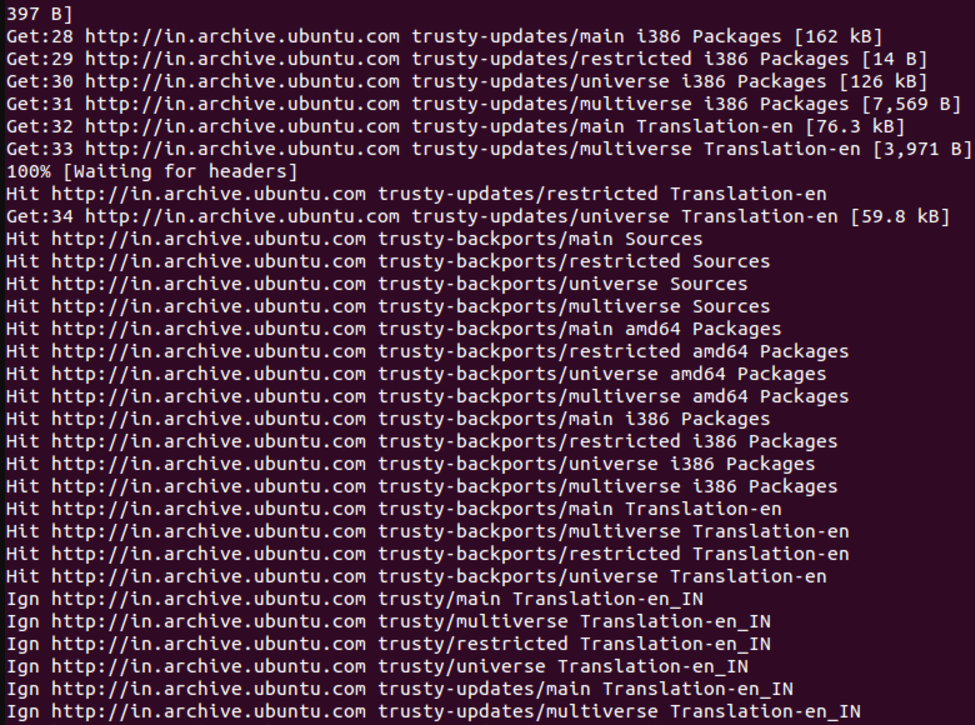
First of all I have updated the source list by using following command in compiler:

sudo apt-get update

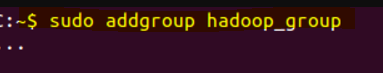
following is screenshot:



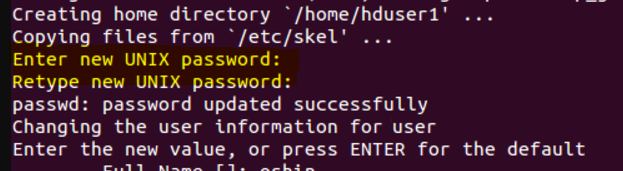
Then we add hadoop user account for running hadoop by using following command:

sudo addgroup hadoop\_group

sudo adduser --ingroup hadoop\_group user







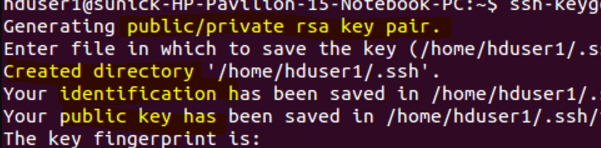
Add user to the sudo group

sudo adduser hduser1 sudo

We have to generate an SSH key for the user user.

su – user

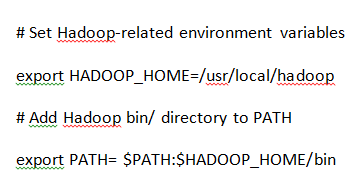
ssh-keygen -t rsa -P ""



I enabled SSH access to my local machine with this newly created key which is done by the following command.

cat $HOME/.ssh/id\_rsa.pub >> $HOME/.ssh/authorized\_keys

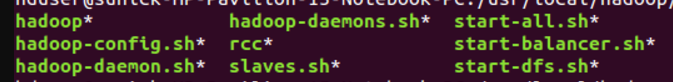
I added the following entries to .bashrc file

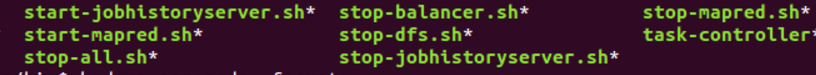


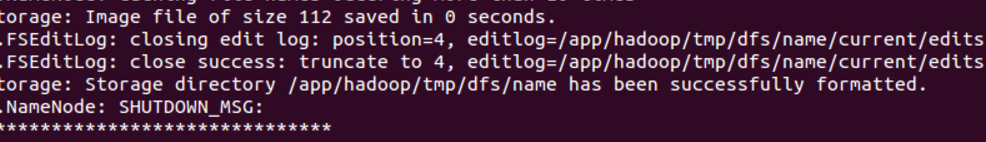
I ran the command to Namenode, Datanode, Jobtracker and a Tasktracker on the machineby following statement



