we need directories for datanode and namenode. Use the following commands to create these directories.

```

~$ cd ~
~$ mkdir -p mydata/hdfs/namenode
~$ mkdir -p mydata/hdfs/datanode

```

The created directories are used to configure the HDFS.

```
~$ sudo nano /usr/local/hadoop/etc/hadoop/hdfs-site
```

Paste the following between the <configuration>...</configuration> tags.

```
<property>
<name>dfs.replication</name>
<value> 1 </value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:/home/hduser/mydata/hdfs/namenode</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>file:/home/hduser/mydata/hdfs/datanode</value>
</property>
```

The final step is to build the Hadoop file structure with the following command.

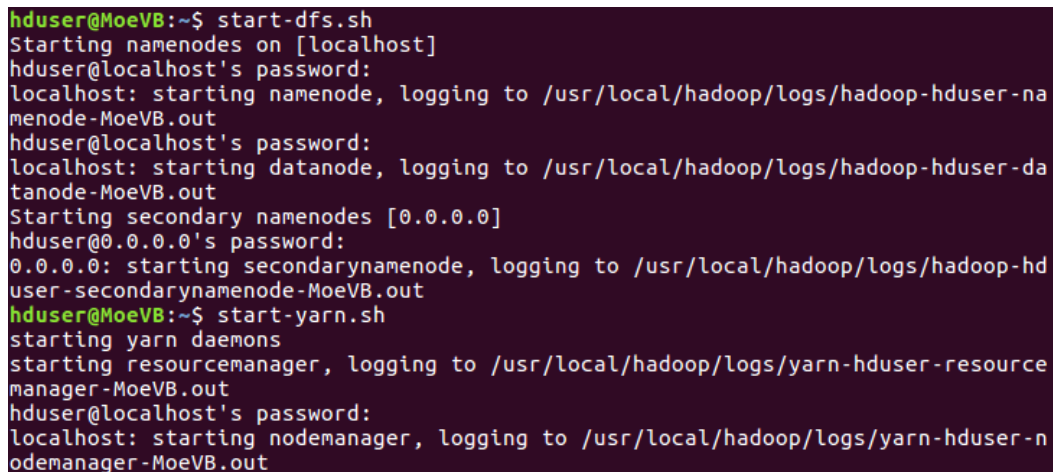
```
~$ hdfs namenode -format
```

Now, the Hadoop is ready and we can run its services by the following commands:

```
~$ Cd /
~$ Cd /usr/local/hadoop/bin
~$ start-dfs.sh
~$ start-yarn.sh
```

Alternatively, use the following command to run all the services. Once started, you should see Fig 8.

```
~$ Cd /
~$ Cd /usr/local/hadoop/bin/start-all.sh
```



```
hduser@MoeVB:~$ start-dfs.sh
Starting namenodes on [localhost]
hduser@localhost's password:
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hduser-na
menode-MoeVB.out
hduser@localhost's password:
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hduser-da
tanode-MoeVB.out
Starting secondary namenodes [0.0.0.0]
hduser@0.0.0.0's password:
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-hd
user-secondarynamenode-MoeVB.out
hduser@MoeVB:~$ start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-hduser-resource
manager-MoeVB.out
hduser@localhost's password:
localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-hduser-n
odemanager-MoeVB.out
```

Fig 8. Starting dfs and yarn services

To ensure that the services are working appropriately, you can use the following command.

```
~$ jps
```

If all the services are working accurately, you should be able to see the outcome shown in Fig 9.

```

hduser@MoeVB:~$ jps
5347 NodeManager
5221 ResourceManager
5558 Jps
4730 NameNode
5066 SecondaryNameNode
4878 DataNode
hduser@MoeVB:~$

```

Fig 9. Run jps command to see the services that are running

If you replace start with stop in the above commands, you can stop Hadoop services.

```

~$ stop-dfs.sh
~$ stop-yarn.sh

```

The final check for the flawless operation of Hadoop on your system is checking the Hadoop web service on this address: <http://localhost:50070/> (the default address for web user interface of the NameNode daemon). The user interface is shown in Fig 10.

The screenshot shows the Hadoop web user interface for the NameNode. The browser address bar shows `localhost:50070/dfshealth.html#tab-overview`. The page title is "Overview 'localhost:9000' (active)".

Started:	Thu Nov 03 21:58:02 EDT 2016
Version:	2.7.1, r15ecc87ccf4a0228f35af08fc56de536e6ce657a
Compiled:	2015-06-29T06:04Z by jenkins from (detached from 15ecc87)
Cluster ID:	CID-13800718-82d1-4270-a9bc-7f1c18c63ec8
Block Pool ID:	BP-404180789-127.0.1.1-1477634695833

Summary

Security is off.
Safemode is off.
94 files and directories, 45 blocks = 139 total filesystem object(s).
Heap Memory used 36.58 MB of 60.06 MB Heap Memory. Max Heap Memory is 966.69 MB.
Non Heap Memory used 39.38 MB of 40.06 MB Committed Non Heap Memory. Max Non Heap Memory is -1 B.

Configured Capacity:	15.62 GB
DFS Used:	9.09 MB (0.06%)
Non DFS Used:	10.16 GB
DFS Remaining:	5.45 GB (34.89%)
Block Pool Used:	9.09 MB (0.06%)
DataNodes usages% (Min/Median/Max/stdDev):	0.06% / 0.06% / 0.06% / 0.00%
Live Nodes	1 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)

Fig 10. Web user interface of NameNodes. Make sure that you have 1 “Live Nodes” and 0 “Dead Nodes”