# TABLE OF CONTENTS

# INTRODUCTION

**Big Data** is a broad term for work with datasets so large or complex that traditional data processing applications are inadequate and distributed databases are needed. Challenges include sensor design, capture, data creation, search, sharing, storage, transfer, analysis, fusion, visualization, and information privacy.

Here, we will be doing outlier detection on large datasets using k-means clustering on Hadoop Map Reduce Paradigm.

**Map Reduce** is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster.Conceptually similar approaches have been very well known since 1995 with the message passing interface standard having reduce and scatter operations.

**K-means Clustering** is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

**Outlier** is an observation point that is distant from other observations. An outlier may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the data set.

## ABSTRACT

### INTRODUCTION:

**Outlier Detection** is the identification of items, events or observations which do not conform to an expected pattern or other items in a dataset. Typically the anomalous items will translate to some kind of problem such as bank fraud, a structural defect, medical problems or errors in a text. Anomalies are also referred to as outliers, novelties, noise, deviations and exceptions.

**Map Reduce** is a programming model that lies above Hadoop distributed file system which performs computation through Map phase and Reduce phase. Shuffle and sorting is done implicitly between Map and Reduce phases.

**CONVENTIONAL SYSTEM AND ITS DRAWBACKS:**

Lack of efficiency in detecting outliers when the data is unstructured, unsorted and huge., although there are techniques that do outlier detection they are very slow when compared to this distributed paradigm.

**PROPOSED SYSTEM**:

Distributed computing using Hadoop Distributed File System where data is divided into splits and stored across the environment. The output is visualized using R Programming.

**SOFTWARE ENVIROMENT:** Ubuntu operating system.

**SOFTWARE REQUIREMENTS:** Hadoop 2.0 , RHadoop 1.0.6

**PRODUCT SYSTEM:**

Outlier detection is faster and more efficient when the data is huge because of parallel processing on distributed systems through map-reduce.

**APPLICATIONS:** Anomaly detection is applicable in a variety of domains, such as intrusion detection, fraud detection, fault detection, system health monitoring, event detection in sensor networks, and detecting Eco-system disturbances.

# LITERATURE REVIEW

**Cluster Analysis** or **Clustering**[1] is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters).

Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a

[1]

cluster and how to efficiently find them. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions. Clustering can therefore be formulated as a multi-objective optimization problem.

**Clustering Categories**:

* Connectivity based clustering - Connectivity based clustering, also known as hierarchical clustering, is based on the core idea of objects being more related to nearby objects than to objects farther away.

* Centroid-based clustering - In centroid-based clustering, clusters are represented by a central vector, which may not necessarily be a member of the data set. When the number of clusters is fixed to k, k-means clustering gives a formal definition as an optimization problem: find the cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized.

* Distribution-based clustering - The clustering model most closely related to statistics is based on distribution models. Clusters can then easily be defined as objects belonging most likely to the same distribution.

* Density-based clustering - In density-based clustering, clusters are defined as areas of higher density than the remainder of the data set. Objects in these sparse areas - that are required to separate clusters - are usually considered to be noise and border points.

**K-means applications:**

k-means clustering, in particular when using heuristics such as Lloyd's algorithm, is rather easy to implement and apply even on large data sets. As such, it has been successfully used in various topics, including market segmentation, computer vision, geostatistics, astronomy and agriculture. It often is used as a preprocessing step for other algorithms, for example to find a starting configuration

## SOFTWARE REQUIREMENT SPECIFICATION

**SOFTWARES REQUIRED** :

Ubuntu ( Linux operating system) , Hadoop 2.7.0, eclipse, java jdk 7

## K-MEANS CLUSTERING WITH MAP-REDUCE PARADIGM

K-Means clustering is a popular algorithm for clustering similar objects into K groups (clusters). It starts with an initial seed of K points (randomly chosen) as centers, and then the algorithm iteratively tries to enhance these centers. The algorithm terminates either when two consecutive iterations generate the same K centers, i.e., the centers did not change, or a maximum number of iterations is reached.

We write map--reduce job that implement the K--Means clustering algorithm. The algorithm will terminate if either of these two conditions become true:

a) The K centers did not change over two consecutive iterations

b) The maximum number of iterations has reached.

The reducer will indicate in its output file whether centers have changed or not.

**Mapper**

OUTPUT: <centroid,point> key is the center point of the cluster which the point from value belongs to

**Combiner**

INPUT: Mapper Output OUTPUT:<centroid, string(point count) the value will be used to calculate new centroid in reducer

**Reducer**

INPUT:

Combiner Output OUTPUT:<new_centroid,count_of_current_unchanged_centroid_point>

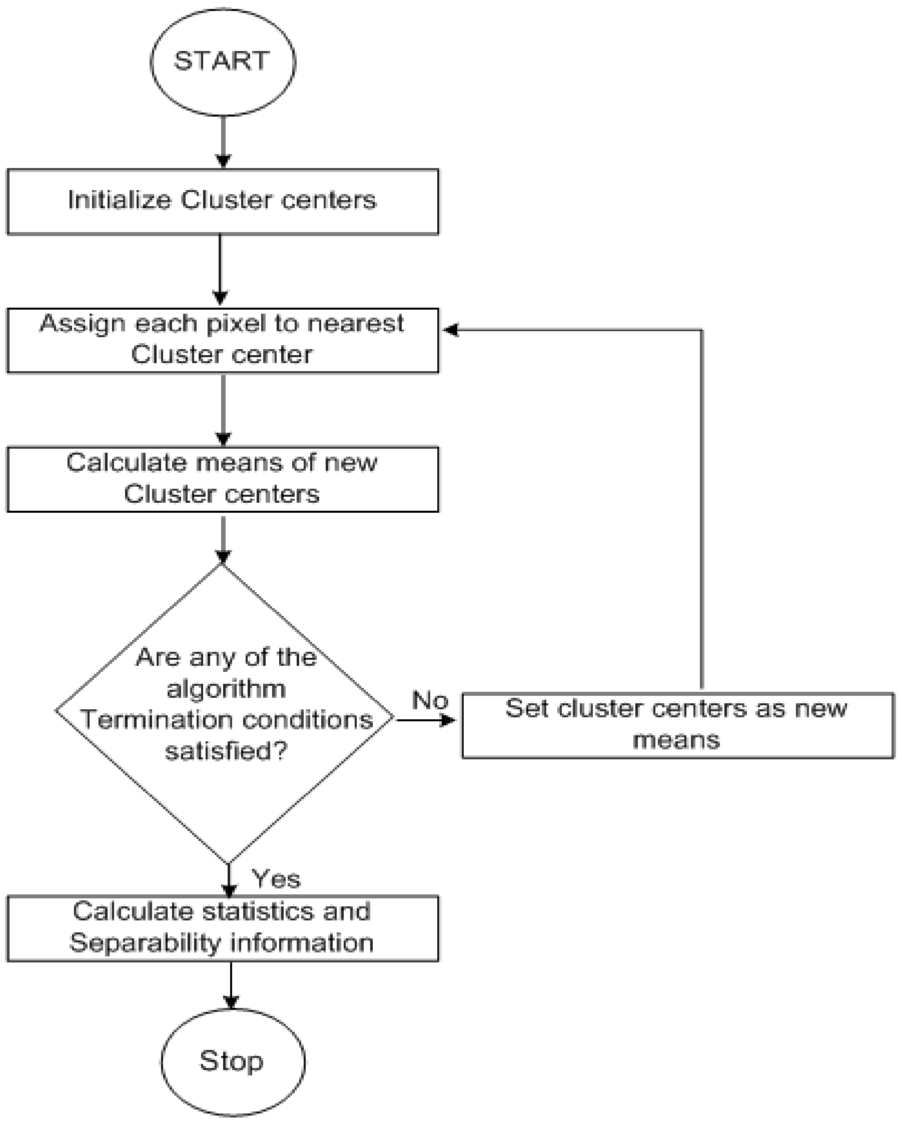

START
Initialize Cluster centers
Assign each pixel to nearest Cluster center
Calculate means of new Cluster centers
Are any of the algorithm Termination conditions satisfied?
No
Set cluster centers as new means
Yes
Calculate statistics and Separability information
Stop

Fig: K- means Clustering on Map Reduce Paradigm flow chart

**ALGORITHM FOR map ( key , value)**

**Input:** Global variable centers, the offset key, the sample value

**Output**: <key, value> pair, where the key is the index of the closest center point an value is a string comprise of sample information.

```
1. Construct the sample instance from value;
2. MinDis = Double.MAX_VALUE;
3. Index = -1;
4. For I=0 to centers.length do
5. Dis = ComputeDist(instance, centers[i]);
        6. If dis < minDis{
        7. minDis = dis;
        8. Index = I;
        9. }
10. End For
11. Take index as key;
12. Construct value as a string comprise of the values of different dimensions;
13. Output < key , value > pair;
14. End
```

**ALGORITHM FOR combine( key , V )**

**Input**: key is the index of the cluster, V is the list of the samples assigned to the same cluster **Output**: < key ,value > pair, where the key is the index of the cluster, value is a string comprised of sum of the samples in the same cluster and the sample number

```
1. Initialize one array to record the sum of value of each dimensions of the samples contained in the same cluster, i.e. the samples in the list V;
2. Initialize a counter num as 0 to record the sum of sample number in the same cluster;
3. while(V.hasNext()){ Construct the sample instance from V.next(); Add the values of different dimensions of instance to the array num++;
4. }
5. Take key as key;
6. Construct value as a string comprised of the sum values of different dimensions and num;
7. output < key ,value  > pair;
8. End
```

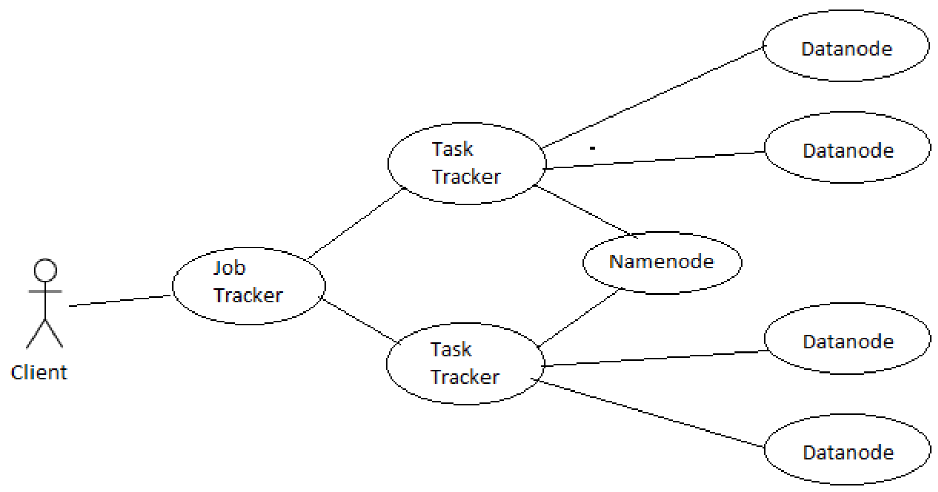
**ALGORITHM FOR reduce ( key ,V )**

**Input:** key is the index of the cluster, V is the list of the partial sums from different host.

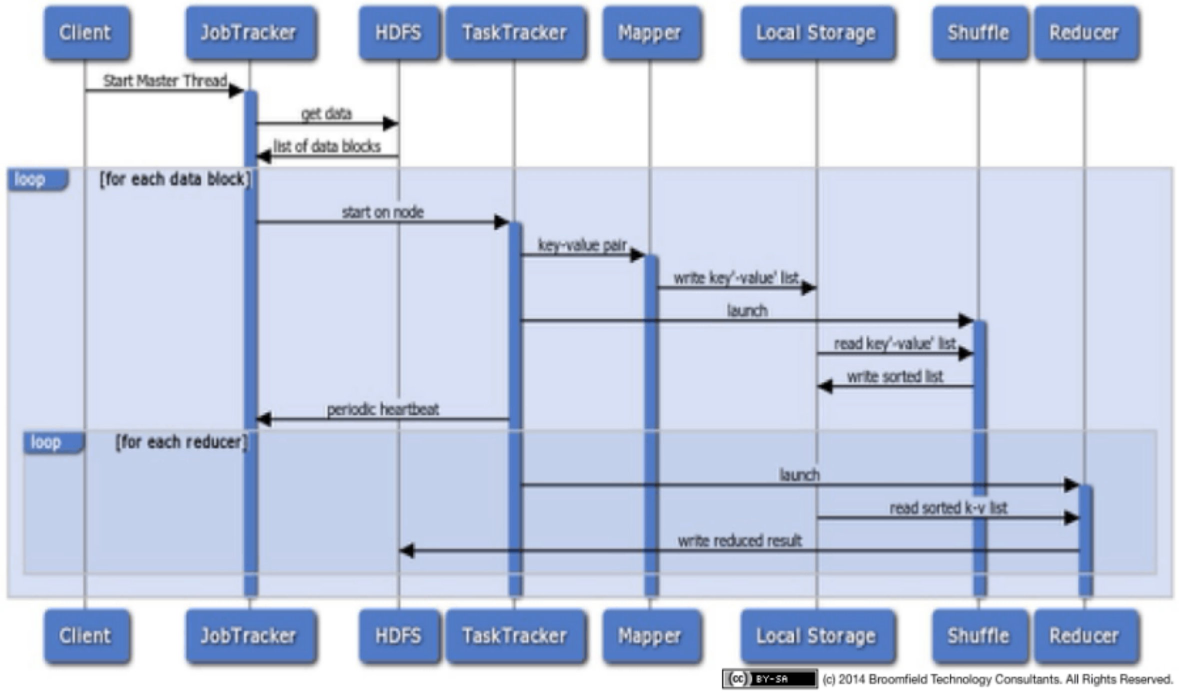
**Output:** < key ,value > pair, where the key is the index of the cluster, value is a string representing the new center

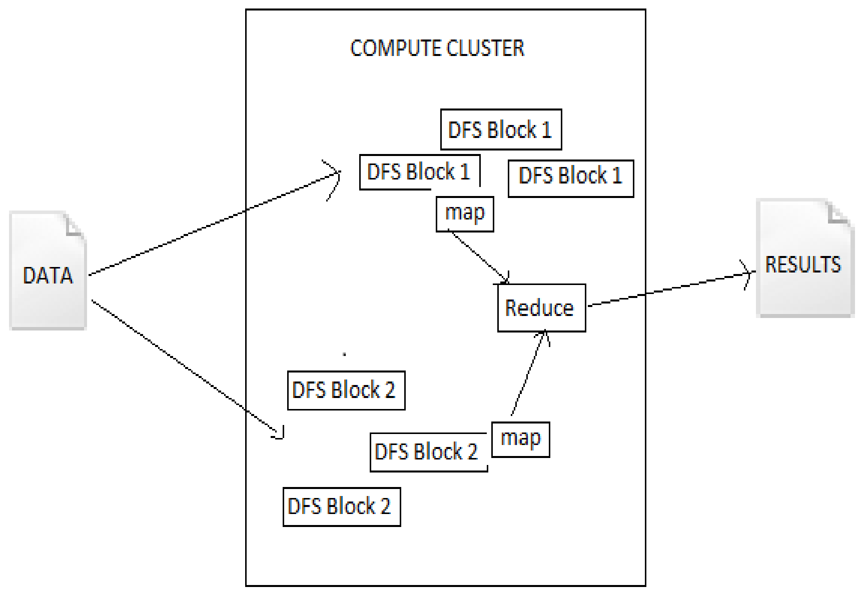
```
1. Initialize one array record the sum of value of each dimensions of the samples contained in the same cluster, e.g. the samples in the list V;
2. Initialize a counter NUM as 0 to record the sum of sample number in the same cluster;
3. while(V.hasNext()){ Construct the sample instance from V.next(); Add the values of different dimensions of instance to the array NUM += num;
4. }
5. Divide the entries of the array by NUM to get the new center's coordinates;
6. Take key as key;
7. Construct value' as a string comprise of the center's coordinates;
8. output < key ,value  > pair;
9. End
```

