HADOOP

1.apt-get install openjdk-8-jdk

2.javac -version

3.update-alternatives --config java

4.gedit /etc/environment

JAVA_HOME="/usr/lib/jvm/java-8-openjdk-i386"(copy this location from command 3)

Save the file

5.source /etc/environment

6.echo $JAVA_HOME

7.apt-get install ssh

8. ssh-keygen -t rsa -P ""

9.cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

10.ssh localhost

11.Open browser and go to "hadoop.apache.org/releases.html" and open the 2.7.2 binary file and click the download link

wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.2/hadoop-2.7.2.tar.gz

12.tar –xvzf ("open downloads and drag and drop the downloaded hadoop-2.7.2.tar.gz folder")

Eg: tar –xvzf '/home/sheeha/Downloads/hadoop-2.7.2.tar.gz'

13.sudo gedit ~/.bashrc

```
#HADOOP VARIABLES START
export JAVA_HOME= <YOUR JAVA HOME PATH>
export HADOOP_INSTALL= <HADOOP HOME PATH>
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=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
#HADOOP VARIABLES END
```

14.source ~/.bashrc

15.cd hadoop-2.7.2

16.cd etc

17.cd hadoop

18.gedit hadoop-env.sh

```
Append the following line to hadoop-env.sh:
export JAVA_HOME=<YOUR JAVA HOME PATH>
```

cd $home

19.mkdir hadoop_store

20.cd hadoop_store

21.mkdir hdfs

22.cd hdfs

23.mkdir namenode

24.mkdir datanode

25.cd

26.cd hadoop-2.7.2/etc/hadoop

27.gedit hdfs-site.xml

```
<property>
  <name>dfs.replication</name>
  <value>1</value>
  <description>Default block replication.
  The actual number of replications can be specified when the file is created.
  The default is used if replication is not specified in create time.
  </description>
 </property>
 <property>
   <name>dfs.namenode.name.dir</name>
  <value>file:######NAMENODE_FOLDER_PATH######</value>
 </property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:######DATANODE_FOLDER_PATH######</value>
 </property>
```

28.cd

29.mkdir hadoop-2.7.2/etc/hadoop

30.gedit core-site.xml

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

 <property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:54310</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
  the FileSystem implementation class.  The uri's authority is used to
  determine the host, port, etc. for a filesystem.</description>
 </property>
```

31.cp mapred-site.xml.template mapred-site.xml

32.gedit mapred-site.xml

```
<property>
  <name>mapred.job.tracker</name>
  <value>localhost:54311</value>
  <description>The host and port that the MapReduce job tracker runs
  at.  If "local", then jobs are run in-process as a single map
  and reduce task.
  </description>
 </property>
```

33.cd

34.hadoop namenode -format

35.cd hadoop-2.7.2/sbin

36.start-all.sh

37.Open browser and goto localhost:8088 and check whether its running

38.Open browser and goto localhost:50070 and check

EX. NO: 5

DATE: **MOUNT THE ONE NODE HADOOP CLUSTER USING FUSE**

AIM

To mount one-node Hadoop cluster using FUSE

THEORY

FUSE (Filesystem in Userspace) enables you to write a normal user application as a bridge for a traditional filesystem interface. The hadoop-hdfs-fuse package enables you to use your HDFS cluster as if it were a traditional filesystem on Linux. It is assumed that you have a working HDFS cluster and know the hostname and port that your NameNode exposes.

PROCEDURE

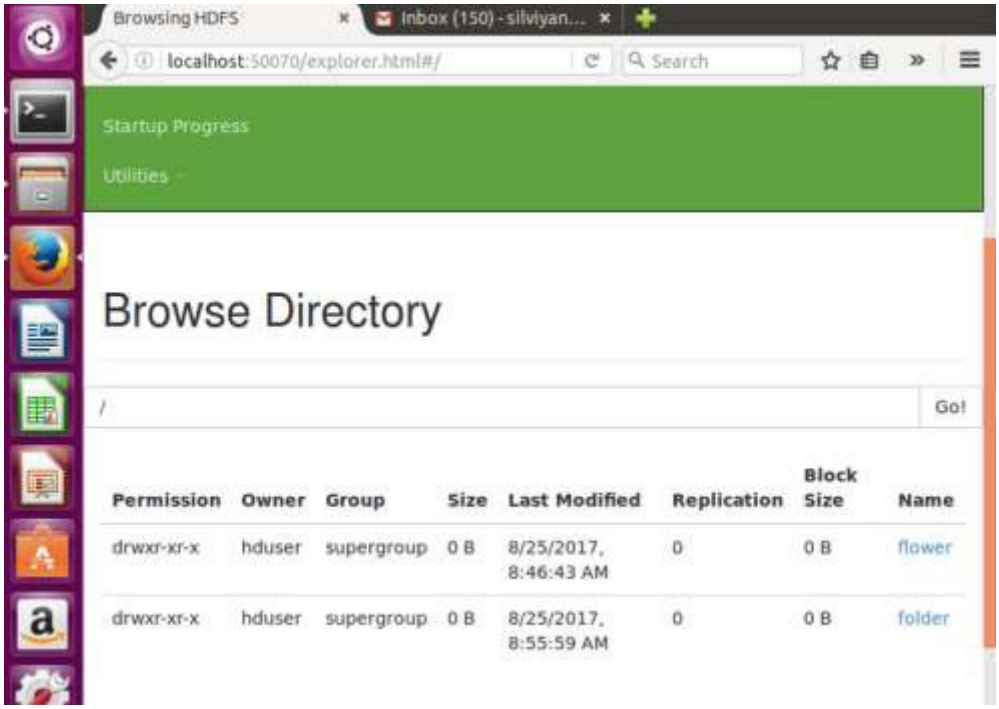
$ su hduser

$ cd /usr/local/hadoop/hadoop-2.7.2/sbin

$ start-dfs.sh

$ start-yarn.sh

$ jps



$ wget http://archive.cloudera.com/cdh5/one-click-install/trusty/amd64/cdh5-repository_1.0_all.deb

$ sudo dpkg -i cdh5-repository_1.0_all.deb

$ sudo apt-get update

$ sudo apt-get install hadoop-hdfs-fuse

$ sudo mkdir -p <mount_point>

Sample :sudo mkdir -p file

