Single Node Hadoop 3.0.3 setup on Linux

1 Prerequisites

Hadoop is a big data storage and analytics framework that is built on top of the Java TM platform and runs on the Java Virtual Machine. Thus installation of JDK is a must. Also we need to setup passphraseless SSH server in order to access the cluster.

1.1 Installing JDK 8

1. Open terminal and type the following command to add Oracle's PPA,

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
```

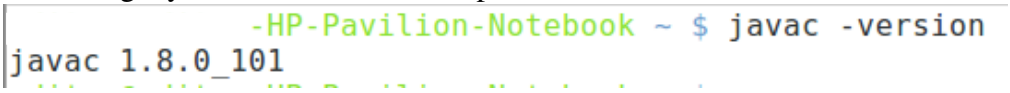
2. Now go on and install JDK 8 on your Machine

```
$ sudo apt-get install oracle-java8-installer
```

3. Check if the installation is working type the following command

```
$ javac -version
```

You will get your JDK version as output.



1.2 Setting JAVA_HOME environment Variable and Adding Java to your PATH

1. Open terminal and go to home directory.
   `$ cd ~`

2. Now open the .bashrc file in your preferred text editor (I am using gedit)
   `$ gedit .bashrc`

3. Add the following lines to the bottom of your .bashrc file.
```
export PATH=$PATH:/opt/java/jdk1.8.0_101/bin
export JAVA_HOME=/opt/java/jdk1.8.0_101
```

4. Compile your .bashrc to make your changes permanent.
```
$ source .bashrc
```

1.3 Setting up passphraseless SSH

1. If you have not installed SSH software you will need to install it.
```
$ sudo apt-get install ssh
$ sudo apt-get install pdsh
```

2. Now in order to ssh to localhost without a passphrase (Empty password), execute the following commands:
```
$ ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
$ chmod 0600 ~/.ssh/authorized_keys
```

2 Downloading and Installing Hadoop 3.0.3

2.1 Downloading Hadoop

Type the following command to download Hadoop 3.0.3 binary tar file to your machine. It will be downloaded to your home directory.

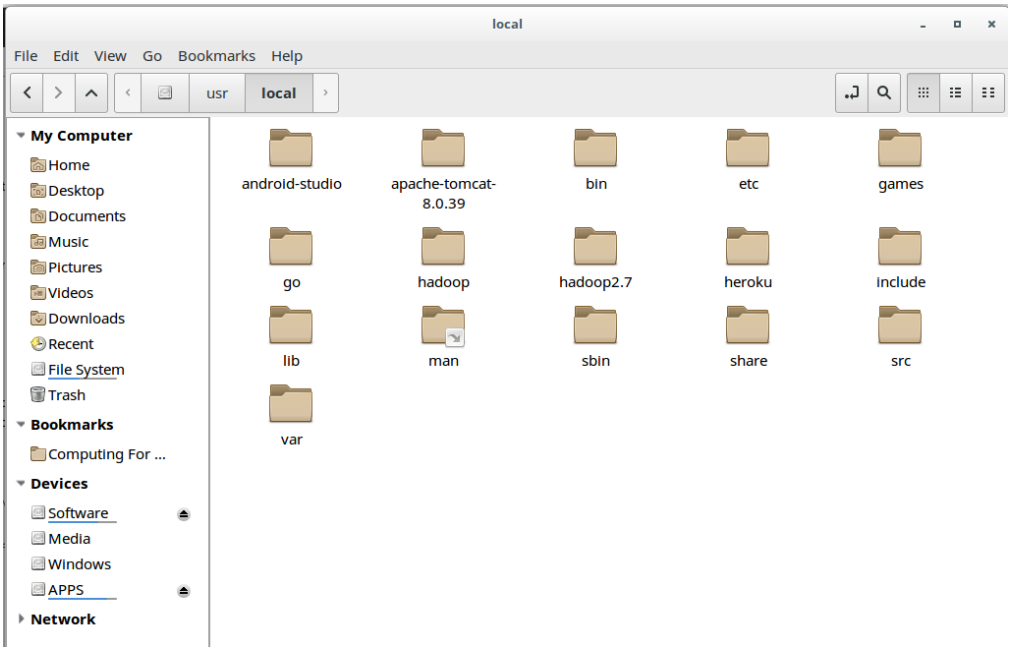
```
$ wget http://www-eu.apache.org/dist/hadoop/common/hadoop-3.0.3/hadoop-3.0.3.tar.g
```

Or to get the latest version go to `http://hadoop.apache.org/releases.html`

2.2 Installing Hadoop

1. Unpack the downloaded tar file to /usr/local
```
$ tar -xvf articles.tar -C /usr/local/
```

2. A new directory named Hadoop will appear in /usr/local/ directory. If it's Hadoop3.0.3 rename it to Hadoop.



3. Now it's time to set the Hadoop specific environment variable. open .bashrc for editing.
   `$ cd ~ $ gedit .bashrc`

4. Add the following lines to your .bashrc

