

BASIC UNDERSTANDING ON CLOUD COMPUTING

Introduction to Cloud Computing

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models. The essential characteristics are on-demand self-service, broadcast network access, resource pooling, rapid elasticity, measured service, etc.

Service Models

Software as a Service (SaaS): The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user specific application configuration settings.

Platform as a Service (PaaS): The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.³ The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

Infrastructure as a Service (IaaS): The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).

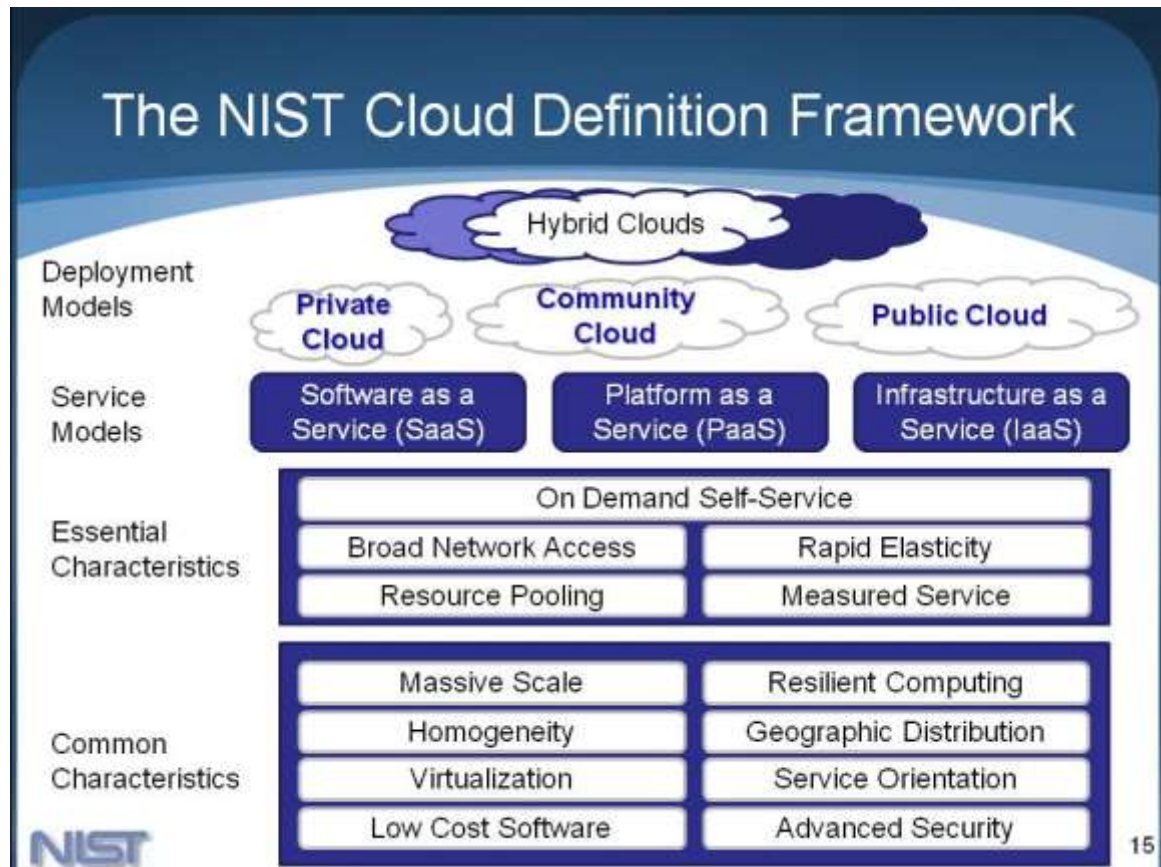
Deployment Models

Private cloud: The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.

Community cloud: The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.

Public cloud: The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider.

Hybrid cloud: The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).



NIST Cloud Definition Framework

EX. NO: 1

**DATE: CREATION AND MANAGEMENT OF VIRTUAL MACHINE IN
VIRTUALIZED ENVIRONMENT – VMWAREWORKSTATION**

AIM

- a) To install VMWare Workstation and configure a virtual machine.
- b) To create Virtual Clone.