```
sudo -H gedit /etc/samba/smb.conf
```

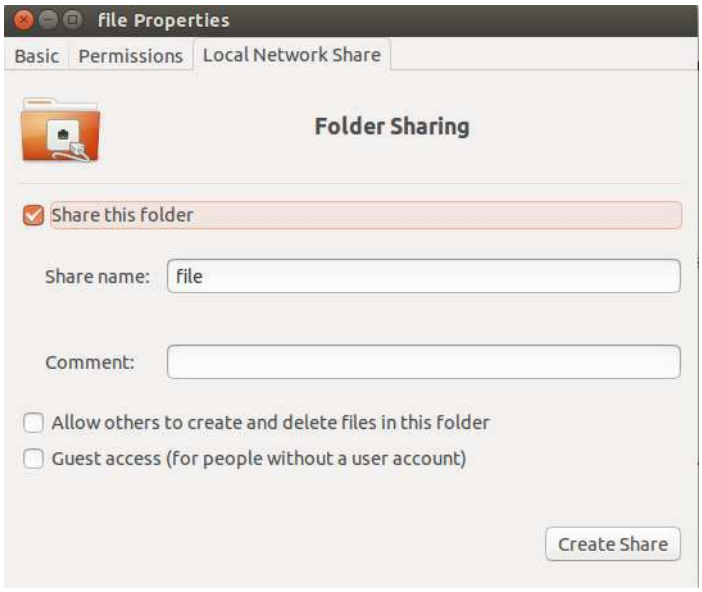
usershare owner only = false

sudo restart smbd

restart the pc