**USECASE DIAGRAM:**

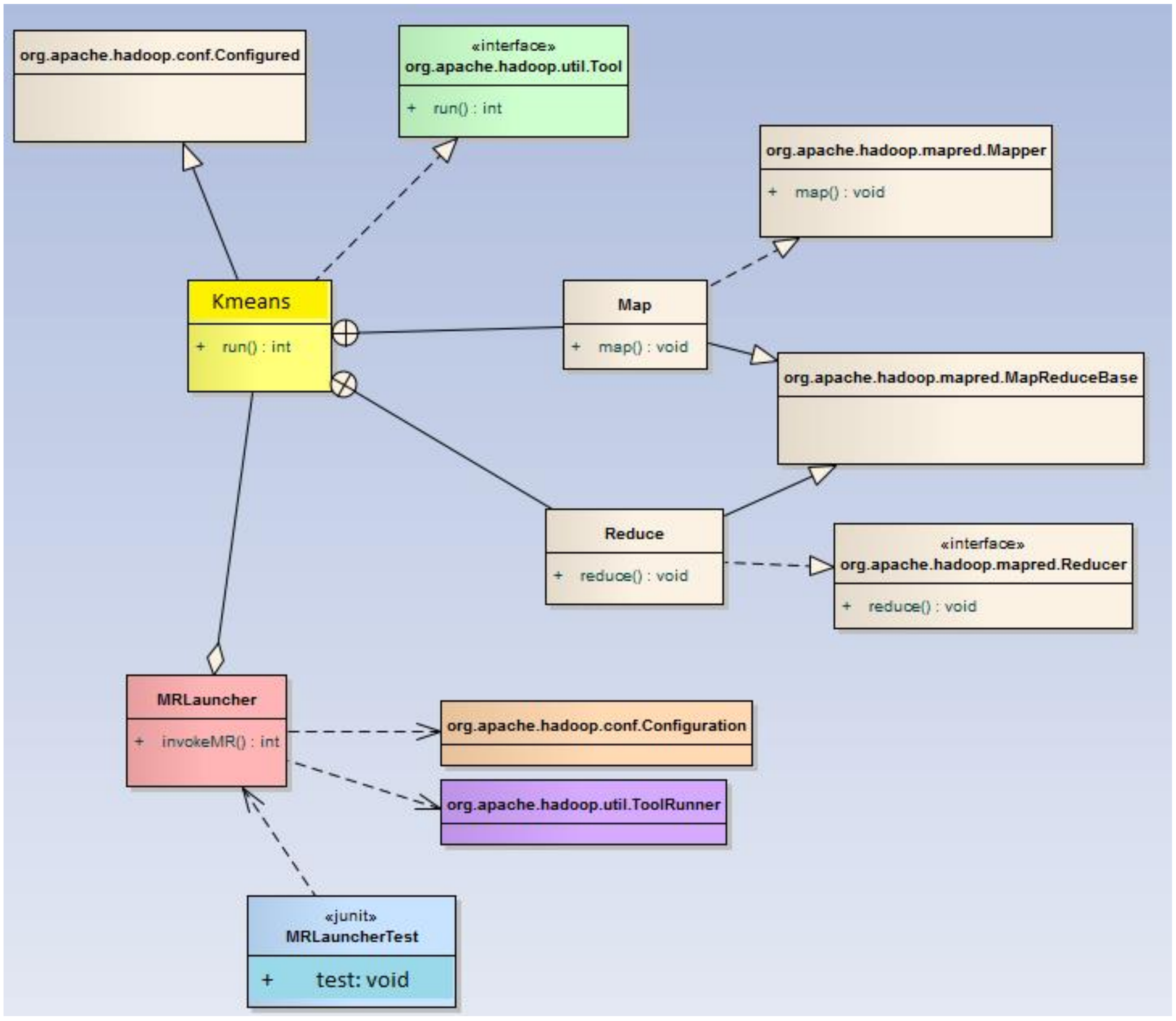


## Hadoop MapReduce Sequence Diagram





**COMPONENT DIAGRAM**


org.apache.hadoop.conf.Configured
«interface»
org.apache.hadoop.util.Tool
+ run() : int
org.apache.hadoop.mapred.Mapper
+ map() : void
Kmeans
+ run() : int
Map
+ map() : void
org.apache.hadoop.mapred.MapReduceBase
Reduce
+ reduce() : void
«interface»
org.apache.hadoop.mapred.Reducer
+ reduce() : void
MRLauncher
+ invokeMR() : int
org.apache.hadoop.conf.Configuration
org.apache.hadoop.util.ToolRunner
«junit»
MRLauncherTest
+ test: void

**CLASS DIAGRAM FOR MAP-REDUCE PARADIGM**

# INSTALLATION

For configuring Hadoop multi-node cluster there are some prerequisites to be satisfied.

- Java
- Hadoop Single node installation

**JAVA INSTALLATION:** Hadoop requires a working Java 1.5+ (aka Java 5) installation. However, using Java 1.6 (aka Java 6) is recommended for running Hadoop. For installing java, Ubuntu provides a command which directly installs java onto the system

1. Update the source list by - "**sudo apt-get update**". This command updates the required source list with latest feature.
2. Now install sun jdk by - "**sudo apt-get install sun-java6-jdk** ". This command installs sun Jdk onto the system. Java6 represents the version.
3. Now we can check whether java is installed or not by - " **java –version**", "**echo $JAVA_HOME**". These commands tell us whether java is installed or nor and the path it is installed.

**HADOOP SINGLE-NODE INSTALLATION:**

1. **Adding dedicated hadoop user:** During installation of Hadoop it is recommended to add a dedicated user as it keeps hadoop directory away from system programs. This can be done by

" **sudo addgroup hadoop**", "**sudo adduser --ingroup hadoop hduser**"

This creates a user named hduser and adds this hduser to hadoop.

2. **Enabling 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.

**ssh** has two main components:

**ssh** : The command we use to connect to remote machines - the client.

**sshd** : The daemon that is running on the server and allows clients to connect to the server.

The **ssh** is pre-enabled on Linux, but in order to start **sshd** daemon, we need to install **ssh** first. Use this command to do that : "**sudo apt-get install ssh"**

This installs ssh onto the local system and it can be checked by command "**which ssh"**

3. **Creating and Setting-up SSH Certificates:** Hadoop requires SSH access to manage its nodes, i.e. remote machines plus our local machine. For our single-node setup of Hadoop, we therefore need to configure SSH access to localhost.

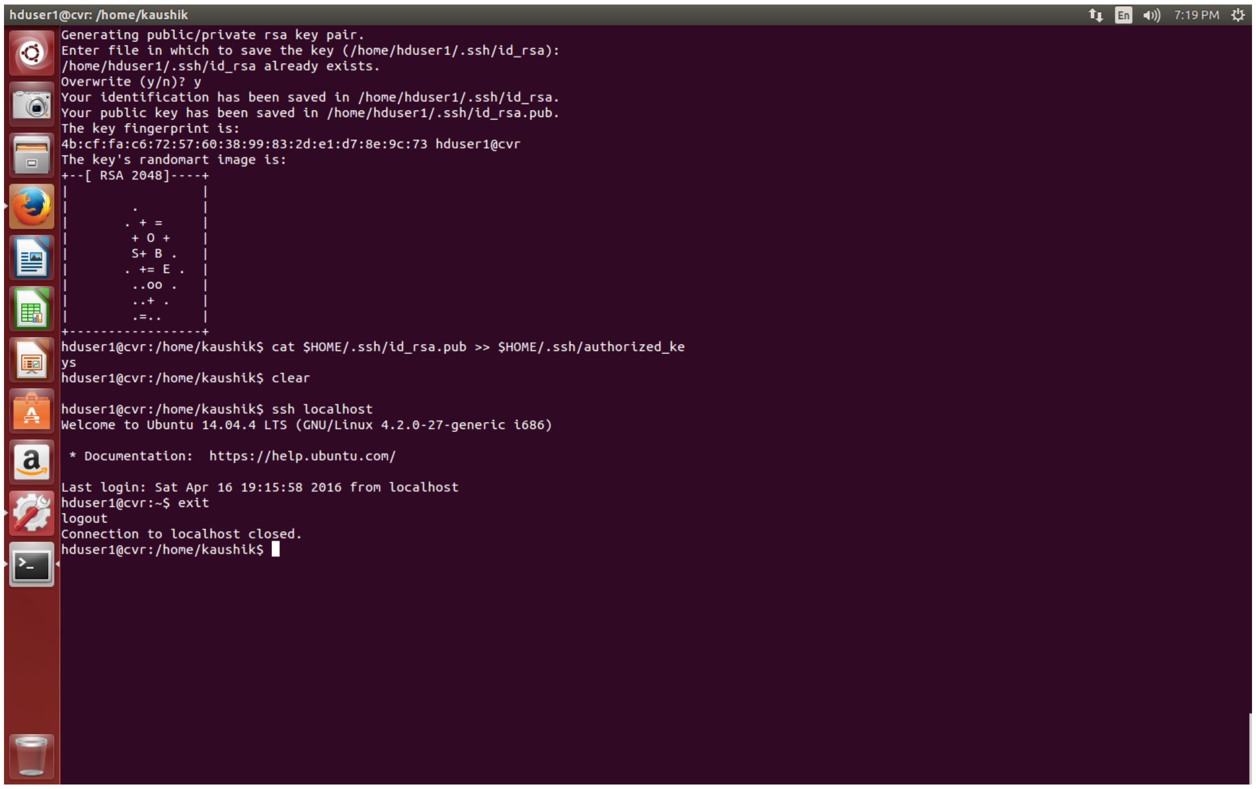
So, we need to have SSH up and running on our machine and configured it to allow SSH public key authentication.

Hadoop uses SSH (to access its nodes) which would normally require the user to enter a password. However, this requirement can be eliminated by creating and setting up SSH certificates using the following commands.

The second command adds the newly created key to the list of authorized keys so that Hadoop can use ssh without prompting for a password. Now we have to add the newly generated key to the key table by:

**"cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys"**

The final step is to test the SSH setup by connecting to your local machine with the hduser. This can be done by the command – **"ssh localhost"**



```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hduser1/.ssh/id_rsa):
/home/hduser1/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Your identification has been saved in /home/hduser1/.ssh/id_rsa.
Your public key has been saved in /home/hduser1/.ssh/id_rsa.pub.
The key fingerprint is:
4b:cf:fa:c6:72:57:60:38:99:83:2d:e1:d7:8e:9c:73 hduser1@cvr
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|        .        |
|       . + =     |
|        + O +    |
|        S+ B .   |
|       . += E .  |
|        ..oo .   |
|        ..+ .    |
|        .=..     |
+-----------------+
hduser1@cvr:/home/kaushik$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_ke
ys
hduser1@cvr:/home/kaushik$ clear

hduser1@cvr:/home/kaushik$ ssh localhost
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic i686)

 * Documentation:  https://help.ubuntu.com/

Last login: Sat Apr 16 19:15:58 2016 from localhost
hduser1@cvr:~$ exit
logout
Connection to localhost closed.
hduser1@cvr:/home/kaushik$
```

4. **Hadoop Single Node:** Firstly we need to download Hadoop from one of its Apache Hadoop's official website

http://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-2.7.2/hadoop-2.7.2.tar.gz

The latest version available is Hadoop-2.7.2 which is stable .

- After the download is complete we need to unzip the .tar file which can be done by – "**sudo tar xzf hadoop-1.0.3.tar.gz".**
- Now we need to move the extracted hadoop to /usr/local folder such that hadoop is available to all users and give all permissions to it by – "**sudo chown -R /usr/local/hadoop**".
- Now we need to set the Configuration files

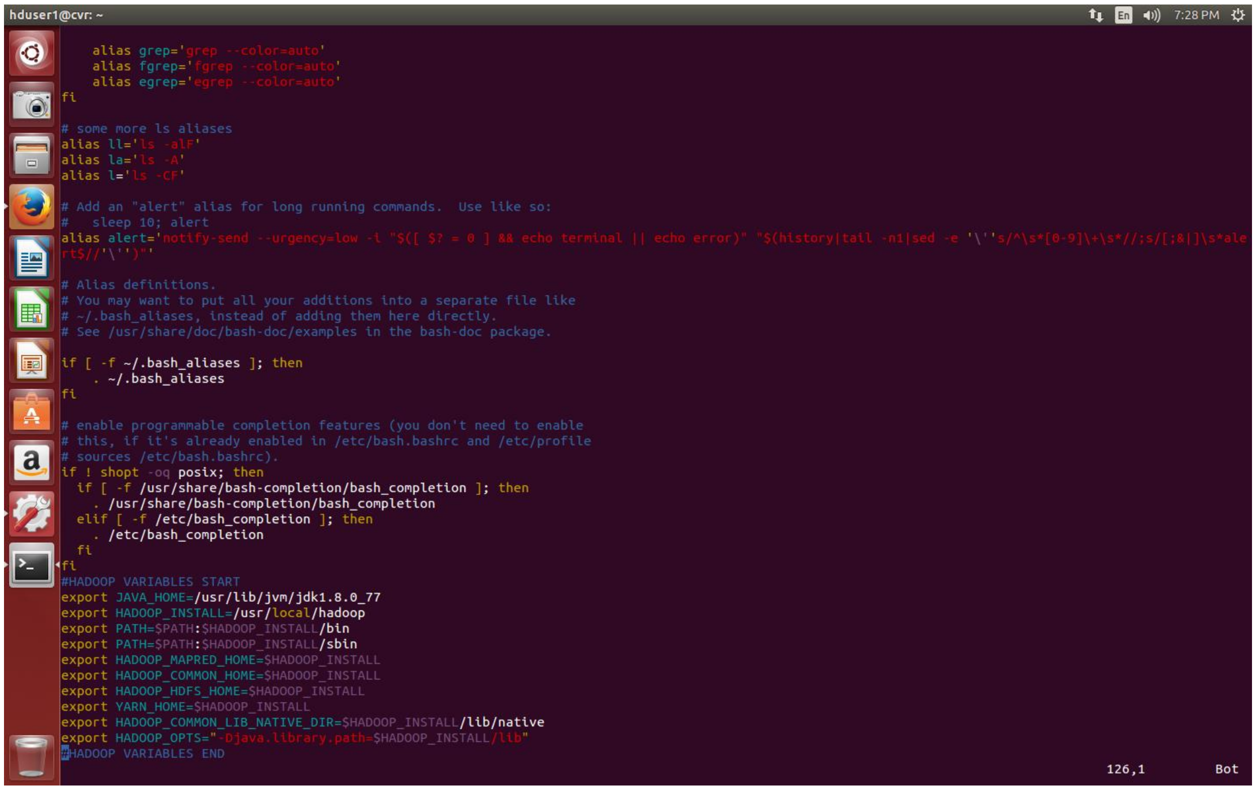
  (a) ~/.bashrc

  (b) /usr/local/hadoop/etc/hadoop/core-site.xml

  (c) /usr/local/hadoop/etc/hadoop/mapred-site.xml

  (d) /usr/local/hadoop/etc/hadoop/hdfs-site.xml

**~/.bashrc :** The .bashrc file can be used to setup the environment, export variables, create aliases and functions etc. Multiple instances can be inserted here.



```
hduser1@cvr: ~                                                    En  7:28 PM
    alias grep='grep --color=auto'
    alias fgrep='fgrep --color=auto'
    alias egrep='egrep --color=auto'
fi

# some more ls aliases
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands.  Use like so:
#   sleep 10; alert
alias alert='notify-send --urgency=low -i "$([ $? = 0 ] && echo terminal || echo error)" "$(history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[;&|]\s*ale
rt$//'\'')"'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
#HADOOP VARIABLES START
export JAVA_HOME=/usr/lib/jvm/jdk1.8.0_77
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=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
#HADOOP VARIABLES END
                                                              126,1         Bot
```

**core-site.xml:** The /usr/local/hadoop/etc/hadoop/core-site.xml file contains configuration properties that Hadoop uses when starting up. This directory in Hadoop will store its data files, the network ports it listens to, etc.