```
# For Hadoop User specific aliases and functions
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_CONF_DIR=/usr/local/hadoop/etc/hadoop
export HADOOP_MAPRED_HOME=/usr/local/hadoop
export HADOOP_COMMON_HOME=/usr/local/hadoop
export HADOOP_HDFS_HOME=/usr/local/hadoop
export YARN_HOME=/usr/local/hadoop
export PATH=$PATH:/usr/local/hadoop
export PATH=$PATH:/usr/local/hadoop/bin
export PATH=$PATH:/usr/local/hadoop/sbin
```

5. Compile your .bashrc to make your changes permanent.
   `$ source .bashrc`

6. To check if it worked out type the following command.
   `$ hadoop version`

7. Your output will be something like.

```
HP-Pavilion-Notebook ~ $ hadoop version
Hadoop 3.0.3
Source code repository https://yjzhangal@git-wip-us.apache.org/repos/asf/hadoop.
git -r 37fd7d752db73d984dc31e0cdfd590d252f5e075
Compiled by yzhang on 2018-05-31T17:12Z
Compiled with protoc 2.5.0
From source with checksum 736cdcefa911261ad56d2d120bf1fa
This command was run using /usr/local/hadoop/share/hadoop/common/hadoop-common-3
.0.3.jar
```

3 HDFS and Yarn Configuration

We have all the Hadoop binaries on our system but that's not it yet. In order to get Hadoop up and running we have to specify certain properties and configurations. Which is easily done with a little bit of XML hacking.

Change to hadoop configuration directory