```
hduser@ubuntu:~$ sudo mkdir -p file
hduser@ubuntu:~$ ls
ls: cannot access hello: No such file or directory
cdh5-repository_1.0_all.deb  Downloads         hadoop-2.7.2.tar.gz  Public
cse                          examples.desktop  hello                Templates
Desktop                      file              Music                Videos
Documents                    folder            Pictures
```

(set folder permission and enable share local network)



(for name_node_hostname>:<namenode_port>

goto (cd /usr/local/hadoop/etc/hadoop/ vi core-site.xml)


hduser@ubuntu:~$ cd /usr/local/hadoop/hadoop-2.7.2/etc/hadoop/
hduser@ubuntu:/usr/local/hadoop/hadoop-2.7.2/etc/hadoop$ sudo nano core-site.xml
[sudo] password for hduser:

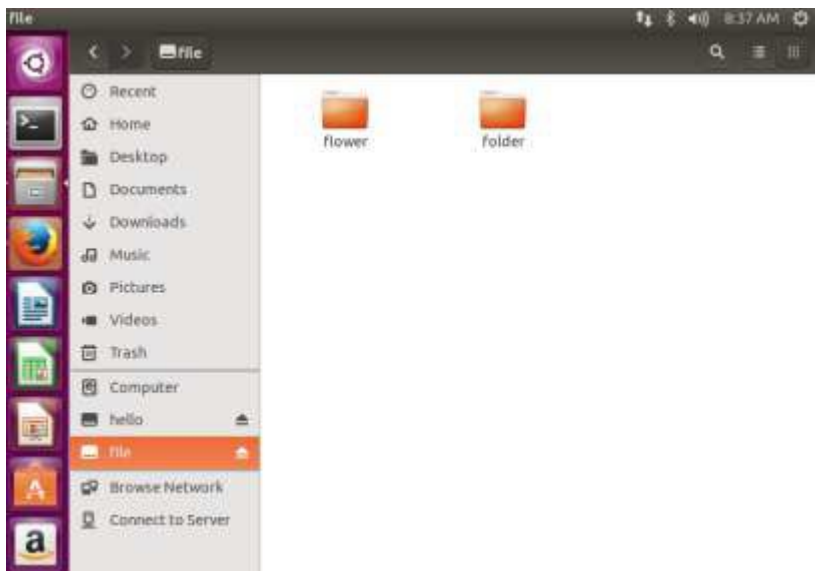
```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
</configuration>
```

$ sudo hadoop-fuse-dfs dfs://<name_node_hostname>:<namenode_port>

<mount_point>

Sample : sudo hadoop-fuse-dfs dfs://localhost:9000 file -o nonempty

```
hduser@ubuntu:~$ sudo hadoop-fuse-dfs dfs://localhost:9000 file
INFO /data/jenkins/workspace/generic-package-ubuntu64-14-04/CDH5.12.0-Packaging-
Hadoop-2017-06-29_04-14-05/hadoop-2.6.0+cdh5.12.0+2512-1.cdh5.12.0.p0.38~trusty/
hadoop-hdfs-project/hadoop-hdfs/src/main/native/fuse-dfs/fuse_options.c:164 Addi
ng FUSE arg file
```



Once HDFS has been mounted at <mount_point>, you can use most of the traditional filesystem operations (e.g., cp, rm, cat, mv, mkdir, rmdir, more, scp). However, random write operations such as rsync, and permission related operations such as chmod, chown are not supported in FUSE-mounted HDFS.

RESULT

Thus the steps to mount one-node Hadoop cluster using FUSE is done successfully.

MAPREDUCE

1.hadoop version

2.javac –version

3. Create a file called WordCount.java in desktop and copy this program and save

```
 import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

  public static class TokenizerMapper
       extends Mapper<Object, Text, Text, IntWritable>{

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
  }

  public static class IntSumReducer
       extends Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,
                       Context context
                       ) throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }

  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

4.Create a new folder in the desktop as “input”

5.Open “input” folder and create a text document called input.txt and type few words in the text file and save

6.Create a folder in the desktop as “tutorial”

7.Go to the terminal and type

   export HADOOP_CLASSPATH=$(hadoop classpath)

8.echo $HADOOP_CLASSPATH

9.hadoop fs –mkdir /WordCountTutorial

10.hadoop fs –mkdir /WordCountTutorial/Input

11.Open browser and go to localhost:50070 and Utilities->Browse the file system and check for WordCountTutorial

12.hadoop fs –put ‘drag and drop input.txt’ /WordCountTutorial/Input

13. Open browser and go to localhost:50070 and Utilities->Browse the file system and open WordCountTutorial,open Input and check whether input.txt is present

14.cd /home/sheeha/Desktop/WordCountTutorial

15.javac –classpath ${HADOOP_CLASSPATH} -d ‘class folder ‘drag and drop wordcount.java file’

16.In files open tutorial folder and check whether three files are present

17.jar –cvf firstTutorial.jar -C class(class folder)/ .