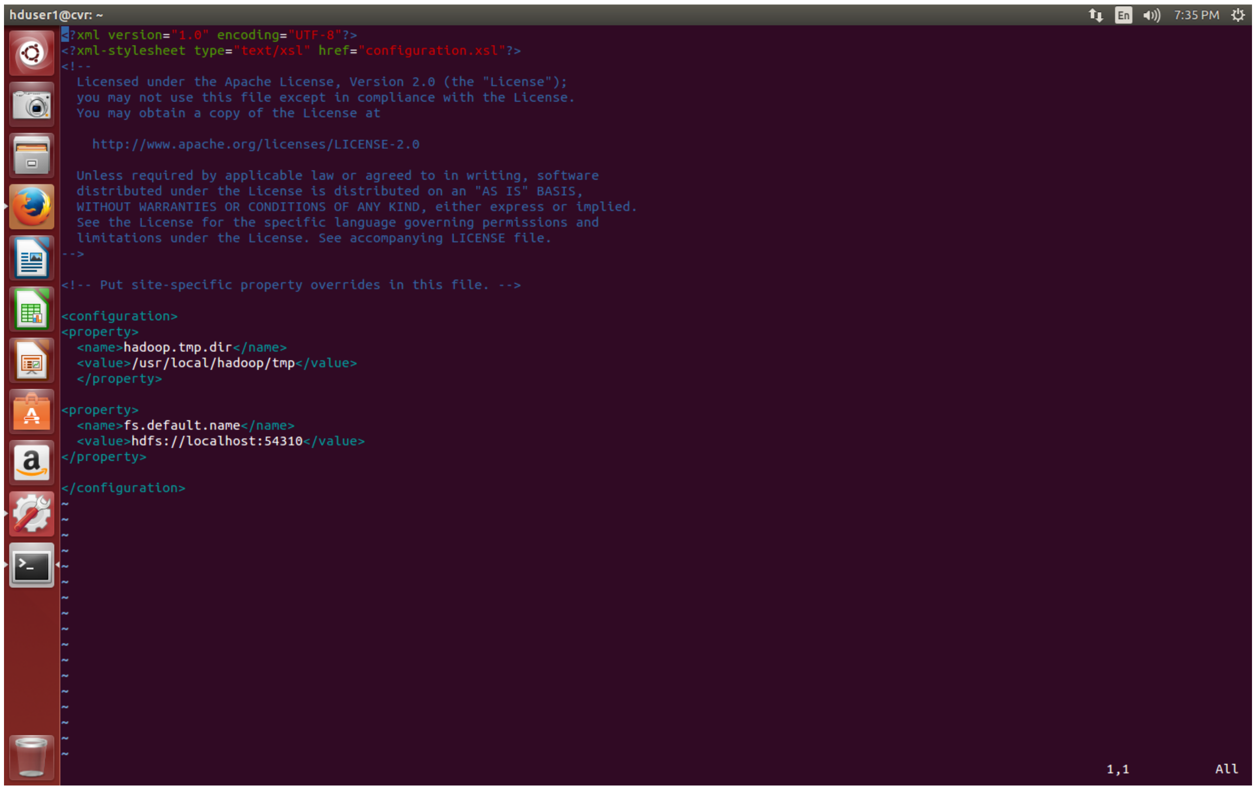
This file can be used to override the default settings that Hadoop starts with. For this we need to create a temporary directory for hadoop using commands

**"sudo mkdir -p /usr/local/hadoop/tmp"**

"**sudo chmod 777 /usr/local/hadoop/tmp**"

After the tmp directory is created we need to setup the core-site .xml

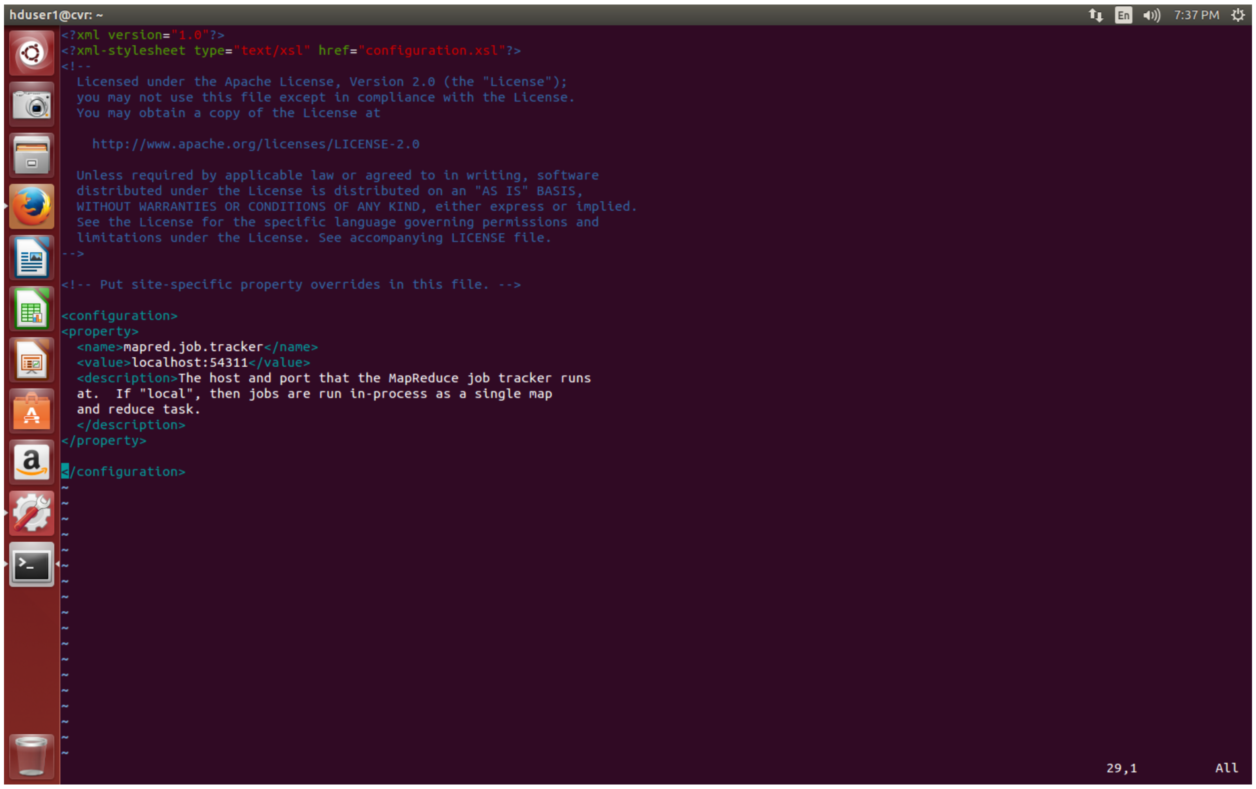
Add the snippets between the <configuration>----</configuration> tags in the respective configuration XML file.



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/usr/local/hadoop/tmp</value>
  </property>

<property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:54310</value>
</property>

</configuration>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                                      1,1           All
```

Now core-site.xml is ready.

**mapred-site.xml:** The mapred-site.xml file is used to specify which framework is being used for MapReduce. Add the snippets between the <configuration>----</configuration> tags in the respective configuration XML file.



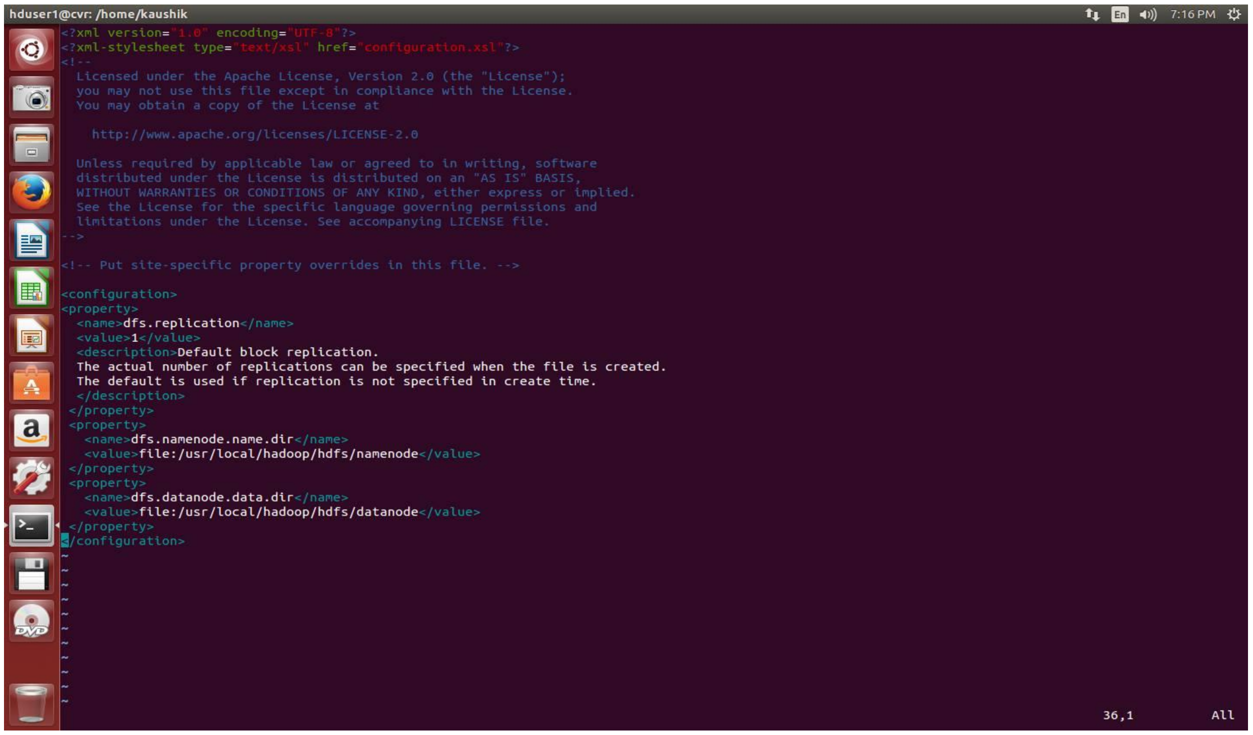
```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<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>

</configuration>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                              29,1          All
```

**hdfs-site.xml:** The **/usr/local/hadoop/etc/hadoop/hdfs-site.xml** file needs to be configured for each host in the cluster that is being used.It is used to specify the directories which will be used as the **namenode** and the **datanode** on that host.

Before editing this file, we need to create two directories which will contain the namenode and the datanode for this Hadoop installation.
This can be done using the following commands:

**“sudo mkdir -p /usr/local/hadoop/hdfs/namenode”**

**“sudo mkdir -p /usr/local/hadoop/hdfs/namenode”**

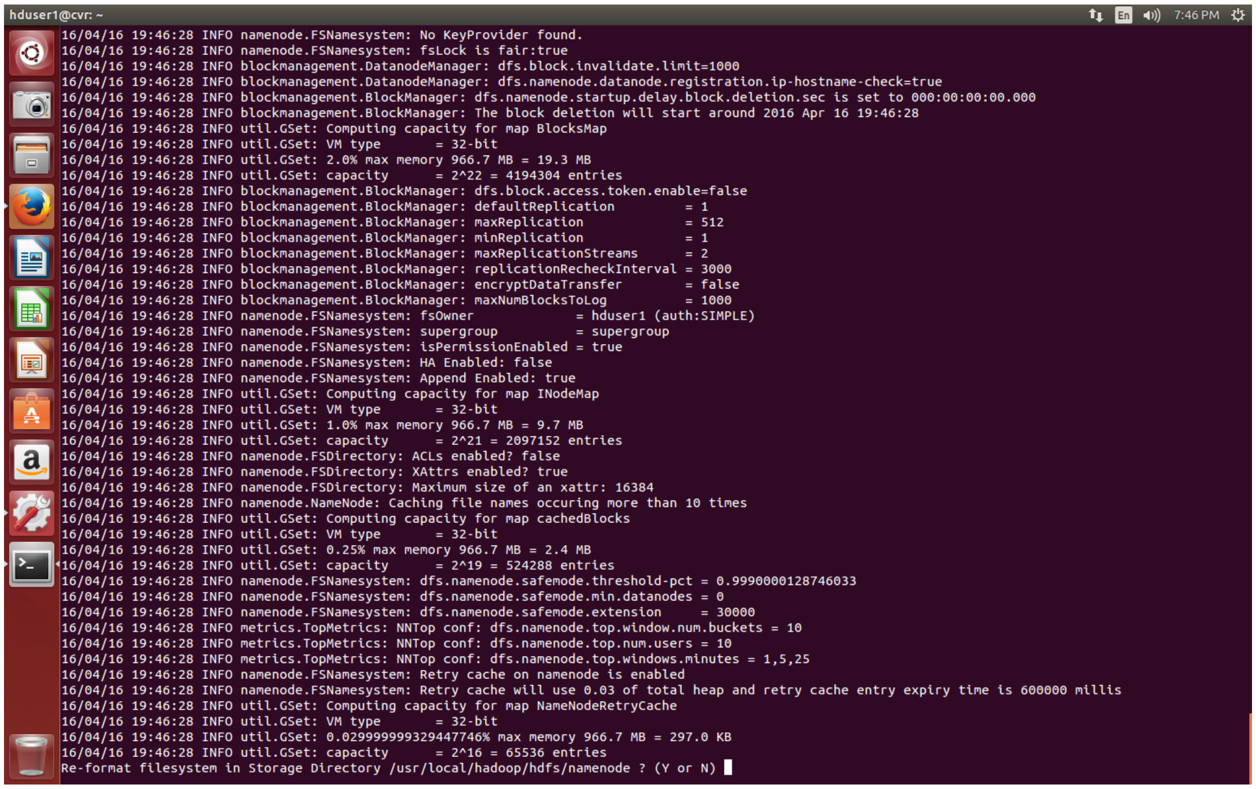


```
hduser1@cvr: /home/kaushik
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<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:/usr/local/hadoop/hdfs/namenode</value>
 </property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/usr/local/hadoop/hdfs/datanode</value>
 </property>
</configuration>
~
~
~
~
~
~
~
~
~
~
~
                                                                  36,1          All
```

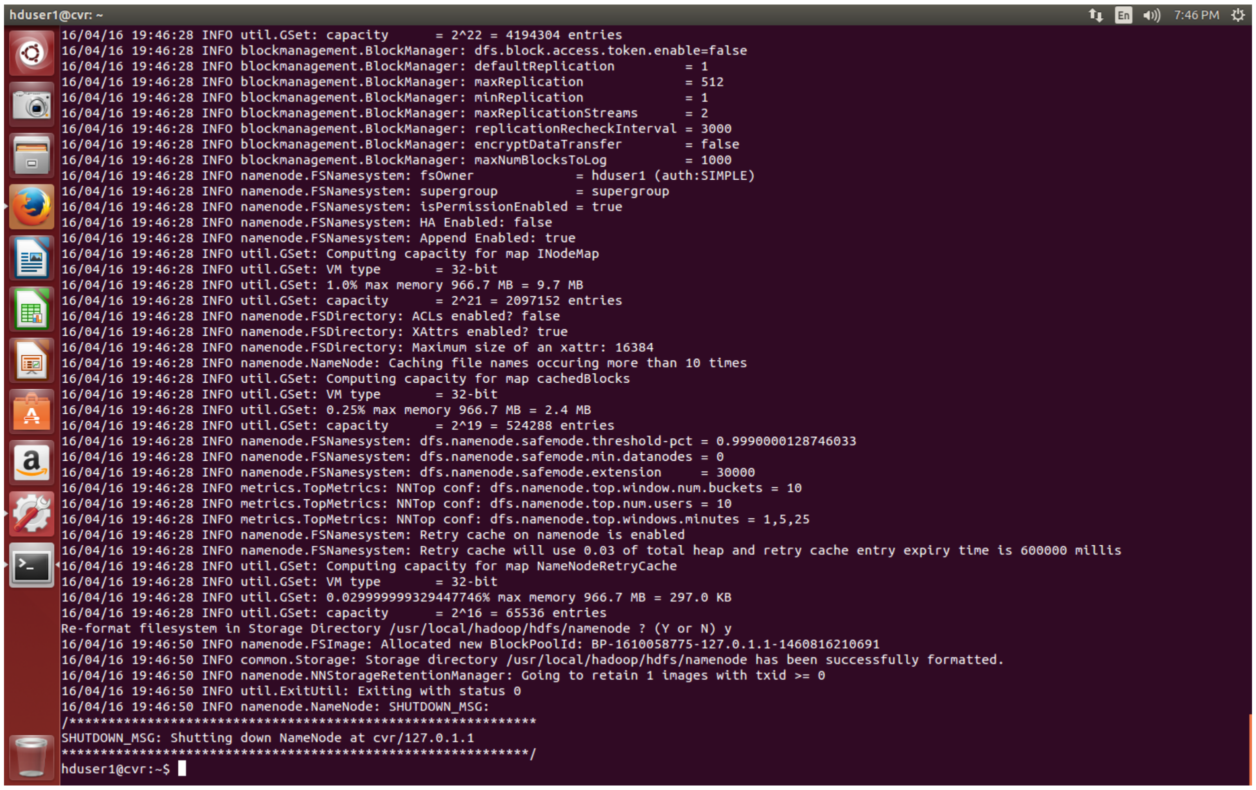
Open the hdfs-site.xml file and enter the following content in between the <configuration>-----------</configuration> tag.

**Format the New Hadoop Filesystem:** Now, the Hadoop file system needs to be formatted so that we can start to use it. The format command should be issued with write permission since it creates current directory under "**/usr/local/hadoop/hdfs/namenode**" folder:

Using – "**hadoop namenode -format**". This will format the hadoop namenode