```
$ cd /usr/local/hadoop/etc/hadoop
```

3.1 Edit hadoop-env.sh

1. Open hadoop-env.sh for editing.
   ```
   $ gedit hadoop-env.sh
   ```
2. Add the following line to point Hadoop installation towards your JDK (Version may change).
   ```
   export JAVA_HOME=/opt/java/jdk1.8.0_101
   ```

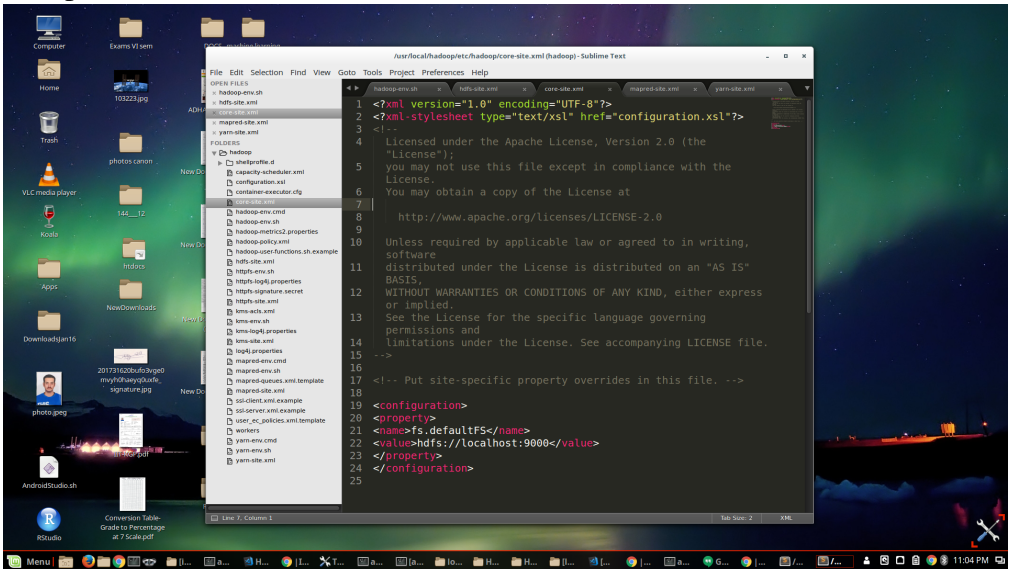
3.2 Edit core-site.xml

This file informs Hadoop daemon where NameNode runs in the cluster.

1. Open core-site.xml for editing.
   ```
   $ gedit core-site.xml
   ```
2. Add the following properties in between the <configuration> and </configuration> tags.

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

3. Complete file will look like below:



3.3 Edit hdfs-site.xml

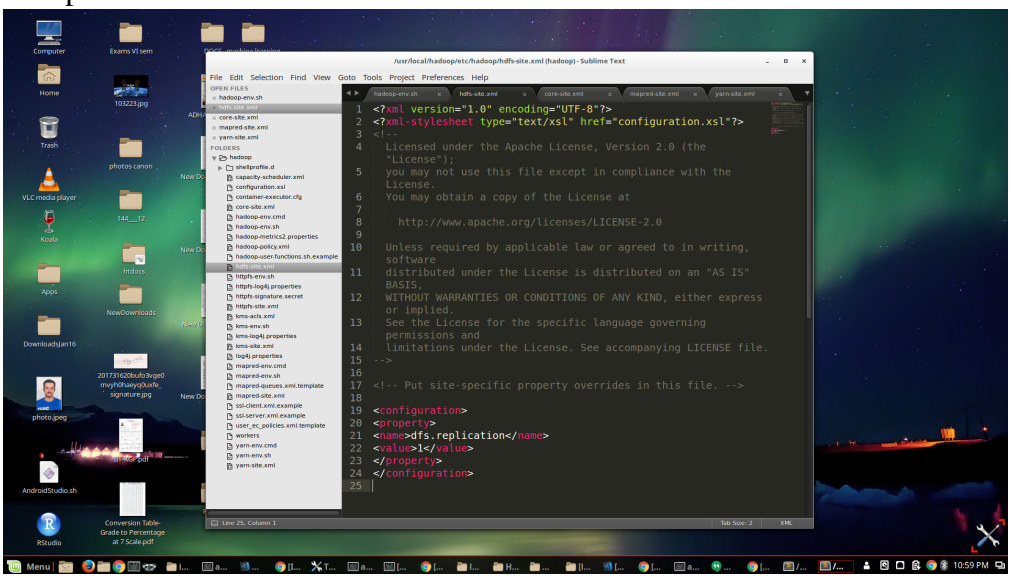
This file contains configuration settings of various HDFS daemons (i.e. NameNode, DataNode, Secondary NameNode). It also includes the replication factor and block size of HDFS.

1. Open hdfs-site.xml for editing.
   ```
   $ gedit hdfs-site.xml
   ```

2. Add the following properties in between the <configuration> and </configuration> tags.

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

3. Complete file will look like below:



3.4 Edit mapred-site.xml

This file contains configuration settings of MapReduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process, etc.
If mapred-site.xml file is not available. So, we have to create the mapred-site.xml fileusing mapred-site.xml template.

1. If mapred-site.xml is not avaliable create it from mapred-site.xmltemplate.
   ```
   $ cp mapred-site.xml.template mapred-site.xml
   ```

2. Open mapred-site.xml for editing.
   ```
   $ gedit mapred-site.xml
   ```

3. Add the following properties in between the <configuration> and </configuration> tags.

```
<configuration>
   <property>
      <name>mapreduce.framework.name</name>
      <value>yarn</value>
   </property>
   <property>
      <name>mapreduce.application.classpath</name>
      <value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME
   </property>
</configuration>
```

4. Complete file will look like below:

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
    <property>
        <name>mapreduce.application.classpath</name>
        <value>$HADOOP_MAPRED_HOME/share/hadoop/
        mapreduce/*:$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/
        lib/*</value>
    </property>
</configuration>
```

3.5 Edit yarn-site.xml

This file contains configuration settings of ResourceManager and NodeManager like application memory management size ,the operation needed on program and algorithm,

1. Open yarn-site.xml for editing.
   ```
   $ gedit yarn-site.xml
   ```

2. Add the following properties in between the <configuration> and </configuration> tags.

```
<configuration>
   <property>
      <name>yarn.nodemanager.aux-services</name>
      <value>mapreduce_shuffle</value>
   </property>
   <property>
      <name>yarn.nodemanager.env-whitelist</name>
      <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CL
   </property>
</configuration>
```