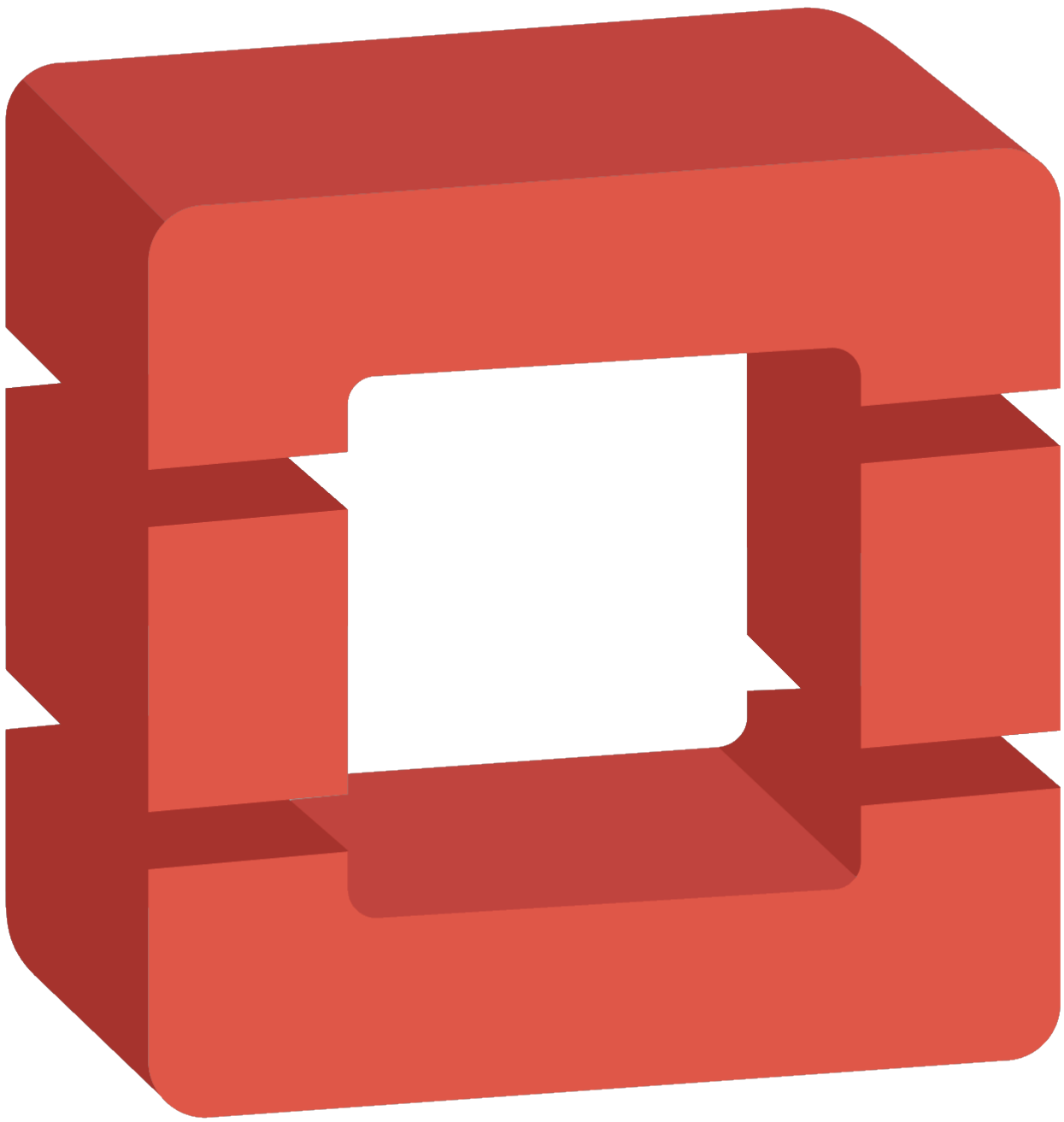
18.Open files and check whether a jar file is present

19.hadoop jar ‘drag and drop firstTutorial.jar file’ WordCount(class name) /WordCountTutorial/Input /WordCountTutorial/Output

20.hadoop dfs –cat /WordCountTutorial/Output/*

Installing

OpenStack



This document contains the steps highlighting the procedures for AIO-OpenStack installation using Devstack installer on Ubuntu 16.04 . Developers have tested Openstack for Ubuntu 16.0.4. There is no official information available regarding support for Ubuntu 18.0.4, which is the latest version at the time of writing this. A successful installation was performed in our lab on computers with the followingconfiguration

- Core i5 (6thgen)
- 8 GBRam
- Windows 10
- Access tointernet
- Virtual Machine with Ubuntu 16.04.5, 4.5 GB RAM and 20 GB storage.