```
16/04/16 19:46:28 INFO namenode.FSNamesystem: No KeyProvider found.
16/04/16 19:46:28 INFO namenode.FSNamesystem: fsLock is fair:true
16/04/16 19:46:28 INFO blockmanagement.DatanodeManager: dfs.block.invalidate.limit=1000
16/04/16 19:46:28 INFO blockmanagement.DatanodeManager: dfs.namenode.datanode.registration.ip-hostname-check=true
16/04/16 19:46:28 INFO blockmanagement.BlockManager: dfs.namenode.startup.delay.block.deletion.sec is set to 000:00:00:00.000
16/04/16 19:46:28 INFO blockmanagement.BlockManager: The block deletion will start around 2016 Apr 16 19:46:28
16/04/16 19:46:28 INFO util.GSet: Computing capacity for map BlocksMap
16/04/16 19:46:28 INFO util.GSet: VM type       = 32-bit
16/04/16 19:46:28 INFO util.GSet: 2.0% max memory 966.7 MB = 19.3 MB
16/04/16 19:46:28 INFO util.GSet: capacity      = 2^22 = 4194304 entries
16/04/16 19:46:28 INFO blockmanagement.BlockManager: dfs.block.access.token.enable=false
16/04/16 19:46:28 INFO blockmanagement.BlockManager: defaultReplication         = 1
16/04/16 19:46:28 INFO blockmanagement.BlockManager: maxReplication             = 512
16/04/16 19:46:28 INFO blockmanagement.BlockManager: minReplication             = 1
16/04/16 19:46:28 INFO blockmanagement.BlockManager: maxReplicationStreams      = 2
16/04/16 19:46:28 INFO blockmanagement.BlockManager: replicationRecheckInterval = 3000
16/04/16 19:46:28 INFO blockmanagement.BlockManager: encryptDataTransfer        = false
16/04/16 19:46:28 INFO blockmanagement.BlockManager: maxNumBlocksToLog          = 1000
16/04/16 19:46:28 INFO namenode.FSNamesystem: fsOwner             = hduser1 (auth:SIMPLE)
16/04/16 19:46:28 INFO namenode.FSNamesystem: supergroup          = supergroup
16/04/16 19:46:28 INFO namenode.FSNamesystem: isPermissionEnabled = true
16/04/16 19:46:28 INFO namenode.FSNamesystem: HA Enabled: false
16/04/16 19:46:28 INFO namenode.FSNamesystem: Append Enabled: true
16/04/16 19:46:28 INFO util.GSet: Computing capacity for map INodeMap
16/04/16 19:46:28 INFO util.GSet: VM type       = 32-bit
16/04/16 19:46:28 INFO util.GSet: 1.0% max memory 966.7 MB = 9.7 MB
16/04/16 19:46:28 INFO util.GSet: capacity      = 2^21 = 2097152 entries
16/04/16 19:46:28 INFO namenode.FSDirectory: ACLs enabled? false
16/04/16 19:46:28 INFO namenode.FSDirectory: XAttrs enabled? true
16/04/16 19:46:28 INFO namenode.FSDirectory: Maximum size of an xattr: 16384
16/04/16 19:46:28 INFO namenode.NameNode: Caching file names occuring more than 10 times
16/04/16 19:46:28 INFO util.GSet: Computing capacity for map cachedBlocks
16/04/16 19:46:28 INFO util.GSet: VM type       = 32-bit
16/04/16 19:46:28 INFO util.GSet: 0.25% max memory 966.7 MB = 2.4 MB
16/04/16 19:46:28 INFO util.GSet: capacity      = 2^19 = 524288 entries
16/04/16 19:46:28 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
16/04/16 19:46:28 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
16/04/16 19:46:28 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension     = 30000
16/04/16 19:46:28 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
16/04/16 19:46:28 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
16/04/16 19:46:28 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
16/04/16 19:46:28 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
16/04/16 19:46:28 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
16/04/16 19:46:28 INFO util.GSet: Computing capacity for map NameNodeRetryCache
16/04/16 19:46:28 INFO util.GSet: VM type       = 32-bit
16/04/16 19:46:28 INFO util.GSet: 0.029999999329447746% max memory 966.7 MB = 297.0 KB
16/04/16 19:46:28 INFO util.GSet: capacity      = 2^16 = 65536 entries
Re-format filesystem in Storage Directory /usr/local/hadoop/hdfs/namenode ? (Y or N)
```

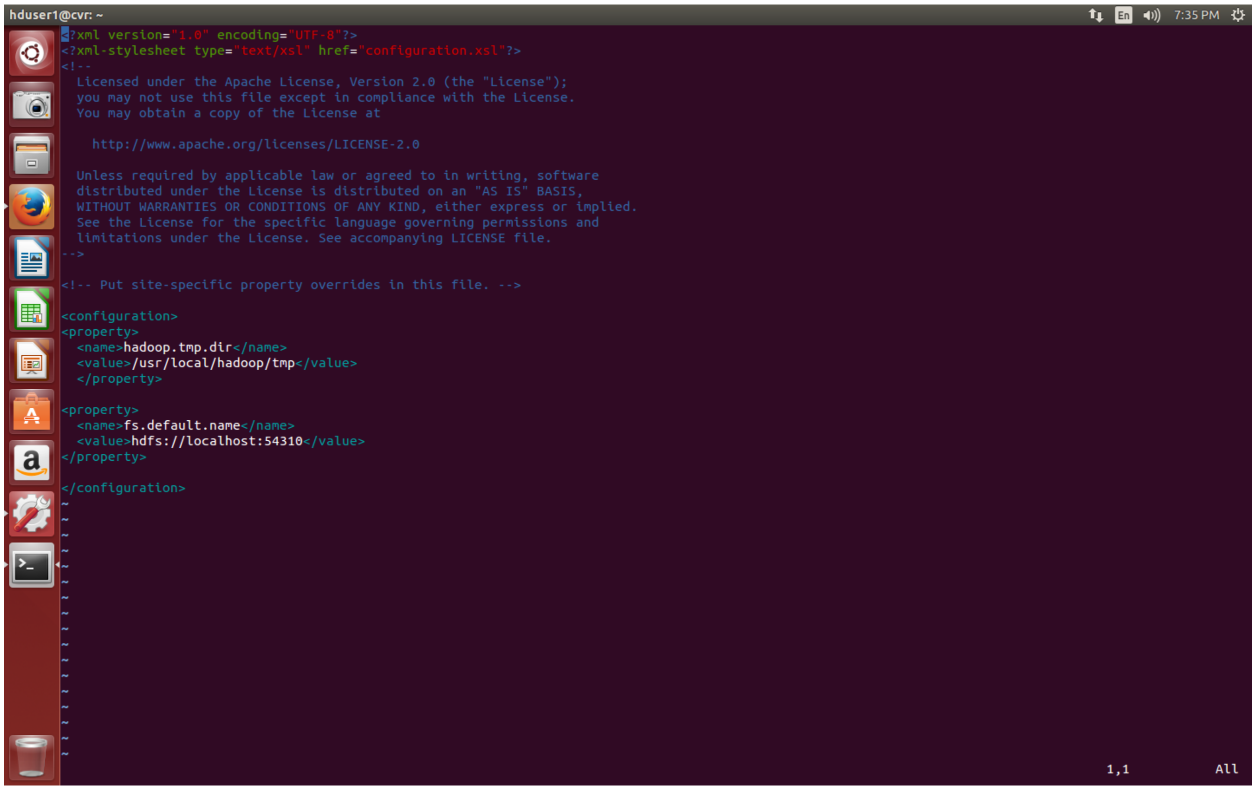


```
16/04/16 19:46:28 INFO util.GSet: capacity      = 2^22 = 4194304 entries
16/04/16 19:46:28 INFO blockmanagement.BlockManager: dfs.block.access.token.enable=false
16/04/16 19:46:28 INFO blockmanagement.BlockManager: defaultReplication         = 1
16/04/16 19:46:28 INFO blockmanagement.BlockManager: maxReplication             = 512
16/04/16 19:46:28 INFO blockmanagement.BlockManager: minReplication             = 1
16/04/16 19:46:28 INFO blockmanagement.BlockManager: maxReplicationStreams      = 2
16/04/16 19:46:28 INFO blockmanagement.BlockManager: replicationRecheckInterval = 3000
16/04/16 19:46:28 INFO blockmanagement.BlockManager: encryptDataTransfer        = false
16/04/16 19:46:28 INFO blockmanagement.BlockManager: maxNumBlocksToLog          = 1000
16/04/16 19:46:28 INFO namenode.FSNamesystem: fsOwner             = hduser1 (auth:SIMPLE)
16/04/16 19:46:28 INFO namenode.FSNamesystem: supergroup          = supergroup
16/04/16 19:46:28 INFO namenode.FSNamesystem: isPermissionEnabled = true
16/04/16 19:46:28 INFO namenode.FSNamesystem: HA Enabled: false
16/04/16 19:46:28 INFO namenode.FSNamesystem: Append Enabled: true
16/04/16 19:46:28 INFO util.GSet: Computing capacity for map INodeMap
16/04/16 19:46:28 INFO util.GSet: VM type       = 32-bit
16/04/16 19:46:28 INFO util.GSet: 1.0% max memory 966.7 MB = 9.7 MB
16/04/16 19:46:28 INFO util.GSet: capacity      = 2^21 = 2097152 entries
16/04/16 19:46:28 INFO namenode.FSDirectory: ACLs enabled? false
16/04/16 19:46:28 INFO namenode.FSDirectory: XAttrs enabled? true
16/04/16 19:46:28 INFO namenode.FSDirectory: Maximum size of an xattr: 16384
16/04/16 19:46:28 INFO namenode.NameNode: Caching file names occuring more than 10 times
16/04/16 19:46:28 INFO util.GSet: Computing capacity for map cachedBlocks
16/04/16 19:46:28 INFO util.GSet: VM type       = 32-bit
16/04/16 19:46:28 INFO util.GSet: 0.25% max memory 966.7 MB = 2.4 MB
16/04/16 19:46:28 INFO util.GSet: capacity      = 2^19 = 524288 entries
16/04/16 19:46:28 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
16/04/16 19:46:28 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
16/04/16 19:46:28 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension     = 30000
16/04/16 19:46:28 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
16/04/16 19:46:28 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
16/04/16 19:46:28 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
16/04/16 19:46:28 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
16/04/16 19:46:28 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
16/04/16 19:46:28 INFO util.GSet: Computing capacity for map NameNodeRetryCache
16/04/16 19:46:28 INFO util.GSet: VM type       = 32-bit
16/04/16 19:46:28 INFO util.GSet: 0.029999999329447746% max memory 966.7 MB = 297.0 KB
16/04/16 19:46:28 INFO util.GSet: capacity      = 2^16 = 65536 entries
Re-format filesystem in Storage Directory /usr/local/hadoop/hdfs/namenode ? (Y or N) y
16/04/16 19:46:50 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1610058775-127.0.1.1-1460816210691
16/04/16 19:46:50 INFO common.Storage: Storage directory /usr/local/hadoop/hdfs/namenode has been successfully formatted.
16/04/16 19:46:50 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
16/04/16 19:46:50 INFO util.ExitUtil: Exiting with status 0
16/04/16 19:46:50 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at cvr/127.0.1.1
************************************************************/
hduser1@cvr:~$
```

**Starting Hadoop:** Now it's time to start the newly installed single node cluster.

We can use “**start-all.sh”.**

We can check if it's really up and running using command “**jps**”



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/usr/local/hadoop/tmp</value>
  </property>

<property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:54310</value>
</property>

</configuration>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                            1,1           All
```

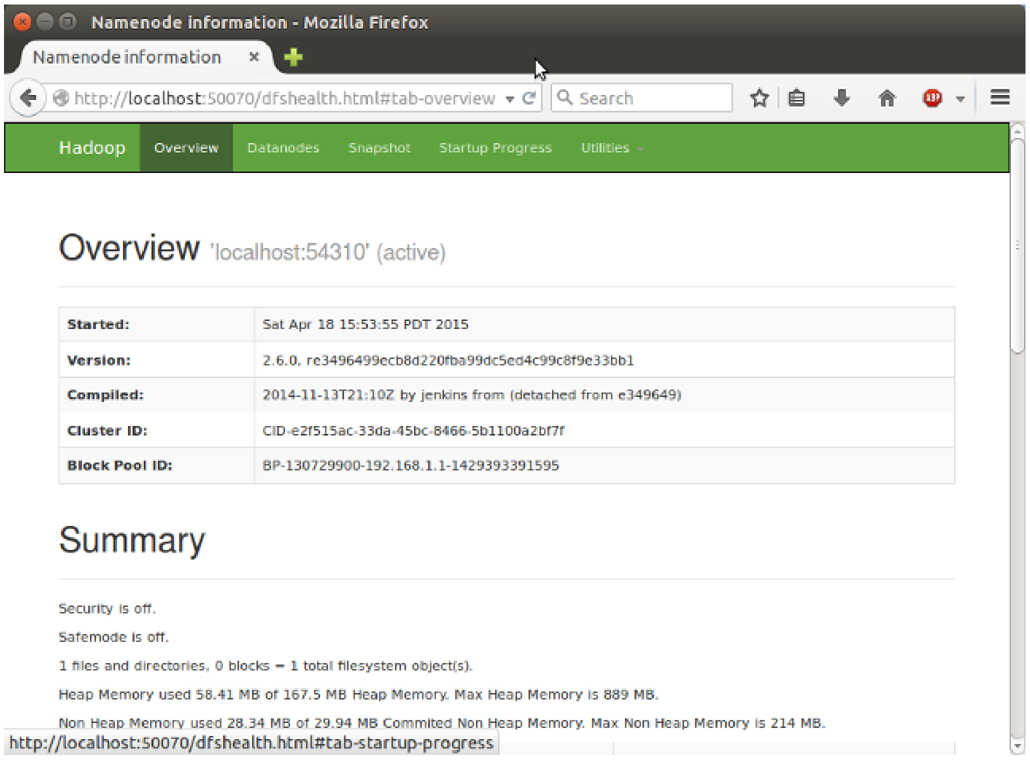
## Stopping Hadoop:

We run **stop-all.sh** or (**stop-dfs.sh** and **stop-yarn.sh**) to stop all the daemons running on our machine.

## Hadoop Web Interfaces:

Let's start the Hadoop again and see its Web UI:

**http://localhost:50070/** - web UI of the NameNode daemon


Namenode information - Mozilla Firefox
Namenode information
http://localhost:50070/dfshealth.html#tab-overview
Search
Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities

Overview 'localhost:54310' (active)

| | |
|---|---|
| **Started:** | Sat Apr 18 15:53:55 PDT 2015 |
| **Version:** | 2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1 |
| **Compiled:** | 2014-11-13T21:10Z by jenkins from (detached from e349649) |
| **Cluster ID:** | CID-e2f515ac-33da-45bc-8466-5b1100a2bf7f |
| **Block Pool ID:** | BP-130729900-192.168.1.1-1429393391595 |

Summary

Security is off.

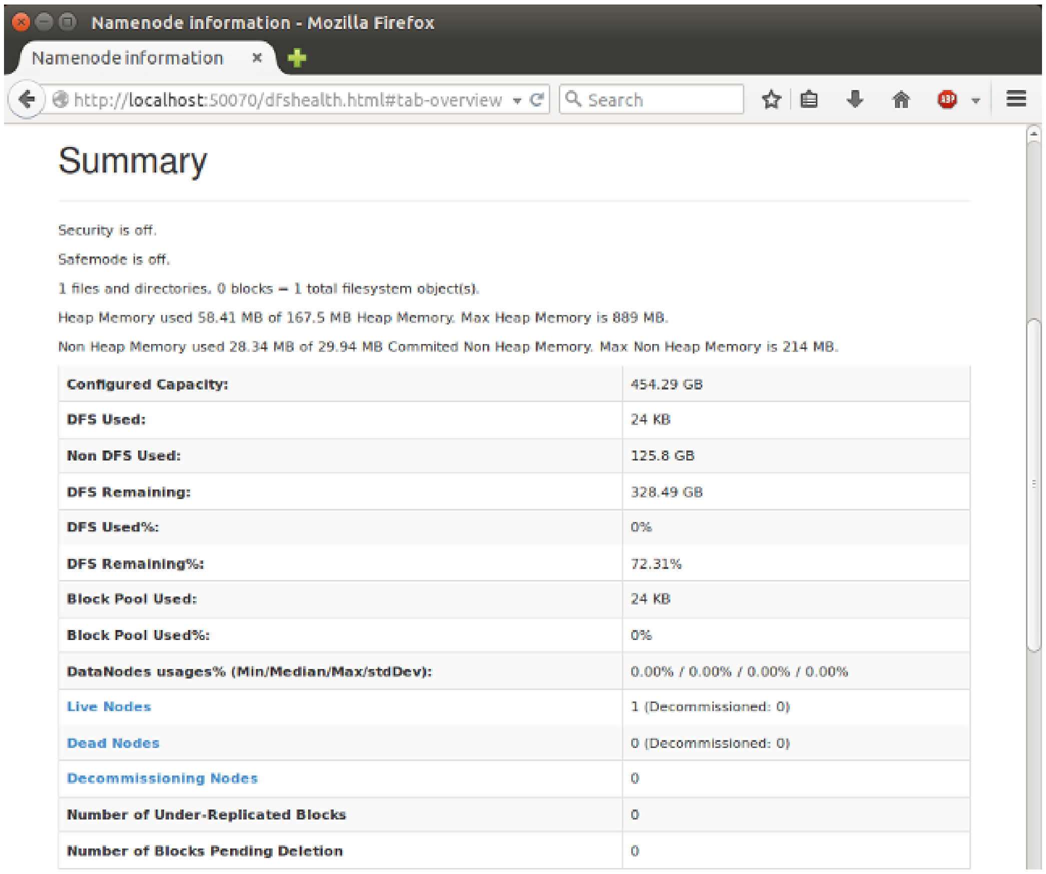
Safemode is off.