THEORY

VIRTUALIZATION

Virtualization is creation of Virtual Machines which can emulate hardware in software or in other words it is the creation of virtual version of something such as a hardware platform, operating system, storage device, or network resources (from Wikipedia). Virtualization is achieved or created with the help of software and this particular software allows you to install any number of OS on your system without using the available hardware directly. When you are running an OS over the top of another on your machine the whole environment acts like a HOST and GUEST OS. The real operating system acts as a HOST and the OS run by virtualization software acts as a GUEST OS. The entire load balancing is actually done by the HOST operating system.

Types of Virtualization: There are mainly three types of virtualization.

- Full virtualization
- OS level virtualization
- Para virtualization

Full virtualization

As the name suggests everything in a system is virtualized which includes the processor, storage, networking components etc. Virtual Box, VMware are example of “Full Virtualization” solutions.

OS Level virtualization

In this type of virtualization only applications are run inside the software. In this case the application is given a platform to work. Isolation is created and the application is made to believe that it is the only thing running on the system.

Para virtualization

It's a semi-virtualized environment created for the guest OS. A modified guest OS is created using a hypervisor. “The intent of the modified interface is to reduce the portion of the guest's execution time spent performing operations which are substantially more difficult to run in a virtual environment compared to a non-virtualized environment. The Para virtualization provides specially defined ‘hooks’ to allow the guest(s) and host to request and acknowledge these tasks, which would otherwise be executed in the virtual domain (where execution performance is worse). A successful Para virtualized platform may allow the virtual machine monitor (VMM) to be simpler (by relocating execution of critical tasks from the virtual domain to

the host domain), and/or reduce the overall performance degradation of machine-execution inside the virtual-guest.”

Advantages of Virtualization

- One of the biggest advantages of virtualization is scalability i.e. the ability to expand. Whenever there is excessive load on some part of application in a server you can easily create a similar virtual environment on a different server and configure the setup.
- Hardware maintenance cost is reduced because you don't need many servers to install different applications.
- You can save a huge amount of energy by running one physical server instead of many and less power backup is required.
- You can get faster and safer backups by taking live snapshot while server is running.
- You will get centralized monitoring of your resources as virtualization provides easy way of connecting and maintaining your virtual servers

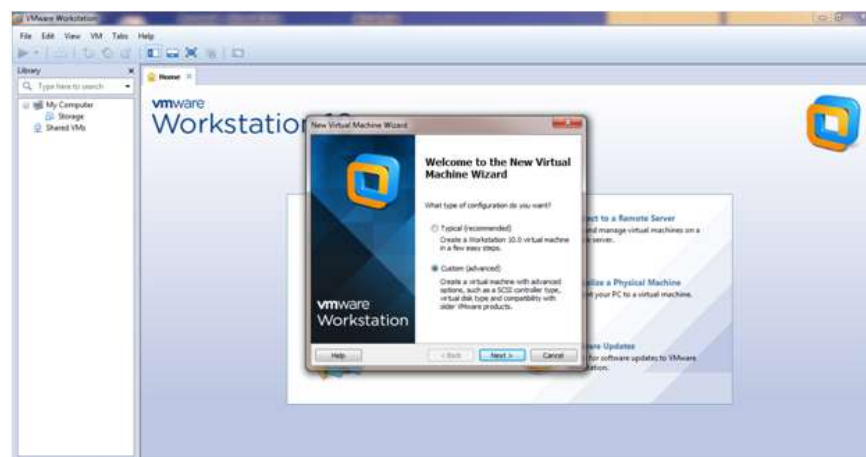
PRODEEDURE

a) To install VMWare Workstation and configure a virtual machine.