3. Complete file will look like below:

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <name>yarn.nodemanager.env-whitelist</name>
        <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HA
        DOOP_CONF_DIR,CLASSPATH_PREPEND_DISTCACHE,HADOOP_YARN_H
        OME,HADOOP_MAPRED_HOME</value>
    </property>
</configuration>
```

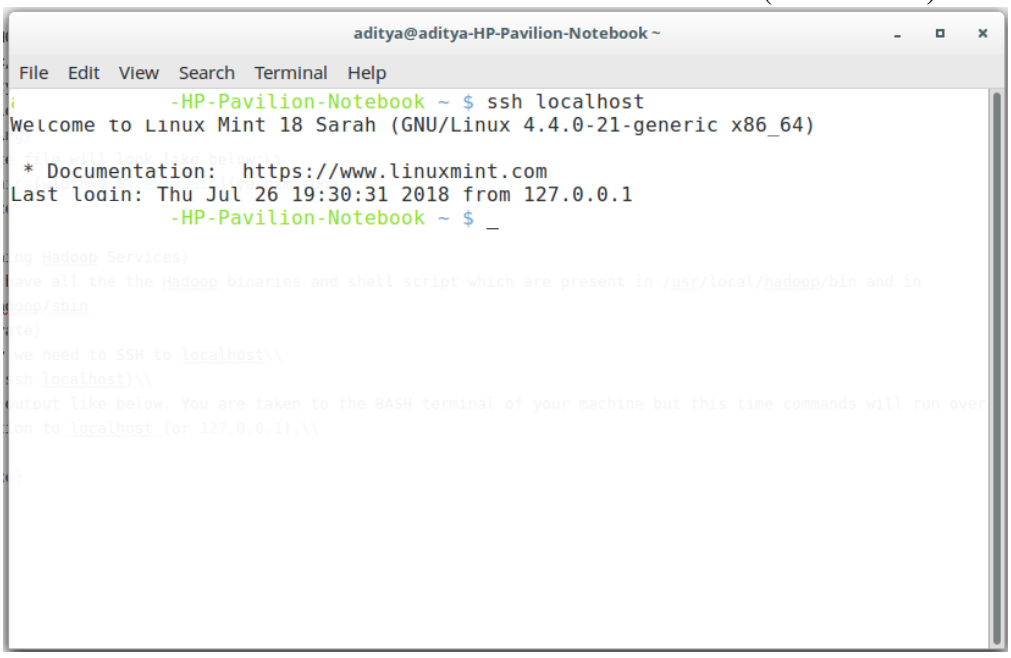
4 Running Hadoop Services

Now since we have added all the the Hadoop binaries and shell script which are present in */usr/local/hadoop/bin and in /usr/local/hadoop/sbin* to our PATH. We can directly run those binaries and scripts from terminal by just typing in their names.

1. Firstly we need to SSH to localhost