1. After successful installation of VM install git .

   sudo apt-get installgit

2. Create a non-root user which is sudo enabled

   I) **sudo useradd -s /bin/bash -d /opt/stack -m stack**

   II) **echo"stack ALL=(ALL) NOPASSWD:ALL"|sudotee/etc/sudoers.d/stack**

   III) **sudosu-stack**

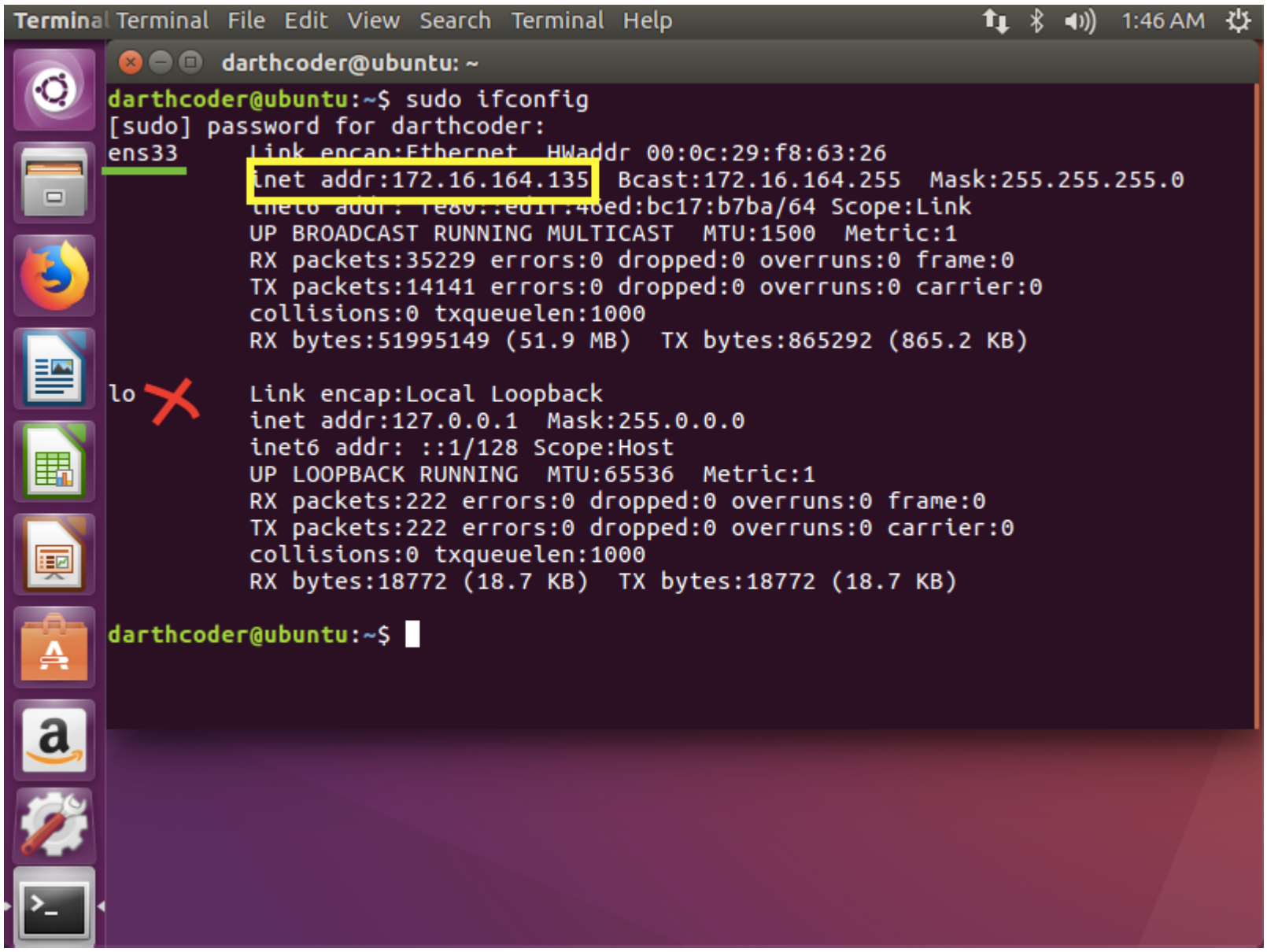
3. Clone the devstack repository

   I) **git clone https://git.openstack.org/openstack-dev/devstack**

   II) **cd devstack**

4. Identify the Host IP. Copy the “inet address” from any device except lo and vibr0

   sudo ifconfig



5. Create a local.conf file inside the devstack directory and containing the following details.

sudo gedit local.conf [[local|localrc]]