1 files and directories, 0 blocks = 1 total filesystem object(s).

Heap Memory used 58.41 MB of 167.5 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 28.34 MB of 29.94 MB Commited Non Heap Memory. Max Non Heap Memory is 214 MB.

http://localhost:50070/dfshealth.html#tab-startup-progress


Namenode information - Mozilla Firefox
Namenode information
http://localhost:50070/dfshealth.html#tab-overview
Search

Summary

Security is off.

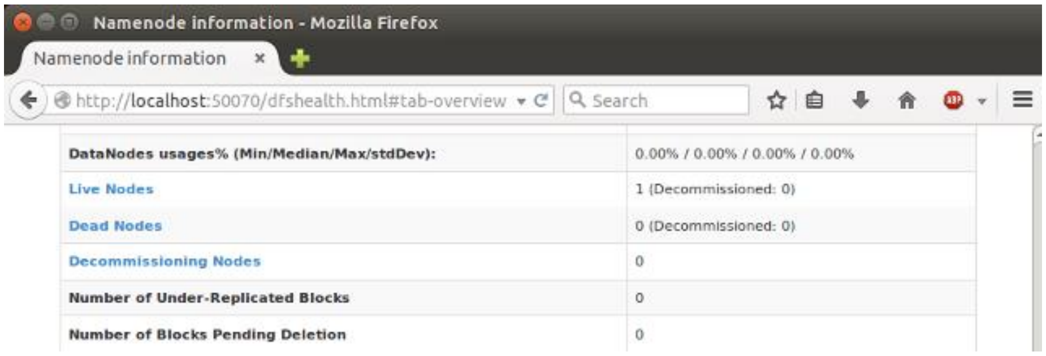
Safemode is off.

1 files and directories, 0 blocks = 1 total filesystem object(s).

Heap Memory used 58.41 MB of 167.5 MB Heap Memory. Max Heap Memory is 889 MB.

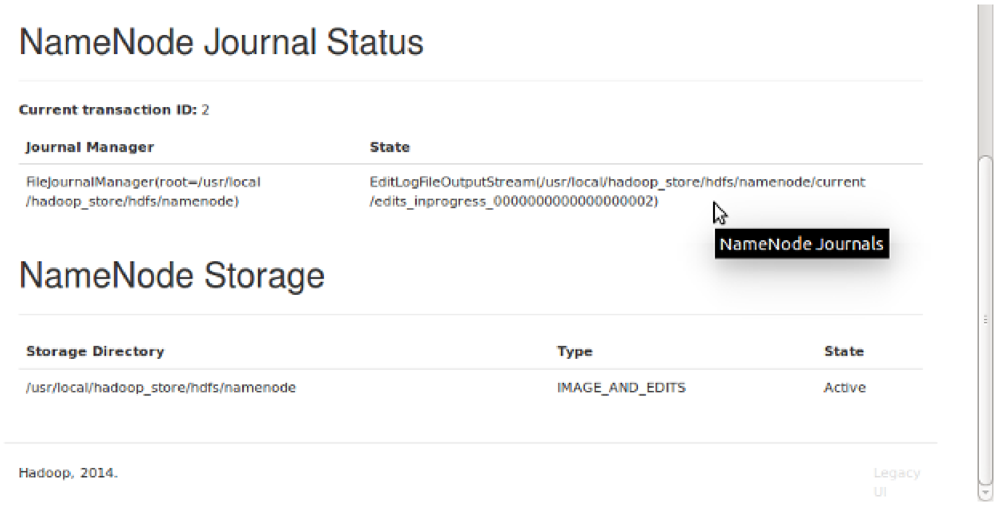
Non Heap Memory used 28.34 MB of 29.94 MB Commited Non Heap Memory. Max Non Heap Memory is 214 MB.

| | |
|---|---|
| **Configured Capacity:** | 454.29 GB |
| **DFS Used:** | 24 KB |
| **Non DFS Used:** | 125.8 GB |
| **DFS Remaining:** | 328.49 GB |
| **DFS Used%:** | 0% |
| **DFS Remaining%:** | 72.31% |
| **Block Pool Used:** | 24 KB |
| **Block Pool Used%:** | 0% |
| **DataNodes usages% (Min/Median/Max/stdDev):** | 0.00% / 0.00% / 0.00% / 0.00% |
| **Live Nodes** | 1 (Decommissioned: 0) |
| **Dead Nodes** | 0 (Decommissioned: 0) |
| **Decommissioning Nodes** | 0 |
| **Number of Under-Replicated Blocks** | 0 |
| **Number of Blocks Pending Deletion** | 0 |


Namenode information - Mozilla Firefox

http://localhost:50070/dfshealth.html#tab-overview

| | |
|---|---|
| **DataNodes usages% (Min/Median/Max/stdDev):** | 0.00% / 0.00% / 0.00% / 0.00% |
| Live Nodes | 1 (Decommissioned: 0) |
| Dead Nodes | 0 (Decommissioned: 0) |
| Decommissioning Nodes | 0 |
| **Number of Under-Replicated Blocks** | 0 |
| **Number of Blocks Pending Deletion** | 0 |


## NameNode Journal Status

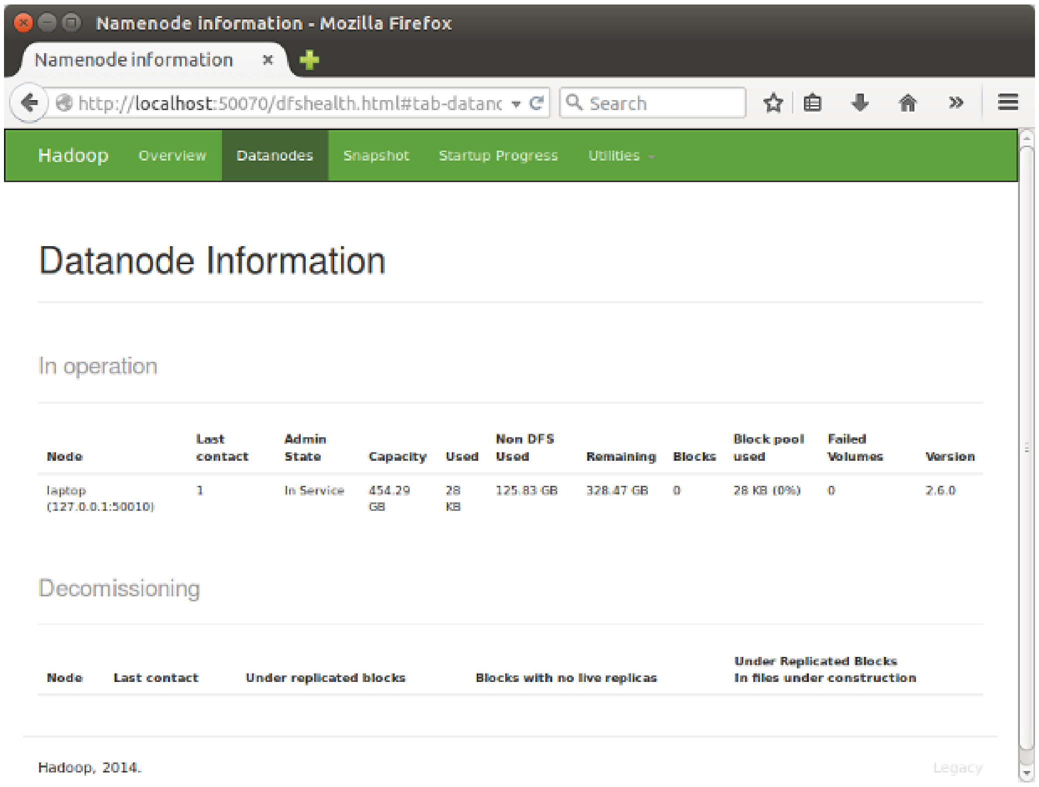
**Current transaction ID:** 2

| Journal Manager | State |
|---|---|
| FileJournalManager(root=/usr/local /hadoop_store/hdfs/namenode) | EditLogFileOutputStream(/usr/local/hadoop_store/hdfs/namenode/current /edits_inprogress_0000000000000000002) |

## NameNode Storage

| Storage Directory | Type | State |
|---|---|---|
| /usr/local/hadoop_store/hdfs/namenode | IMAGE_AND_EDITS | Active |

Hadoop, 2014.


Namenode information - Mozilla Firefox
Namenode information
http://localhost:50070/dfshealth.html#tab-datanc
Search
Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities

Datanode Information

In operation

| Node | Last contact | Admin State | Capacity | Used | Non DFS Used | Remaining | Blocks | Block pool used | Failed Volumes | Version |
|---|---|---|---|---|---|---|---|---|---|---|
| laptop (127.0.0.1:50010) | 1 | In Service | 454.29 GB | 28 KB | 125.83 GB | 328.47 GB | 0 | 28 KB (0%) | 0 | 2.6.0 |

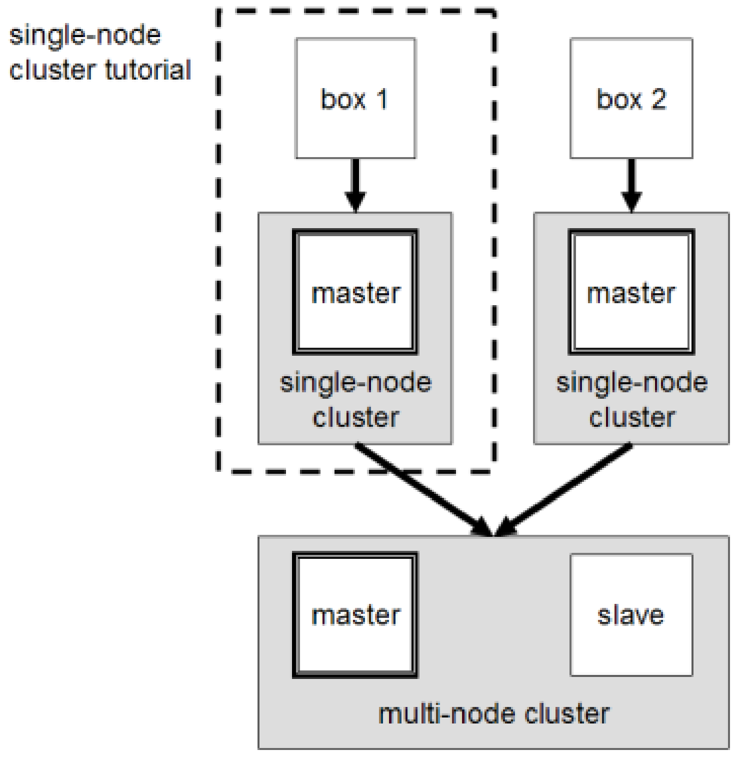
Decomissioning

| Node | Last contact | Under replicated blocks | Blocks with no live replicas | Under Replicated Blocks In files under construction |
|---|---|---|---|---|

Hadoop, 2014. Legacy

This Completes the Hadoop single node installation.

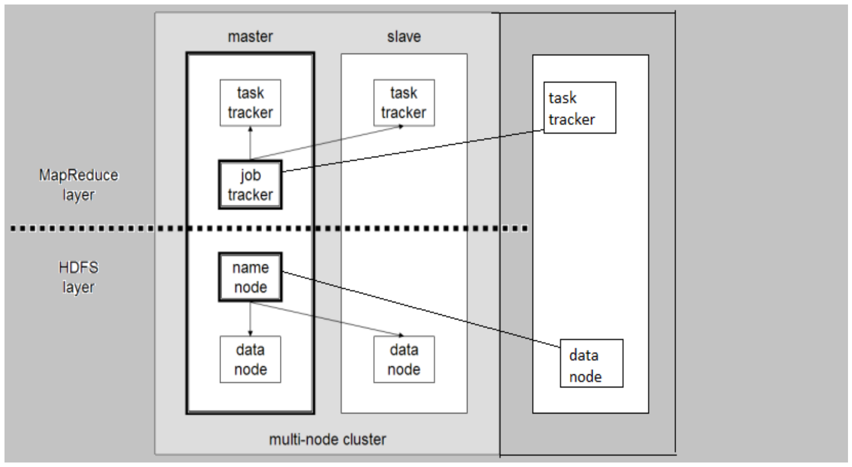
**HADOOP MULTI-NODE INSTALLATION:** The best way to do this is to install, configure and test a “local” Hadoop setup(Single-node) and in a second step to “merge” these three single-node clusters into one multi-node cluster in which one will become the designated master (but also act as a slave with regard to data storage and processing), and the other two will become slaves. It’s much easier to track down any problems you might encounter due to the reduced complexity of doing a single-node cluster setup first on each machine.



To cofigure a Multi-node cluster the following changes are to be done:

- **Networking:** both machines must be able to reach each other over the network. The easiest is to put both machines in the same network with regard to hardware and software configuration, for example connect both machines via a single hub or switch and configure the network interfaces to use a common network such as 172.16.25.X
  For eg: we will assign the IP address 192.16.25.11 to the master machine and 192.16.25.12 to the slave-1 machine and 172.16.25.13 to slave-2 machine.
  Now Update /etc/hosts on all machines with the following lines:
  ```
  192.16.25.11    master
  192.16.25.12    slave1
  192.16.25.13    slave2
  ```

- **Cluster Overview (aka the goal):** The master node will also act as a slave because we only have three machines available in our cluster but still want to spread data storage and processing to multiple machines.

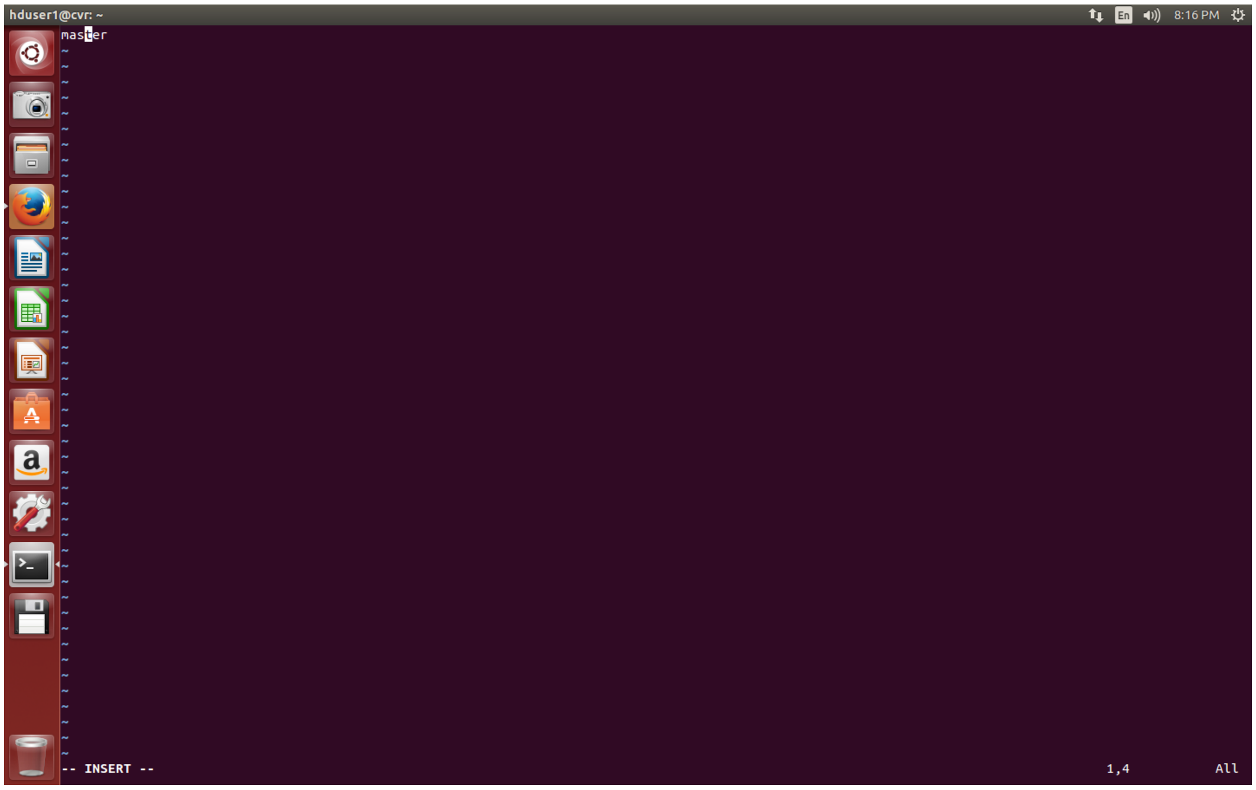


The master node will run the “master” daemons for each layer: NameNode for the HDFS storage layer, and JobTracker for the MapReduce processing layer. All machines will run the “slave” daemons: DataNode for the HDFS layer, and TaskTracker for MapReduce processing layer. Basically, the “master” daemons are responsible for coordination and management of the “slave” daemons while the latter will do the actual data storage and data processing work.