   `$ ssh localhost`

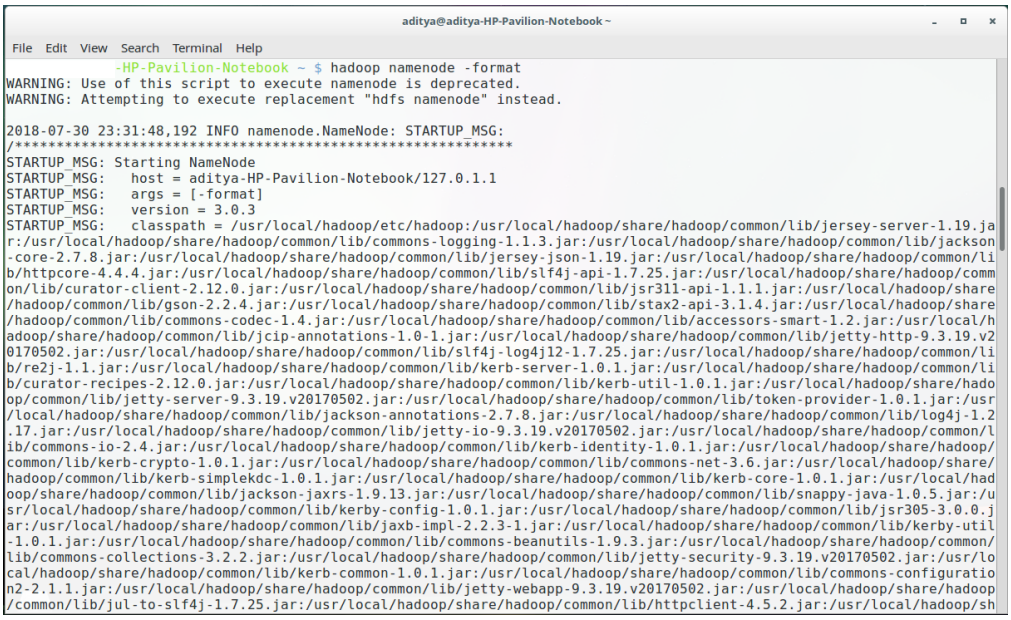
   You will see output like below. You are taken to the BASH terminal of your machine but this time commands will run via a SSH connection to localhost (or 127.0.0.1).



2. Before running our cluster for the first time we need to format our namenode

   `$ hadoop namenode -format`

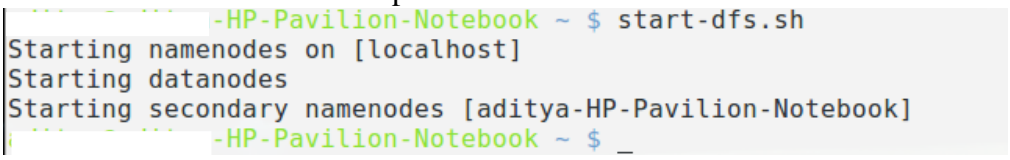
   You will receive the below output:



3. Now to start the HDFS file system with its Namenodes and Datanodes enter the following command.

   `$ start-dfs.sh`

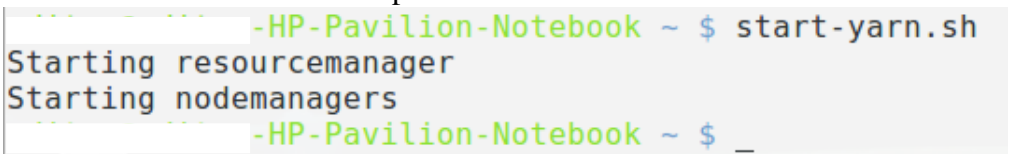
   You will receive the below output:



4. Now to start the Yarn resource manager type.

   `$ start-yarn.sh`