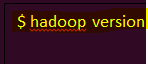
This gives us following results:



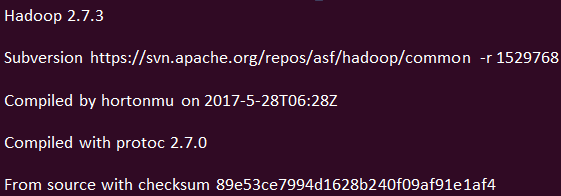




In order to check if hadoop is working properly we do following:



Then it gives us specifications of hadoop:

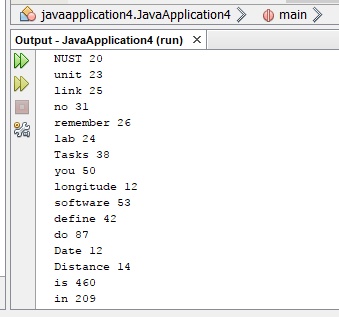


Hadoop has been installed and configured completely.

I have taken help from tutorials for installing and configuring from following website:

<http://doctuts.readthedocs.io/en/latest/hadoop.html>

**Final result of lab:**



I created two classes called mapper function and reducer function.

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

H =<dfs.http.address,50070>;

Reduce=Dfs.datanode.http.address<50075>;

Mapper m= mapred.job.tracker.http.address<50030>;

Main-Class: reduce\_function

import java.io.BufferedReader;

import org.apache.hadoop.io.LongWritable;

hadoop.log.dir.

/dfsnodelist.jsp?whatNodes=(DEAD|LIVE)

/listPaths/user/philip?filter=file.\*

/data

/filechecksum

/browseBlock.jsp.

/browseDirectory.jsp.

/streamFile

/getFileChecksum

/browseDirectory.tail.

import java.io.FileReader;

/metrics?format=json

import java.io.IOException;

package beginnersbook.com;

import java.util.HashMap;

import java.util.Map;

import java.util.Iterator;import java.io.IOException;

import org.apache.hadoop.mapreduce.Reducer;

import java.util.Set;

package mapper\_function;

package reduce\_function;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import java.awt.Frame;

import java.awt.Label;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

export JAVA HOME=/usr/java/default

export PATH=${JAVA\_HOME}/bin:${PATH}

export HADOOP\_CLASSPATH=${JAVA\_HOME}/lib/tools/jar

/\*\*

\*

\* @author test1

\*/

public class mapper\_function {

/\*\*

\* @param args the command line arguments

\*/public class master{}

Public Class slave{}

class mapperfunction{

public static void main(String[] args) {FILE \*lab1,\*lab2, \*lab3,\*lab4, \*lab5, \*lab6, \*lab7,

\*lab8, \*lab9, \*lab10, \*lab11, \*lab12;

List<\*FILE> records = new ArrayList<\*FILE>(); List<String>L=new List<String>;

Records(1)=fopen(”lab1.txt”);

Records(2)= fopen(”lab2.txt”);

Records(3)= fopen(”lab3.txt”);

Records(4)= fopen(”lab4.txt”);

Records(5)= fopen(”lab5.txt”);

Records(6)= fopen(”lab6.txt”);

Records(7)= fopen(”lab7.txt”);

Records(8)= fopen(”lab8.txt”);

Records(9)= fopen(”lab9.txt”);

Records(10)= fopen(”lab10.txt”);

Records(11)= fopen(”lab11.txt”);

Records(12)= fopen(”lab12.txt”);

string str[999]; FILE \* file;

file = fopen( "test.txt" , "w");

For(int i=1;i<=12;i++){file= Records(i);

if (file) {

map(file);

}

}

void map(FILE\* textfile){

string[] words=file;

for(string i=textfile.begin();i!="\0";i++)

{ string.delimiter(" ");

HashMap<String, integer > h = new HashMap< String, integer >;

h .put<key,1 >;

}

Reduce\_function r=new reduce\_function();r.reduce(h);

}

}

public class reduce\_function{

void reduce(hashmap<string,int> filewords){int p=0;

for(int i=filewords.value.begin();i!="\0";i++){

for(int j=filewords.value.begin-i;j!="\0";j++){

if(i.key()==j.key()&&i!=j)

{

p=i.value()+j.value();

}

}

jar cfm Manifest.txt reduce\_function.jar reduce\_function.class;

}

}}

Void print(){HashMap<String,integer>h=new HashMap<String,integer>;

h=hash;

For(i=hashmap.begin();i<=h.end();i++)

System.out.println(h.key);

System.out.println(h.value);

System.out.println(“\n”);

}

}