- **Configuration:**

1) **conf/masters (master only):** Despite its name, the conf/masters file defines on which machines Hadoop will start secondary NameNodes in our multi-node cluster. In our case, this is just the master machine. The primary NameNode and the JobTracker will always be the machines on which you run the bin/start-dfs.sh and bin/start-mapred.sh scripts, respectively (the primary NameNode and the JobTracker will be started on the same machine if you run bin/start-all.sh).

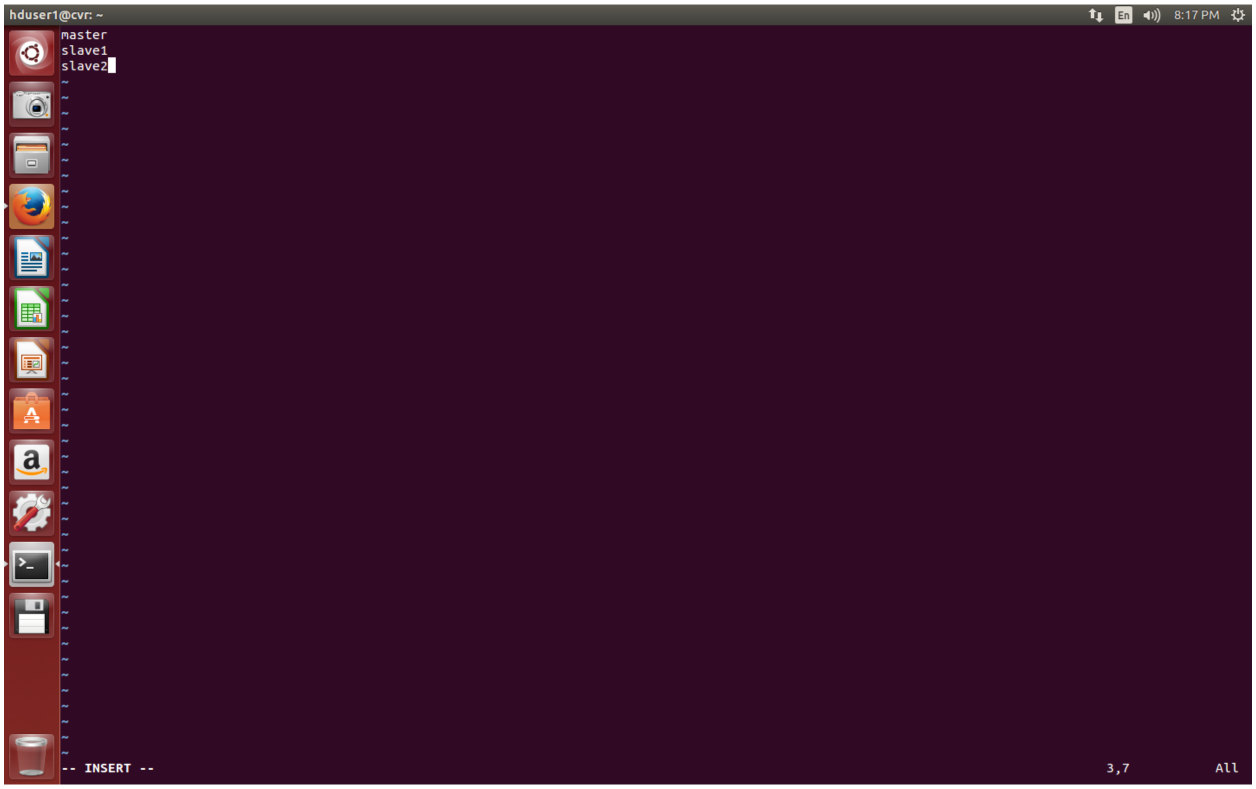
   On master, update conf/masters that it looks like this:



```
master
~
-- INSERT --                                                    1,4           All
```

2) **conf/slaves (all nodes):** The conf/slaves file lists the hosts, one per line, where the Hadoop slave daemons (DataNodes and TaskTrackers) will be run. We want both the master box and the slave box to act as Hadoop slaves because we want both of them to store and process data.

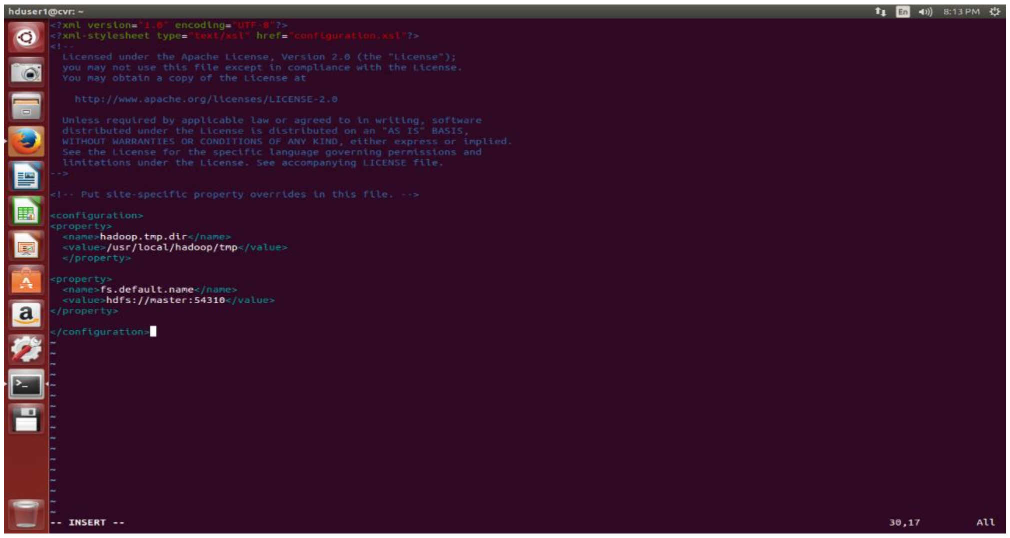
On master, update conf/slaves that it looks like this:



```
master
slave1
slave2
-- INSERT --                                              3,7          All
```

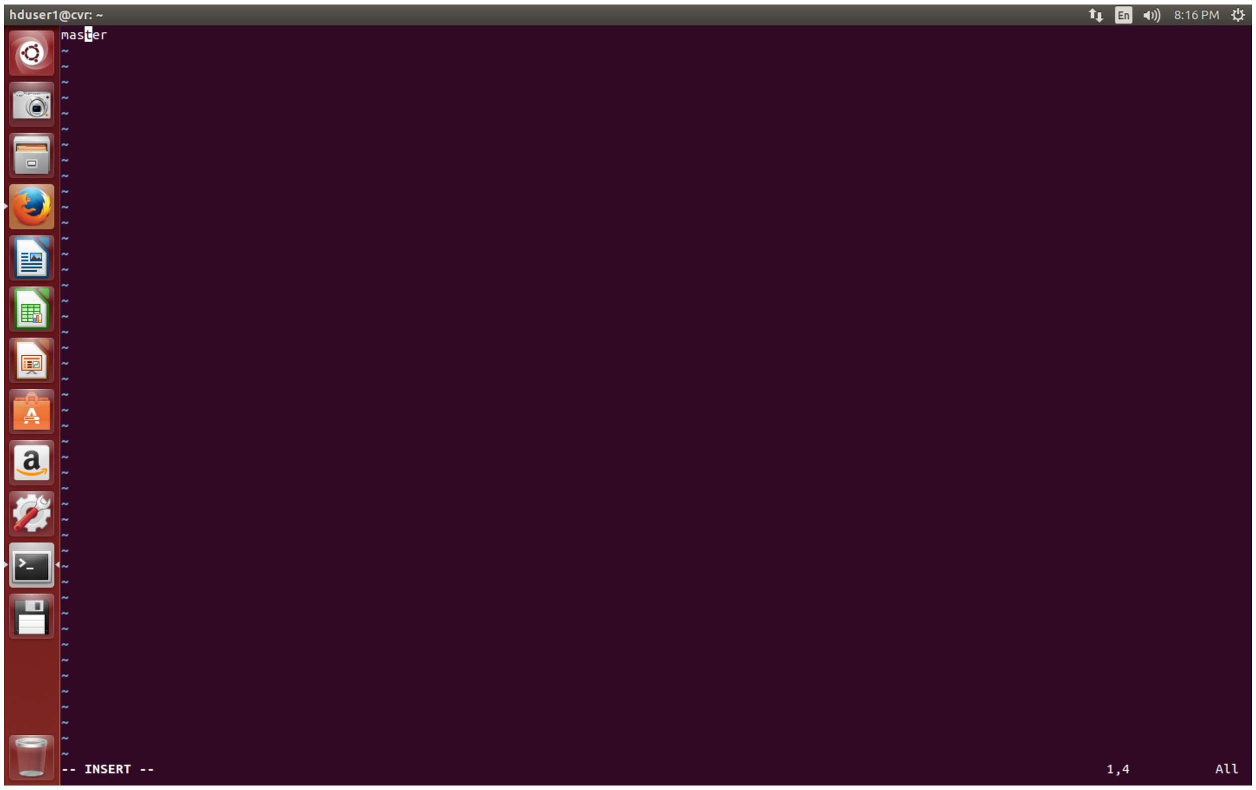
3) **conf/*-site.xml (all nodes):** You must change the configuration files conf/core site.xml, conf/mapred-site.xml and conf/hdfs-site.xml on ALL machines as follows.

First, we have to change the fs.default.name parameter (in conf/core-site.xml), which specifies the NameNode (the HDFS master) host and port. In our case, this is the master machine.



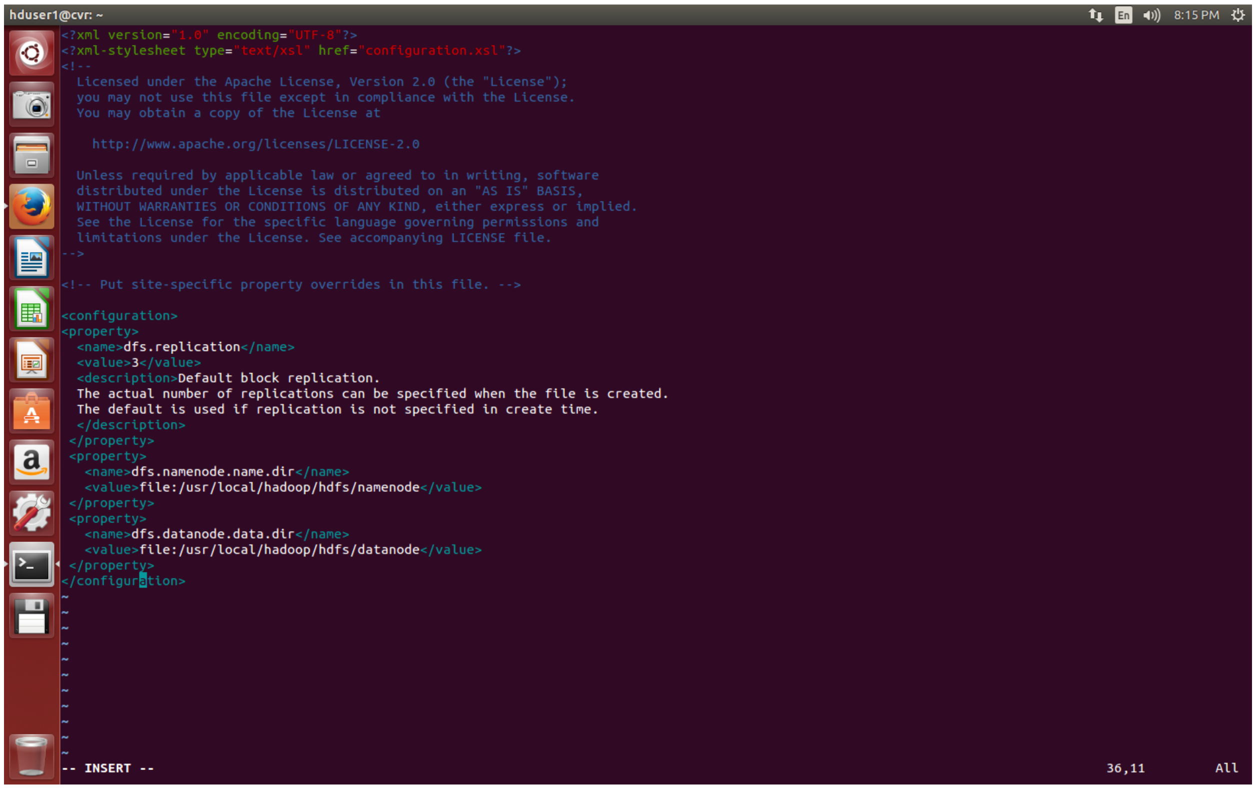
```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/usr/local/hadoop/tmp</value>
  </property>

<property>
  <name>fs.default.name</name>
  <value>hdfs://master:54310</value>
</property>

</configuration>
-- INSERT --                                              30,17        All
```

Second, we have to change the mapred.job.tracker parameter (in conf/mapred site.xml), which specifies the JobTracker (MapReduce master) host and port. Again, this is the master in our case.



```
master
~
-- INSERT --                                                                1,4           All
```

Third, we change the dfs.replication parameter (in conf/hdfs-site.xml) which specifies the default block replication. It defines how many machines a single file should be replicated to before it becomes available. The default value of dfs.replication is 3.