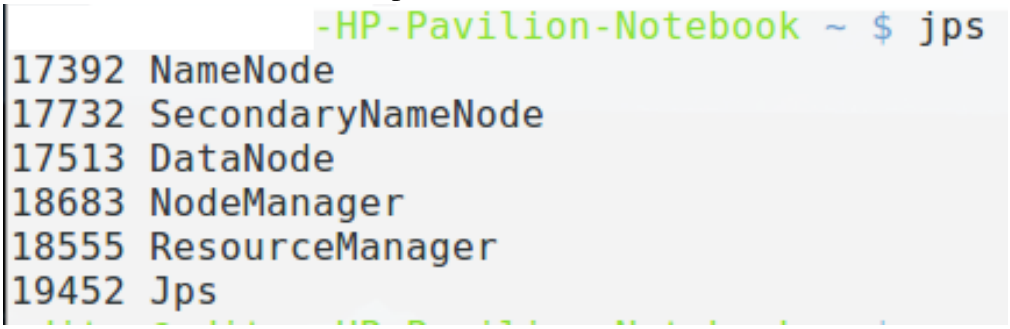
   You will receive the below output:



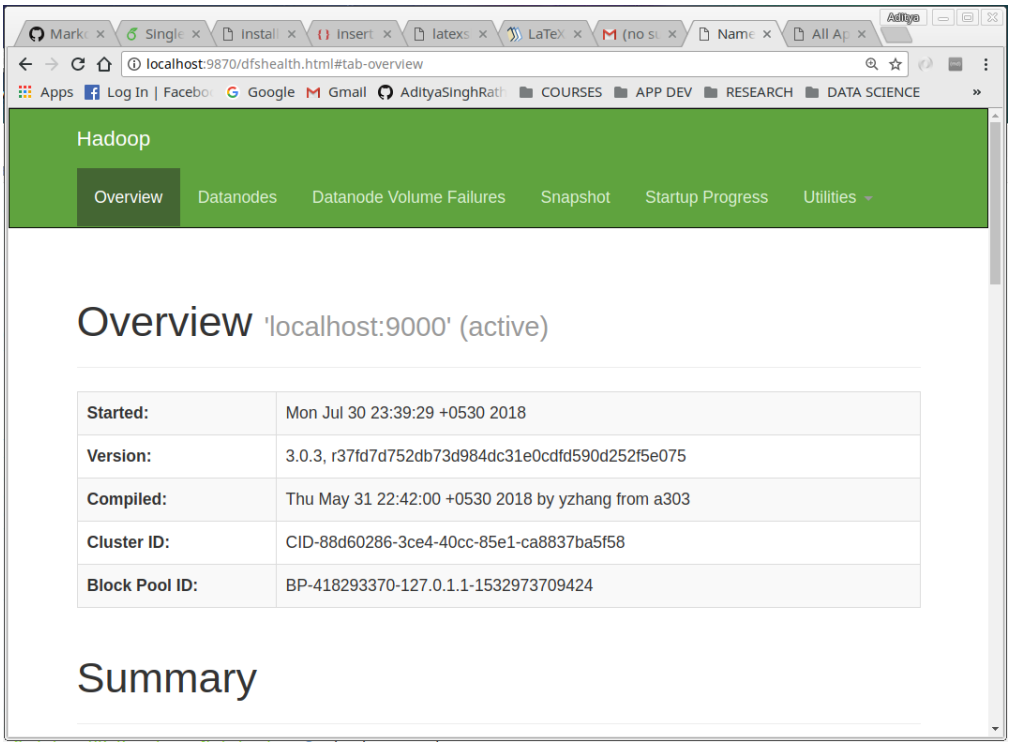
5. Using *jps* command we can see all the running services

   `$ jps`

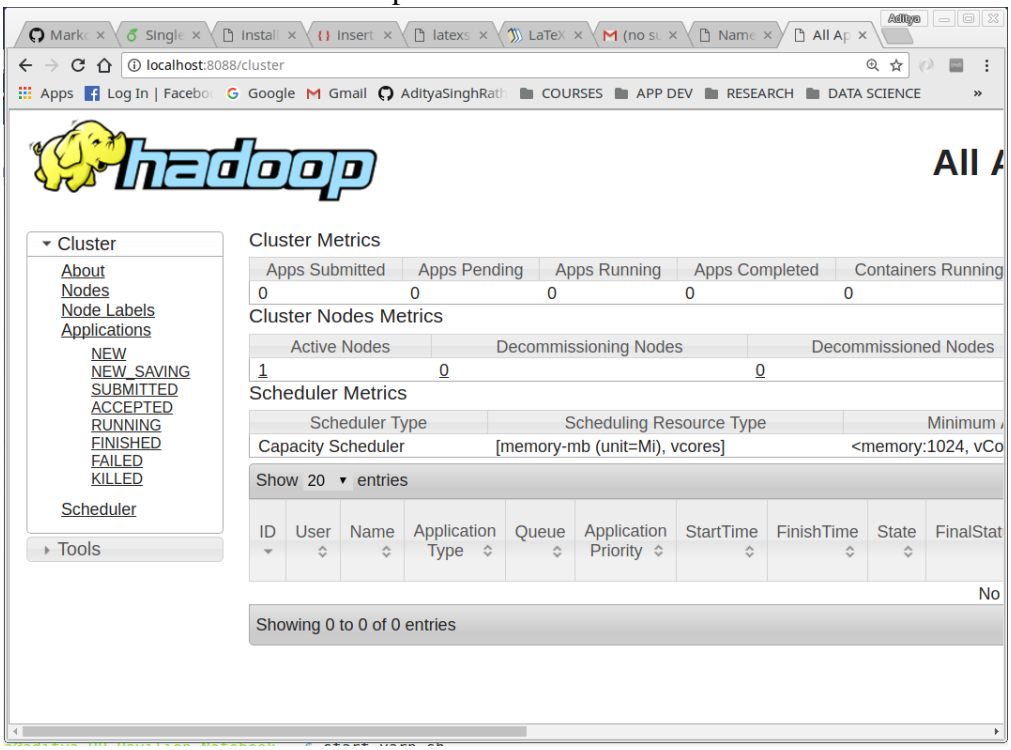
   You will receive the below output:



6. Open *http://localhost:9870* in browser to see the Namenode interface.
   You will receive the below output:

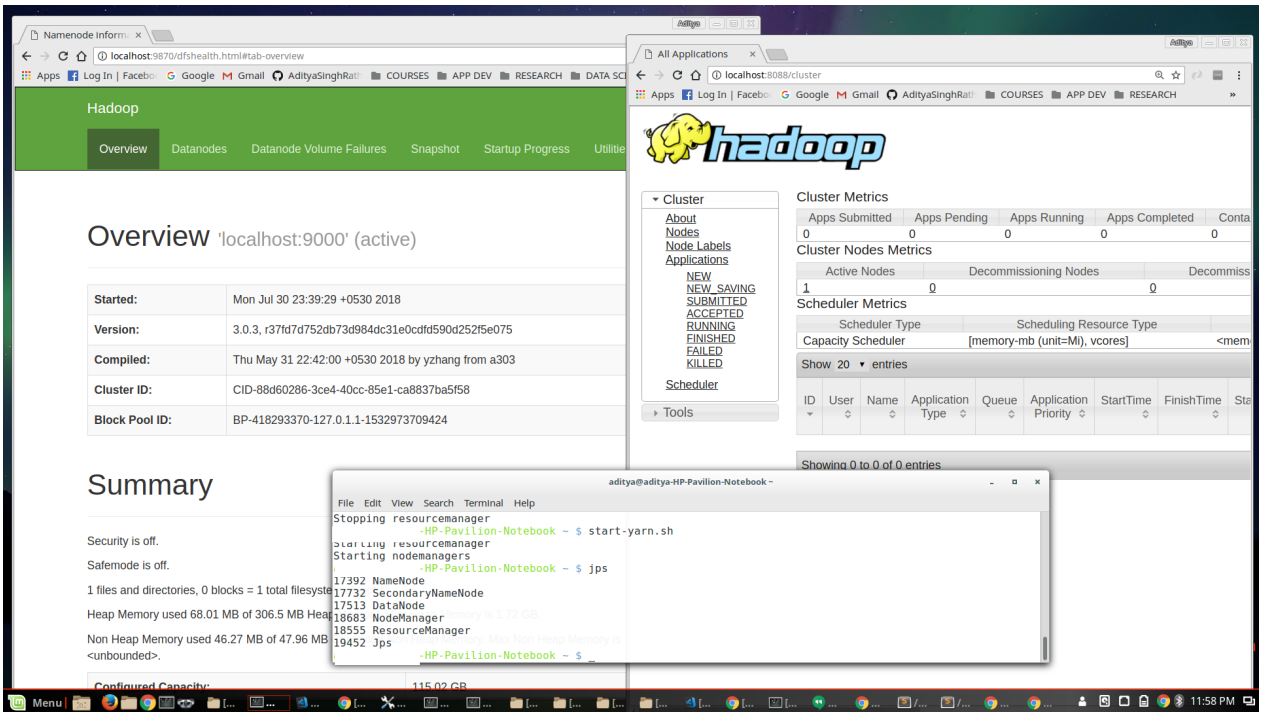


7. Open *http://localhost:8088* in browser to see the Cluster interface.
   You will receive the below output:



5 Conclusion and Stopping

Congratulations !, Your Linux Mint 18 Machine has a fully functional single Node Hadoop 3.0.3 cluster up and running.



Next up we will run a simple MapReduce Wordcount example on our Cluster. For now use the following commands to stop the cluster.

1. To start the HDFS file system enter the following command.

   ```
   $ stop-dfs.sh
   ```

   You will receive the below output:

   ```
   -HP-Pavilion-Notebook ~ $ stop-dfs.sh
   Stopping namenodes on [localhost]
   Stopping datanodes
   Stopping secondary namenodes [ -HP-Pavilion-Notebook]
   -HP-Pavilion-Notebook ~ $ _
   ```

2. To stop the Yarn resource manager type.

   ```
   $ stop-yarn.sh
   ```

   You will receive the below output:

   ```
   -HP-Pavilion-Notebook ~ $ stop-yarn.sh
   Stopping nodemanagers
   Stopping resourcemanager
   -HP-Pavilion-Notebook ~ $ _
   ```