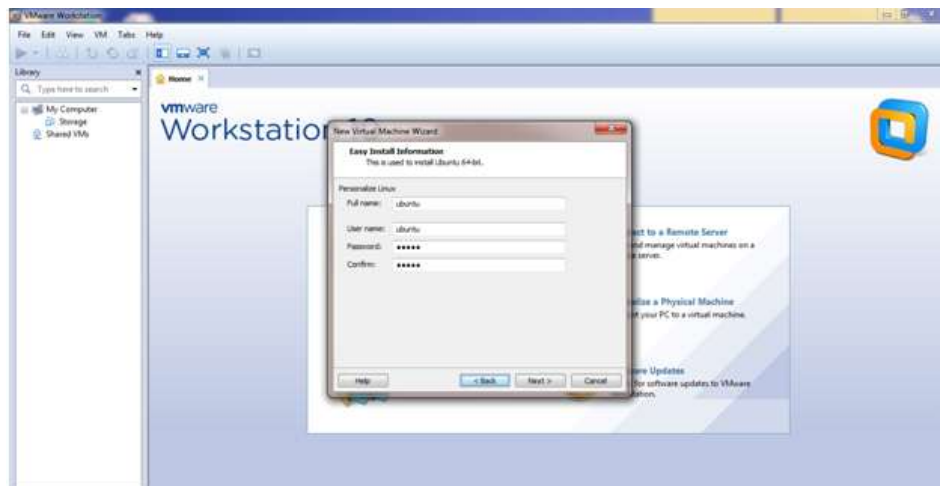
Step: 1 – Create new virtual machine



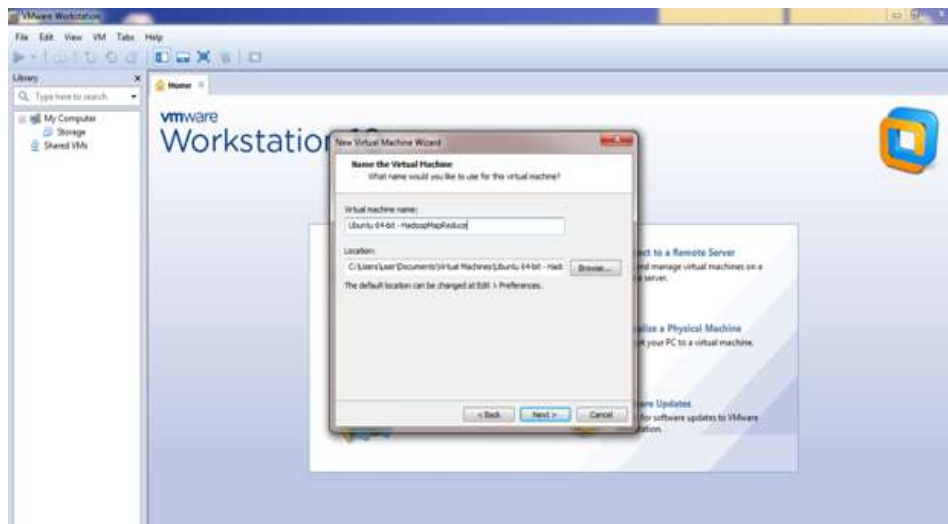
Step: 2 – Customize the set-up



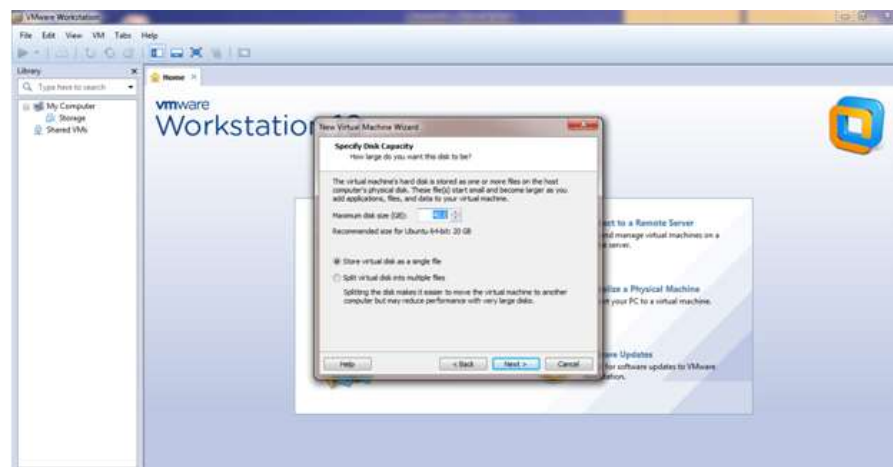
Step: 3 – Set Username and Password