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>dfs.replication</name>
  <value>3</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:/usr/local/hadoop/hdfs/namenode</value>
 </property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/usr/local/hadoop/hdfs/datanode</value>
 </property>
</configuration>
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                                                         36,11         All
```

4) **Formatting the HDFS filesystem via the NameNode**: Before we start our new multi-node cluster, we must format Hadoop's distributed filesystem (HDFS) via the NameNode. You need to do this the first time you set up an Hadoop cluster. To format the filesystem (which simply initializes the directory specified by the dfs.name.dir variable on the NameNode), run the command – "**hadoop namenode -format**".

5) **Starting the multi-node cluster:** Starting the cluster is performed in two steps.
   We begin with starting the HDFS daemons: the NameNode daemon is started on master, and DataNode daemons are started on all slaves (here: master and slave).
   Then we start the MapReduce daemons: the JobTracker is started on master, and TaskTracker daemons are started on all slaves (here: master and slave).
   (OR) we can simply start it with "**start-all.sh**"

6) **Stopping the multi-node cluster**: Like starting the cluster, stopping it is done in two steps. The workflow however is the opposite of starting.

We begin with stopping the MapReduce daemons: the JobTracker is stopped on master, and TaskTracker daemons are stopped on all slaves (here: master and slave).

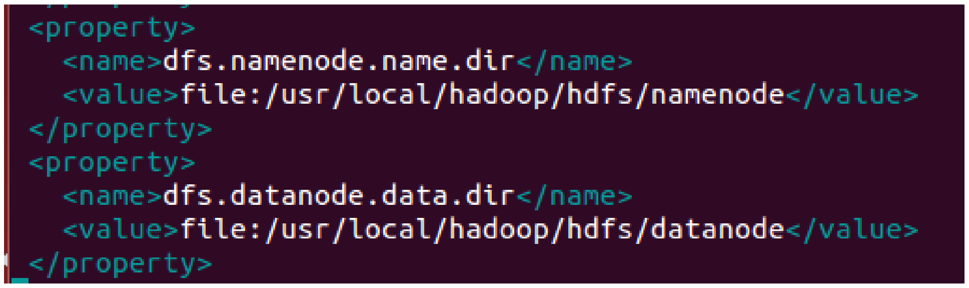
Then we stop the HDFS daemons: the NameNode daemon is stopped on master, and DataNode daemons are stopped on all slaves (here: master and slave).

## EXPECTED ERRORS AND SOLUTIONS:

1) Experiencing problem while installing java using apt-get command: For this we need to goto Settings→Details→software updates→Under sources check all. This will enable Ubuntu to download third party apps via "**apt-get**" command.

2) Not able to communicate using ssh localhost: This generally happens if "Ipv6" is not disabled. Hadoop requires disabling of Ipv6 for communication. This can be done by the command – "**export HADOOP_OPTS=-Djava.net.preferIPv4Stack=true**"

3) Problem with Data node: If data node does not show up then there may be problem with the path specified in mapre-site.xml. Here we need to specify the absolute path in the Value.



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

4) If still problem exists then the temporary directory should be cleaned. This can be done by the following command – "**rm -Rf /usr/local/hadoop/tmp/***". This command deletes the temporary file in the directory.

5) If still problem persists with datanode then it may be due to wrong NanmeSpace ID's od data and name nodes. The should be in same parent directory.

6) Problem with Namenode: This problem occurs with wrong path value of Namenode. The correct Absolute path should be given in hdfs-site.xml.

7) "Jps" Does not display anything: This will occur if the java path which is set in ~/.bashrc is not same as the one specifies in hadoop. This can be changed via "java-alternatives" command.

8) Unable to establish connection between master and slave in multimode cluster: This will occur if the IP addresses mentioned in etc/hosts is not valid. Even etc/hostname should be changed.

9) If still problem shows up, then there may be a problem in slaves and master folders in "/usr/local/hadoop/etc/hadoop/slaves" OR "/usr/local/hadoop/etc/hadoop/master" .

This completes Hadoop multinode installation.

# R-PROGRAMMING

## INTRODUCTION

R is a language and environment for statistical computing and graphics. It is GNU project  which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R.

R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, …) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity.

One of R’s strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control.

R is available as Free Software under the terms of the Free software's GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS.

## THE R ENVIRONMENT

R is an integrated suite of software facilities for data manipulation, calculation and graphical display. It includes

- an effective data handling and storage facility,
- a suite of operators for calculations on arrays, in particular matrices,
- a large, coherent, integrated collection of intermediate tools for data analysis,
- graphical facilities for data analysis and display either on-screen or on hardcopy, and
- a well-developed, simple and effective programming language which includes conditionals, loops, user-defined recursive functions and input and output facilities.

  The term "environment" is intended to characterize it as a fully planned and coherent system, rather than an incremental accretion of very specific and inflexible tools, as is frequently the case with other data analysis software.

  R, like S, is designed around a true computer language, and it allows users to add additional functionality by defining new functions. Much of the system is itself written in the R dialect of which makes it easy for users to follow the algorithmic choices made. For computationally-intensive tasks, C, C++ and Fortran code can be linked and called at run time. Advanced users can write C code to manipulate R objects directly.

  Many users think of R as a statistics system. We prefer to think of it of an environment within which statistical techniques are implemented. R can be extended (easily) via *packages*. There are about eight packages supplied with the R distribution and many more are available through the CRAN family of Internet sites covering a very wide range of modern statistics.

  R has its own LaTeX-like documentation format, which is used to supply comprehensive documentation, both on-line in a number of formats and in hardcopy.

## APPLICATIONS OF R PROGRAMMING

R applications span the universe from theoretical computational statistics and the hard sciences such as astronomy, chemistry and genomics to practical applications in business, drug development, finance, health care, marketing, medicine and much more. Because R has nearly 5,000 packages (libraries of functions) many of which are dedicated to specific applications you don't have to be an R genius to begin developing your own applications. All it takes to get started is some modest experience with R, some examples to emulate and the right libraries of R functions.

Not only is R free, but it's also open source. That means anyone can examine the source code to see exactly what it's doing. This also means that you, or anyone, can fix bugs and/or add features, rather than waiting for the vendor to find/fix the bug and/or add the feature–at their discretion–in a future release.

R allows you to integrate with other languages (C/C++, Java, Python) and enables you to interact with many data sources: ODBC-compliant databases (Excel, Access) and other statistical packages (SAS, Stata,SPSS, Minitab).

Explicit parallelism is straightforward in R (see the High Performance Computing Task ): several packages allow you to take advantage of multiple cores, either on a single machine or across a network. You can also build R with custom blas

**R-FOUNDATION**

The R Foundation is a not for profit organization working in the public interest. It has been founded by the members of the R Development Core Team in order to

- Provide support for the R project and other innovations in statistical computing. We believe that R has become a mature and valuable tool and we would like to ensure its continued development and the development of future innovations in software for statistical and computational research.
- Provide a reference point for individuals, instititutions or commercial enterprises that want to support or interact with the R development community.
- Hold and administer the copyright of R software and documentation.

R is an official part of the Free software foundation`'s GNU project, and the R Foundation has similar goals to other open source software foundations like the Apache foundation or the gnome foundation

Among the goals of the R Foundation are the support of continued development of R, the exploration of new methodology, teaching and training of statistical computing and the organization of meetings and conferences with a statistical computing orientation. We hope to attract sufficient funding to make these goals realities.

R supports procedural programming with functions and, for some functions, object-oriented programming with generic functions A generic function acts differently depending on the classes of arguments passed to it. In other words, the generic function dispatches the function (method) specific to that class of object. For example, R has a generic print function that can print almost every class of object in

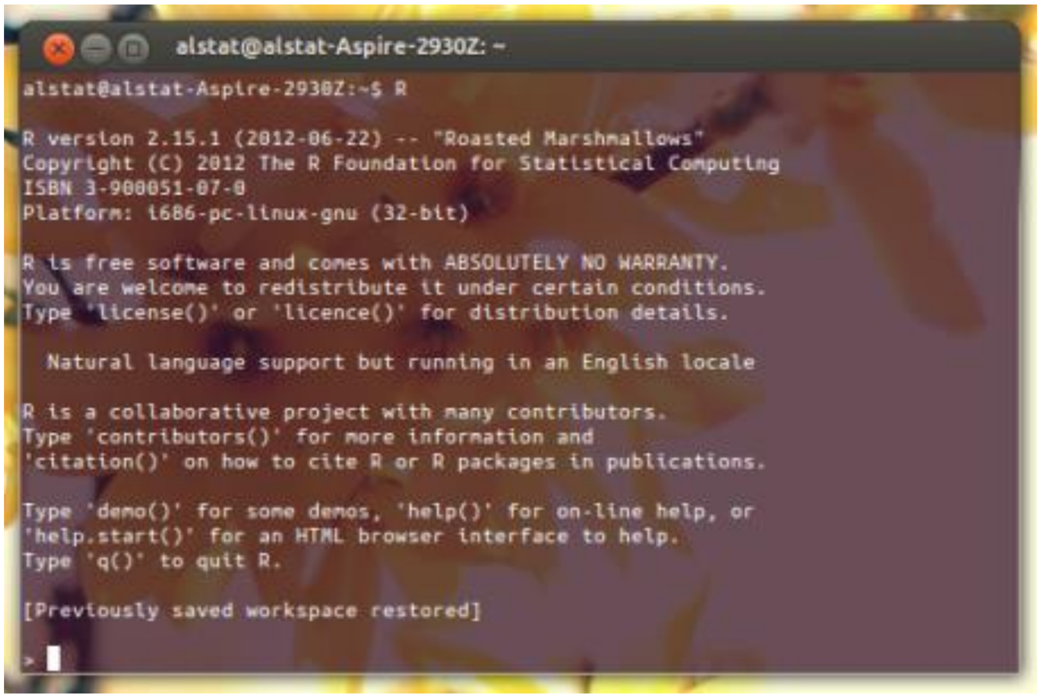
**DOWNLOAD AND INSTALL R IN UBUNTU**

There are two ways to install R in Ubuntu. One is through the terminal, and the other is through the Ubuntu Software Center.

Through **Terminal**

- Press Ctrl+Alt+T to open **Terminal**
- Then execute **sudo apt-get update**
- After that, **sudo apt-get install r-base**

To run R statistical package, execute **R** in the Terminal



```
alstat@alstat-Aspire-2930Z:~$ R

R version 2.15.1 (2012-06-22) -- "Roasted Marshmallows"
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i686-pc-linux-gnu (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> 
```

Through **Ubuntu Software Center**

- Open **Ubuntu Software Center**
- Search for **r-base**
- And click **Install**

- Then run R by executing **R** in the **Terminal**

Working through **Terminal** would be inconvenient, we suggest to download a user-friendly interface for R. For Ubuntu, we recommend using RStudio IDE or RKWard KDE.

## I. INSTALLATION OF R STUDIO

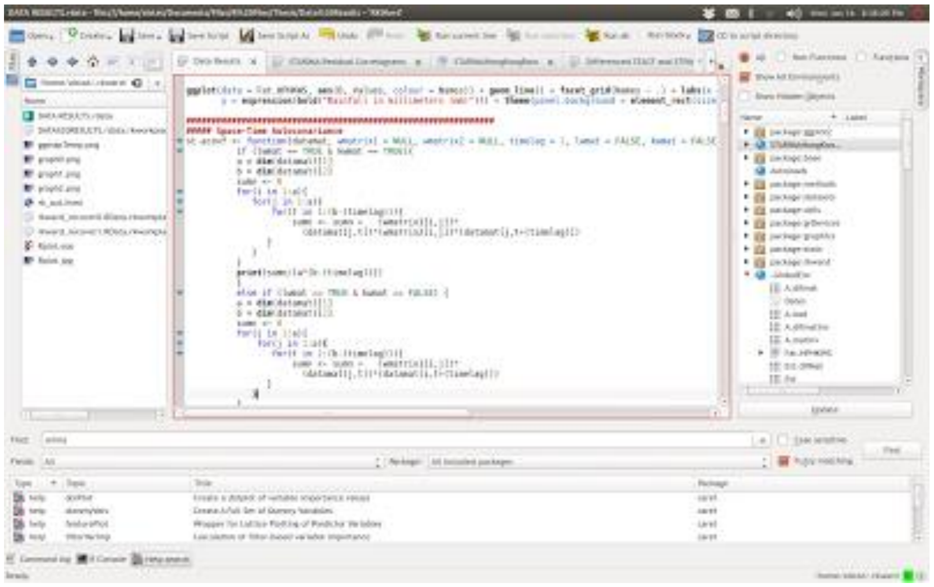
To install RStudio IDE, do the following:

- Go to RStudio IDE download page
- Click Download RStudio Desktop
- Then click for the download link recommended for your system
- Run the downloaded file (double click the file) to start the setup wizard
- Click “Next” until “Finish”

To open RStudio IDE, search for RStudio in the Ubuntu dash.

To install RKWard KDE, do the following:

- Open **Ubuntu Software Center**
- Search for **RKWard**
- Then click **Install**



To open RKWard KDE, search for RKWard in the Ubuntu dash and hit Enter.

**FUNCTIONS USED IN R PROGRAMMING**

type=”p” Plot individual points (the default)

type=”l” Plot lines type=”b” Plot points connected by lines (both)

type=”o” Plot points overlaid by lines

type=”h” Plot vertical lines from points to the zero axis (high-density)

type=”n” No plotting at all.

However axes are still drawn (by default) and the coordinate system is set up according to the data. Ideal for creating plots with subsequent low-level graphics functions

xlab=string

ylab=string Axis labels for the x and y axes. Use these arguments to change the default labels, usually the names of the objects used in the call to the high-level plotting function.

main=string Figure title, placed at the top of the plot in a large font.

sub=string Sub-title, placed just below the x-axis in a smaller font.

Plotting functions in R can be divided into three basic groups: High-level plotting functions create a new plot on the graphics device, possibly with axes, labels, titles and so on. Low-level plotting functions add more information to an existing plot, such as extra points, lines and labels. Interactive graphics functions allow you interactively add information to, or extract information from, an existing plot, using a pointing device such as a mouse. In addition, R maintains a list of graphical parameters that affect the result of various plot functions.

The plot() function. The Generic plot() function can deal with the task of plotting several types of R

## R COLOURS

Thus far, we have frequently used numbers in plot to refer to a simple set of colors. There are 8 colors where 0:8 are white, black, red, green, blue, cyan, magenta, yellow and grey. If you provide a number greater than 8, the colors are recycled. Therefore for plots where other or greater numbers of colors are required, we need to access a larger palette of colors.

A very useful RColorBrewer http://colorbrewer.org. This package will generate a ramp color to provide color palattes that are sequential, diverging, and qualitative ramped, for example:

• Sequential palettes are suited to ordered data that progress from low to high. Lightness steps dominate the look of these schemes, with light colors for low data values to dark colors for high data values.

• Diverging palettes put equal emphasis on mid-range critical values and extremes at both ends of the data range. The critical class or break in the middle of the legend is emphasized with light colors and low and high extremes are emphasized with dark colors that have contrasting hues.

• Qualitative palettes do not imply magnitude differences between legend classes, and hues are used to create the primary visual differences between classes. Qualitative schemes are best suited to representing nominal or categorical data.

**PLOTTING OF POINTS IN R STUDIO**

First, download the file and load it into your favorite text editor. Replace ss+ with t to create two tab delimited columns. I think this is probably easier than trying to get R to read data separated by at least 2 spaces, as the source file seems to be. Now, load your data into R.

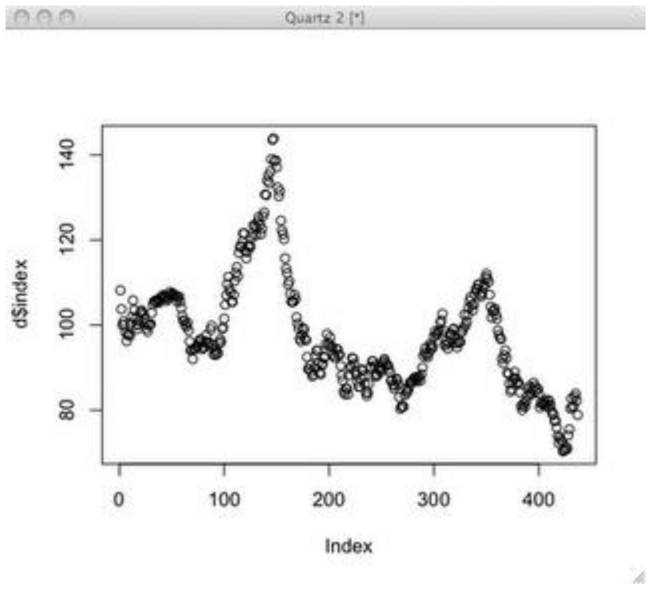
```
d = read.table('dollar_vs_major_currencies_index.txt', header=F, sep="t", col.names=c("month", "index"))

dim(d)

[1] 437   2

head(d)

  month    index

1 JAN 1973 108.1883

2 FEB 1973 103.7461

3 MAR 1973 100.0000

4 APR 973 100.8251

5 MAY 1973 100.0602

6 JUN 1973  98.2137
```

R will show you the structure of an object using the str() command:

```
str(d)

'data.frame': 437 obs. of  2 variables:

 $ month: Factor w/ 437 levels "APR 1973","APR 1974",..: 147 110 256 1 293 220 184 38 402
366 ...

 $ index: num  108 104 100 101 100 ...
```

So far so good. R is all about stats, so why not do this?

```
summary(d$index)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.