ADMIN_PASSWORD=secret

DATABASE_PASSWORD=$ADMIN_PASSWORD

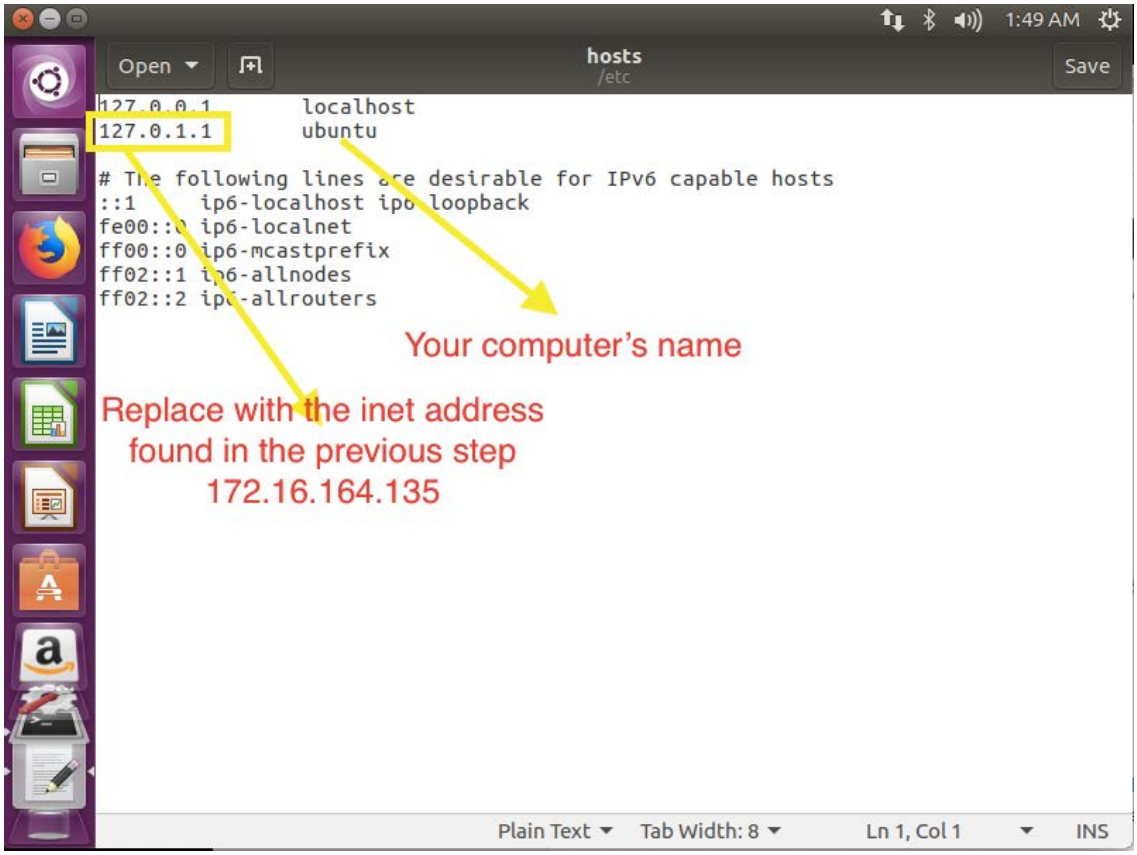
RABBIT_PASSWORD=$ADMIN_PASSWORD

SERVICE_PASSWORD=$ADMIN_PASSWORD

HOST_IP= “the inet address after running sudo ifconfig”
FLOATING_RANGE=

6. Edit the hosts file. Replace your 127.0.0.1 with the host ip.

sudo gedit /etc/hosts/



1.

7. Start the installation.

./stack.sh

The installation would typically take between 30 and 50 minutes. During installation you may be prompted with entering the passwords for different Openstack services. Post installation, visit your host ip in the web browser for the dashboard.

Important commands

Ps aux | grep -I apt

Sudo kill -9

sudo chown -R rec /dest

for lock error

sudo rm /var/lib/dpkg/lock

sudo rm /var/lib/apt/lists/lock

sudo rm /var/cache/apt/archives/lock

sudo rm -rf /var/lib/apt/lists/*

//openstack

virtualenv /opt/stack/requirements/.venv/

sudo chown user(stack): /opt/stack directory