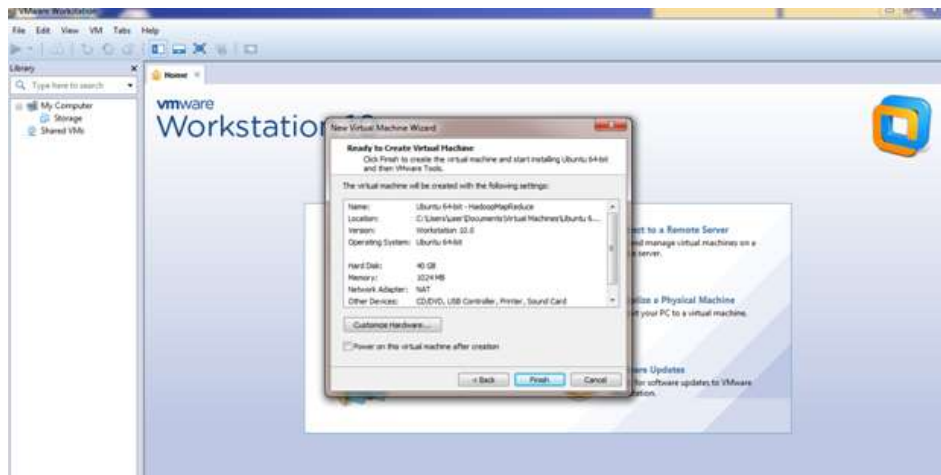
Step: 4 – Browse for .iso file of an operating system (Ubuntu)



Step: 5– Specify the hardware capacity (Hard-disk size)

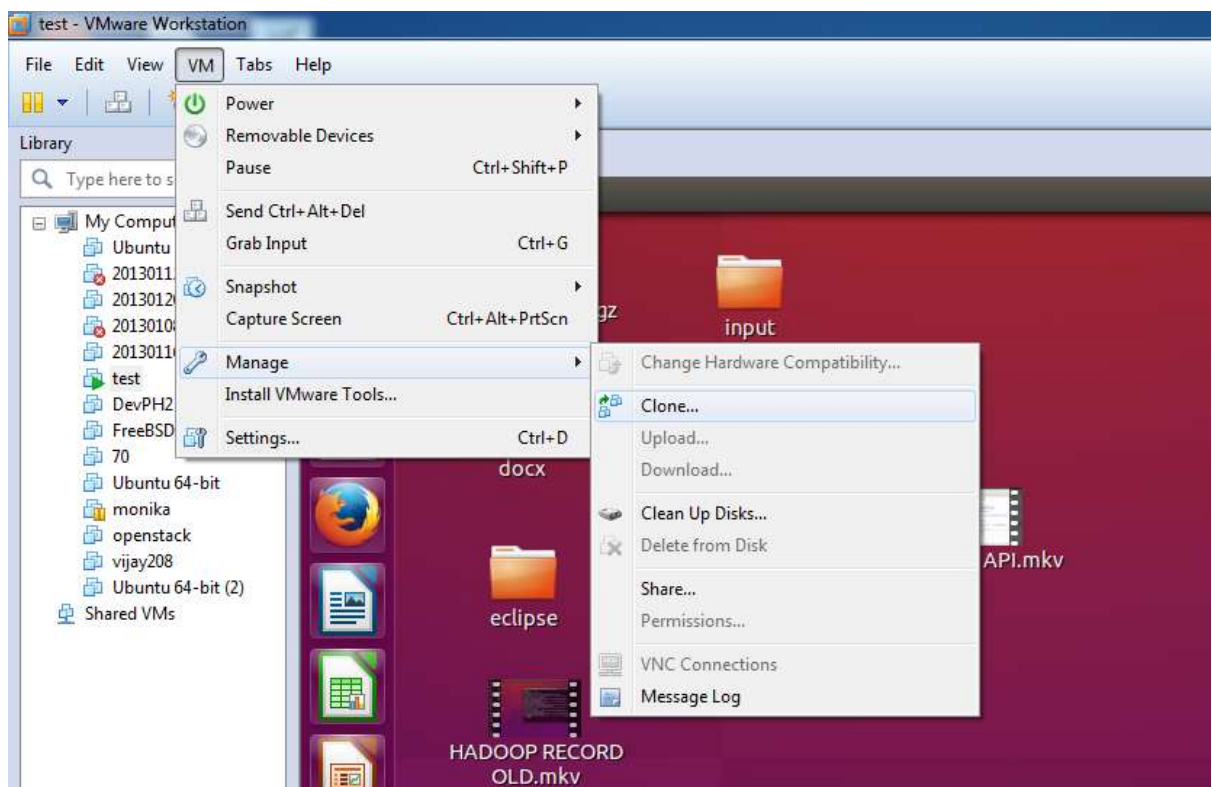


Step: 6 - Finish



b) To create Virtual Clone.