  70.32   87.93   95.89   97.48  105.40  143.90
```

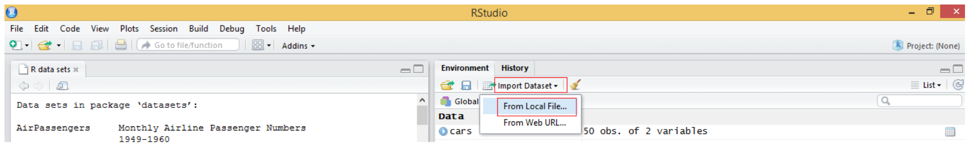


**VISUALISTION OF CLUSTERS IN RSTUDIO**

Here is the visualisation of a population dataset containing the population of different countries during several years

STEP 1

First we need to import the dataset into R studio

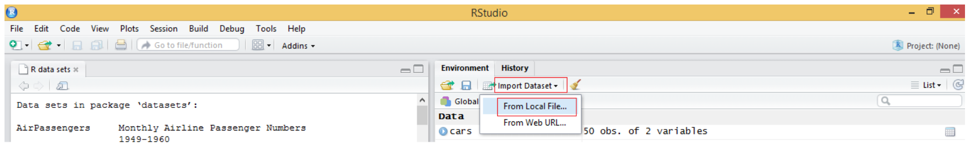


```
Data sets in package 'datasets':

AirPassengers       Monthly Airline Passenger Numbers
                    1949-1960
```

```
Data
cars                50 obs. of 2 variables
```

STEP 2

Display the dataset using view function in rstudio

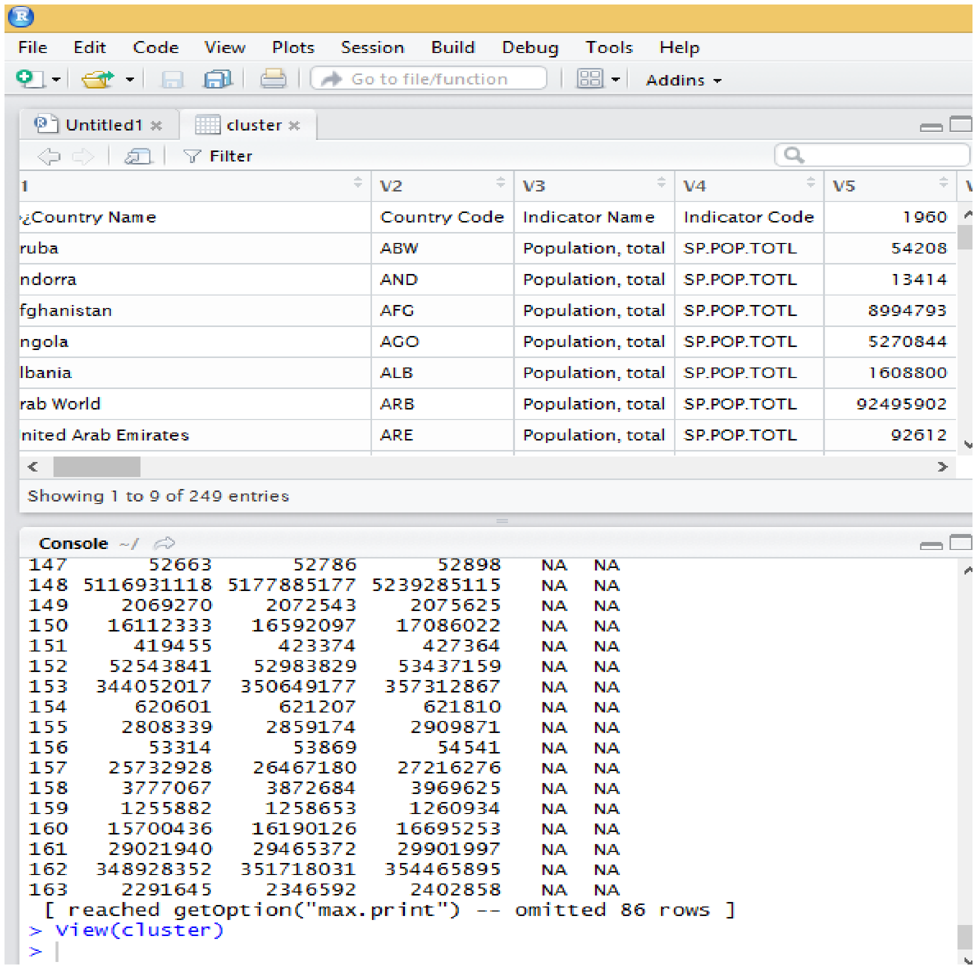


```
Data sets in package 'datasets':

AirPassengers       Monthly Airline Passenger Numbers
                    1949-1960
```

```
Data
cars                50 obs. of 2 variables
```

STEP 3

Display the number of observations and variables using R-functions



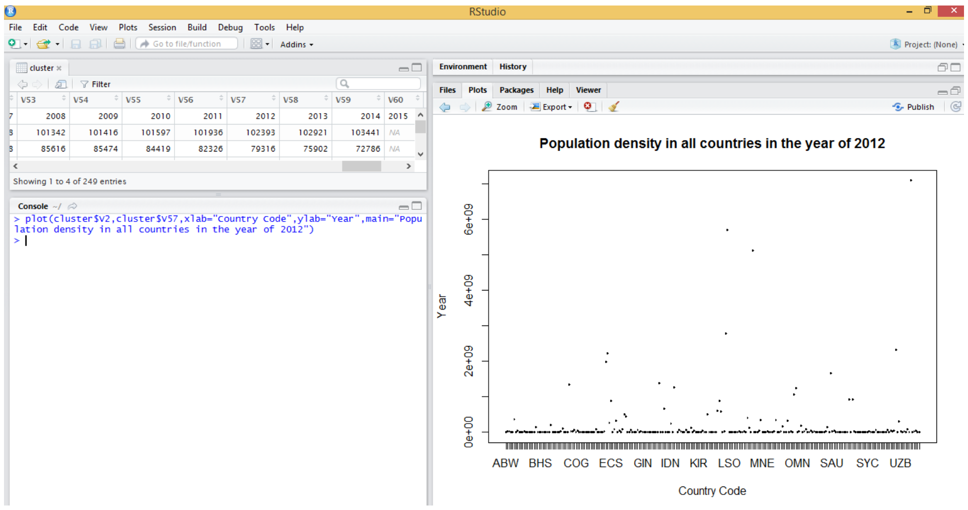
| 1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|
| ¿Country Name | Country Code | Indicator Name | Indicator Code | 1960 |
| ruba | ABW | Population, total | SP.POP.TOTL | 54208 |
| ndorra | AND | Population, total | SP.POP.TOTL | 13414 |
| fghanistan | AFG | Population, total | SP.POP.TOTL | 8994793 |
| ngola | AGO | Population, total | SP.POP.TOTL | 5270844 |
| lbania | ALB | Population, total | SP.POP.TOTL | 1608800 |
| rab World | ARB | Population, total | SP.POP.TOTL | 92495902 |
| nited Arab Emirates | ARE | Population, total | SP.POP.TOTL | 92612 |

Showing 1 to 9 of 249 entries

```
147       52663      52786      52898   NA  NA
148 5116931118 5177885177 5239285115   NA  NA
149     2069270    2072543    2075625   NA  NA
150    16112333   16592097   17086022   NA  NA
151      419455     423374     427364   NA  NA
152    52543841   52983829   53437159   NA  NA
153   344052017  350649177  357312867   NA  NA
154      620601     621207     621810   NA  NA
155     2808339    2859174    2909871   NA  NA
156       53314      53869      54541   NA  NA
157    25732928   26467180   27216276   NA  NA
158     3777067    3872684    3969625   NA  NA
159     1255882    1258653    1260934   NA  NA
160    15700436   16190126   16695253   NA  NA
161    29021940   29465372   29901997   NA  NA
162   348928352  351718031  354465895   NA  NA
163     2291645    2346592    2402858   NA  NA
 [ reached getOption("max.print") -- omitted 86 rows ]
> View(cluster)
> 
```

STEP 4

Plot the points using the plot function


RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function
Addins
Project: (None)
cluster
Filter
Environment History
Files Plots Packages Help Viewer
Zoom Export
Publish
Population density in all countries in the year of 2012
Year
0e+00
2e+09
4e+09
6e+09
ABW BHS COG ECS GIN IDN KIR LSO MNE OMN SAU SYC UZB
Country Code

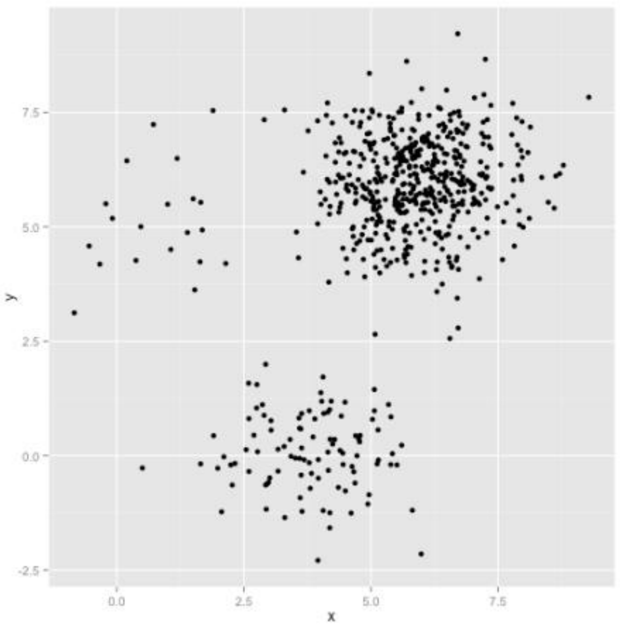
| V53 | V54 | V55 | V56 | V57 | V58 | V59 | V60 |
|---|---|---|---|---|---|---|---|
| 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | 2014 | 2015 |
| 101342 | 101416 | 101597 | 101936 | 102393 | 102921 | 103441 | NA |
| 85616 | 85474 | 84419 | 82326 | 79316 | 75902 | 72786 | NA |

Showing 1 to 4 of 249 entries

```
> plot(cluster$V2,cluster$V57,xlab="Country Code",ylab="Year",main="Popu
lation density in all countries in the year of 2012")
> |
```

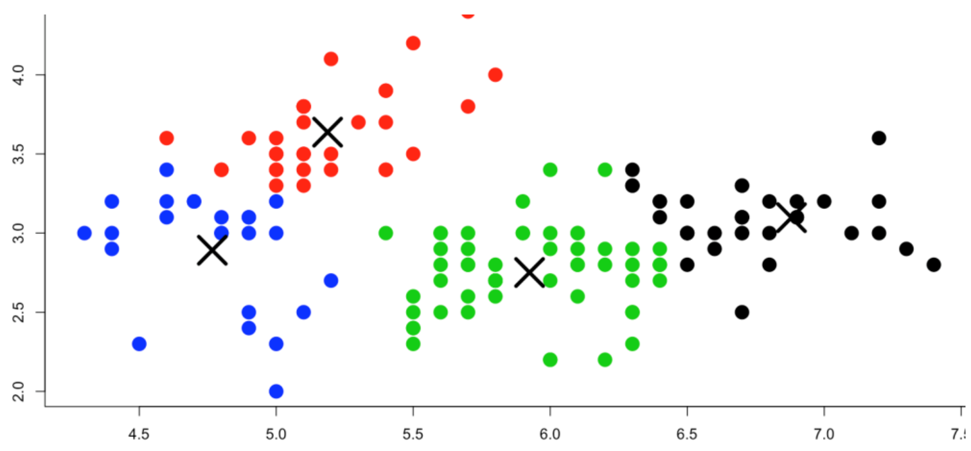
STEP 5

Visualize clusters in plots tab



STEP 6

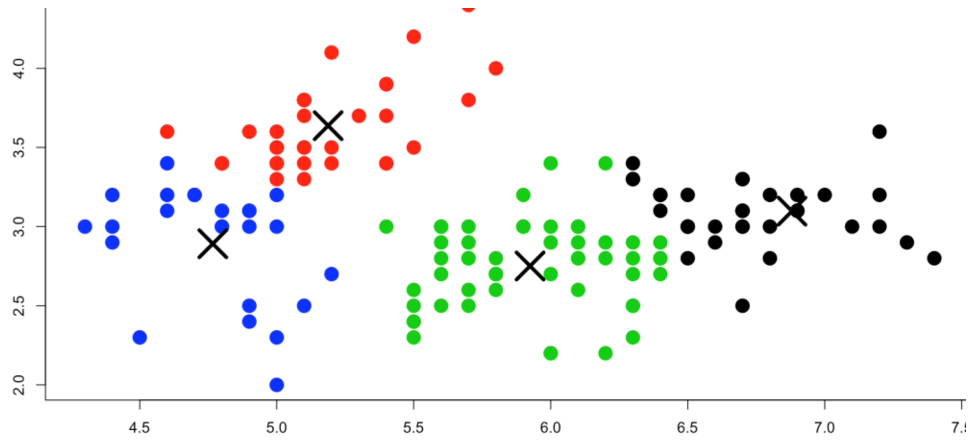
Add colours to the clusters using functions



## RESULTS

We get the result from Map reduce implementation. Then the output of Map Reduce is sent to R Hadoop for output visualization.

The Output of the Map Reduce is the set of centroids we get after the last pass of Map Reduce. This output is sent as input to the R Hadoop and with the use of different functions in R we get the Graph of the population dataset which displays clusters, each of which represent the countries that have similar range of population growth.



X-axis- country code

Y-axis- years

The threshold radius of clusters is set to get the left out points. These points are our required outliers.

# CONCLUSION

In this project, we presented a method by which we have implemented K-Means clustering algorithm on a parallel framework of Map Reduce. This idea mainly focuses on distributing the core computational part of the algorithm on multiple nodes. The incorporation of the clustering algorithm along with the Hadoop-MapReduce framework has demonstrated significant performance gain and speed up in our experiment. The algorithm has been tested with different number of dimensions, data points and clusters.

Future work will include experimenting with the other variations of k-means such as automatic centroid selection and also load balancing among nodes which performs the computation.

## REFERENCES

1. *Hornik, Kurt (November 26, 2015). "R FAQ". The Comprehensive R Archive Network. 2.1 What is R?. Retrieved 2015-12-06.*

2. *Ihaka, Ross (1998). R : Past and Future History(PDF) (Technical report). Statistics Department, The University of Auckland, Auckland, New Zealand.*

3. R language and environment

   · *Hornik, Kurt (November 26, 2015). "R FAQ". The Comprehensive R Archive Network. 2.1 What is R?. Retrieved 2015-12-06.*

   R Foundation

   · *Hornik, Kurt (November 26, 2015). "R FAQ". The Comprehensive R Archive Network. 2.13 What is the R Foundation?. Retrieved 2015-12-06.*

   The R Core Team asks authors who use R in their data analysis to cite the software using:

   · R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

4. widely used

   · *Fox, John and Andersen, Robert (January 2005)."Using the R Statistical Computing Environment to Teach Social Statistics Courses" (PDF). Department of Sociology, McMaster University. Retrieved 2006-08-03.*

   · *Vance, Ashlee (2009-01-06). "Data Analysts Captivated by R's Power". New York Times. Retrieved 2009-04-28. R is also the name of a popular programming language used by a growing number of data analysts inside corporations and academia. It is becoming their lingua franca...*

5. *Vance, Ashlee (2009-01-06). "Data Analysts Captivated by R's Power". New York Times. Retrieved 2009-04-28.R is also the name of a popular programming language*

*used by a growing number of data analysts inside corporations and academia. It is becoming their lingua franca...*

6. R's popularity

   · David Smith (2012); *R Tops Data Mining Software Poll*, Java Developers Journal, May 31, 2012.

   · Karl Rexer, Heather Allen, & Paul Gearan (2011);*2011 Data Miner Survey Summary*, presented at Predictive Analytics World, Oct. 2011.

   · *Robert A. Muenchen (2012). "The Popularity of Data Analysis Software".*

   · *Tippmann, Sylvia (29 December 2014). "Programming tools: Adventures with R". Nature (517): 109–110.doi:10.1038/517109a.*

7. ^ *Morandat, Frances; Hill, Brandon (2012). "Evaluating the design of the R language: objects and functions for data analysis" (PDF). ECOOP'12 Proceedings of the 26th European conference on Object-Oriented Programming.* External link in |journal= (help)

   *"R: What is R?". R-Project. Retrieved 7 February2016.*

8. *Gentleman, Robert (9 December 2006). "Individual Expertise profile of Robert Gentleman". Archived fro*

9. *"ValidR". Mongo Solutions. Retrieved April 15, 2016.*

10. Oracle Corporation's Big Data Appliance

   · Doug Henschen (2012); *Oracle Makes Big Data Appliance Move With Cloudera*, InformationWeek, January 10, 2012.

   · Jaikumar Vijayan (2012); *Oracle's Big Data Appliance brings focus to bundled approach*, ComputerWorld, January 11, 2012.

   · Timothy Prickett Morgan (2011); Oracle rolls its own NoSQL and Hadoop *Oracle rolls its own NoSQL and Hadoop*, The Register, October 3, 2011.