Step: 1 –Click VM -> Manage -> Clone



RESULT

Thus the procedure to create virtual machine with instance and virtual clone is done successfully.

EX. NO: 2

DATE: ATTACHING A VIRTUAL BLOCK

AIM

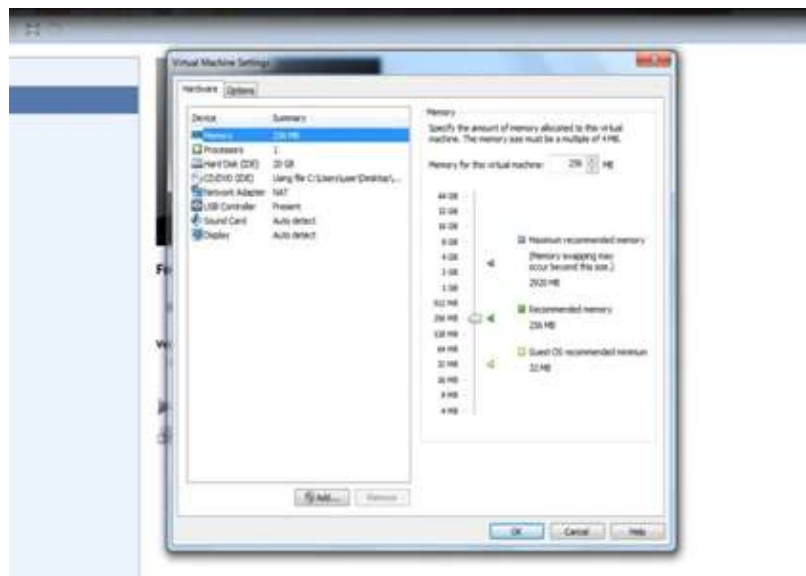
To find procedure to attach a virtual block to virtual machine and check whether it holds data even after the release of the virtual machine.

- a) Adding an additional block to Hard-Disk in Virtual Machine.

PROCEDURE

- a) **Adding an additional block to Hard-Disk in Virtual Machine.**

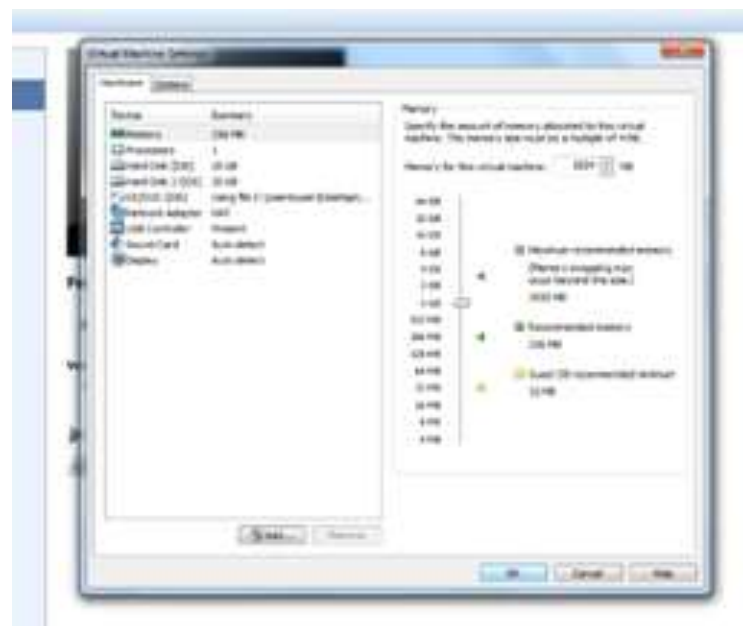
Step: 1 - Select hard Disk from the virtual machine settings dialog box. Select add button. In the Add Hardware wizard select the hardware type as hard disk and click next. Select the virtual disk type as IDE and click next.



Step: 2 – In the next page select the Create a new virtual disk option and click next. Provide the disk capacity as 20GB and select store virtual disk as single file option and click next. Specify the disk file and click finish.



Step: 3 – Additional Hard-Disk



RESULT

Thus the procedure to attach a volume to a virtual instance and adding an additional block to Hard-Disk in Virtual Machine is completed successfully.

EX. NO: 3

**DATE: INSTALL A C COMPILER IN THE VIRTUAL MACHINE AND
EXECUTE A SAMPLE PROGRAM**

AIM


To install and run a C Compiler in Ubuntu.

PROCEDURE

Step 1 – Open Console

Step 2 – Install gcc


- a) gcc -v
- b) sudo apt-get install gcc



The screenshot shows a terminal window with the following output:

```
student@student-Vostro-38002:~$ sudo apt-get install gcc
p --enable-plugins --with-system-zlib --disable-browser-plugin --enable-java-awt
gtk --enable-gtk-cairo --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-4.8-and64/jr
e --enable-java-home --with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-4.8-and64
--with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-4.8-and64 --with-arch-dir
ctory=amd64 --with-ecj-jar=/usr/share/java/eclipse-ecj.jar --enable-objc-gc --en
able-multitarch --disable-webkit --with-arch=i386 --with-abi=amd64 --with-multil
ib-list=i386,i486 --with-tune=generic --enable-checking=release --build=x86_64
4-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 4.8.4 (Ubuntu 4.8.4-1ubuntu1~12.04.3)
student@student-Vostro-38002:~$ sudo apt-get install gcc
[sudo] password for student:
Sorry, try again.
[sudo] password for student:
Reading package lists... Done
Building dependency tree
Reading state information... Done
gcc is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
W: Ignoring file 'cloudarchive-kilo.list' in directory '/etc/apt/sources.list.d
/' as it has an invalid filename extension
W: Ignoring file 'cloudarchive-kilo.list' in directory '/etc/apt/sources.list.d
/' as it has an invalid filename extension
student@student-Vostro-38002:~$
```

- c) sudo apt-get install build-essential

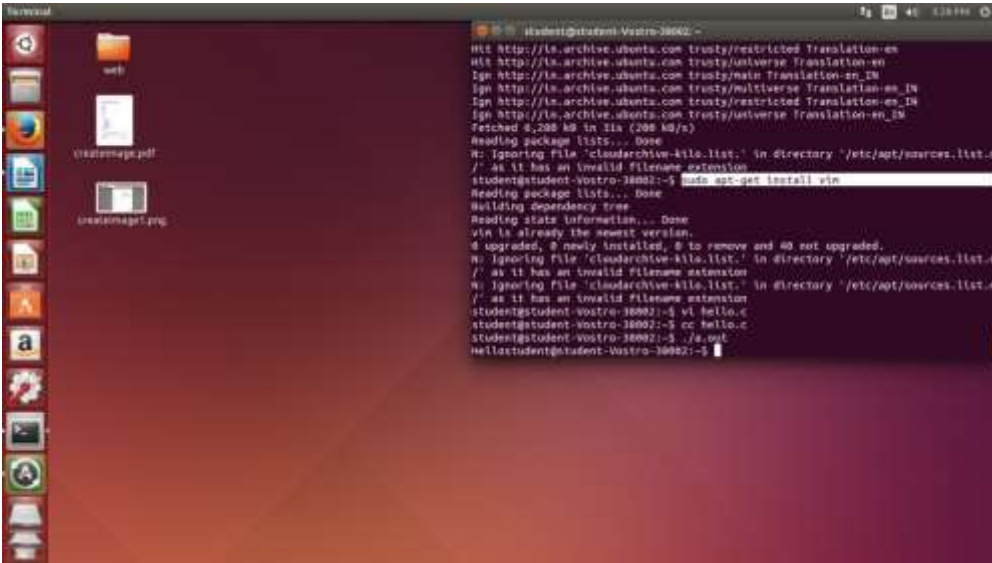


```
student@student-Vostro-38002:~$ sudo apt-get install build-essential
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  dpkg-dev fakeroot g++ g++-4.8 libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libfakeroot
  libstdc++-4.8-dev
Suggested packages:
  debian-keyring g++-multilib g++-4.8-multilib gcc-4.8-doc libstdc++-4.8-dbg
  libstdc++-4.8-doc
The following NEW packages will be installed:
  build-essential dpkg-dev fakeroot g++ g++-4.8 libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libfakeroot
  libstdc++-4.8-dev
0 upgraded, 10 newly installed, 0 to remove and 0 not upgraded.
Need to get 28.1 MB of archives.
After this operation, 42.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu/trusty-updates/main libstdc++-4.8-dev
and4.8.4-2ubuntu1-14.04.3 [1,053 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu/trusty-updates/main g++-4.8 amd64 4.8
.4-2ubuntu1-14.04.3 [18.1 MB]
39% [2 g++-4.8 6,777 kB/18.1 MB 37%]
```

C compiler will be installed

Step 3 - To Install Vim Editor

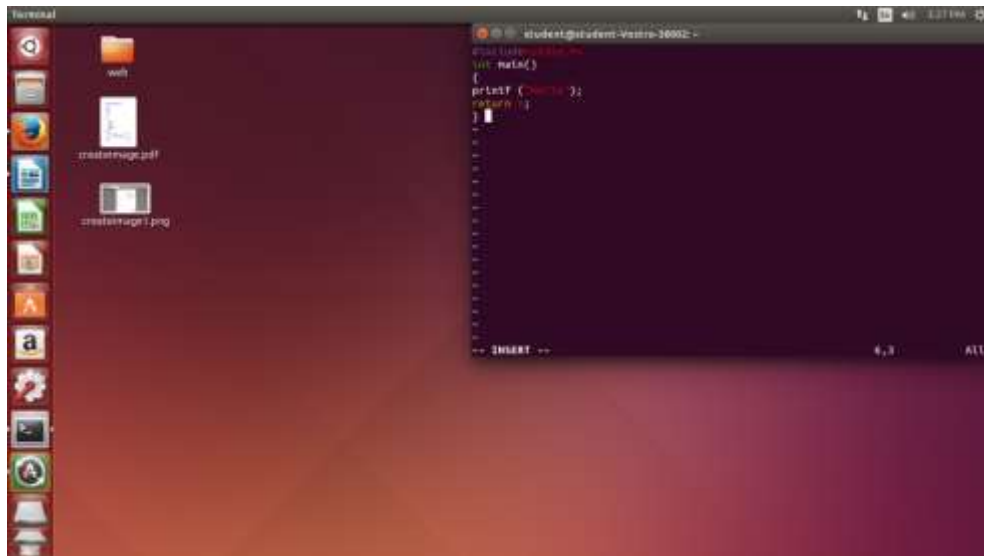
- a) sudo apt-get update
- b) sudo apt-get install vim



```
student@student-Vostro-38002:~$ sudo apt-get update
Hit http://in.archive.ubuntu.com/trusty/restricted Translation-en
Hit http://in.archive.ubuntu.com/trusty/universe Translation-en
Ign http://in.archive.ubuntu.com/trusty/main Translation-en_IN
Ign http://in.archive.ubuntu.com/trusty/multiverse Translation-en_IN
Ign http://in.archive.ubuntu.com/trusty/restricted Translation-en_IN
Ign http://in.archive.ubuntu.com/trusty/universe Translation-en_IN
Fetched 8,298 kB in 11s (298 kB/s)
Reading package lists... Done
W: Ignoring file 'cloudarchive-kilo.list.' in directory '/etc/apt/sources.list.d'
/* as it has an invalid filename extension
student@student-Vostro-38002:~$ sudo apt-get install vim
Reading package lists... Done
Building dependency tree
Reading state information... Done
vim is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
W: Ignoring file 'cloudarchive-kilo.list.' in directory '/etc/apt/sources.list.d'
/* as it has an invalid filename extension
W: Ignoring file 'cloudarchive-kilo.list.' in directory '/etc/apt/sources.list.d'
/* as it has an invalid filename extension
student@student-Vostro-38002:~$ vi hello.c
student@student-Vostro-38002:~$ cc hello.c
student@student-Vostro-38002:~$ ./a.out
hello
student@student-Vostro-38002:~$
```

Step 4–Type the C program in terminal

```
hello.c
#include<stdio.h>
int main()
{
printf ("Hello");
return 0;
}
```



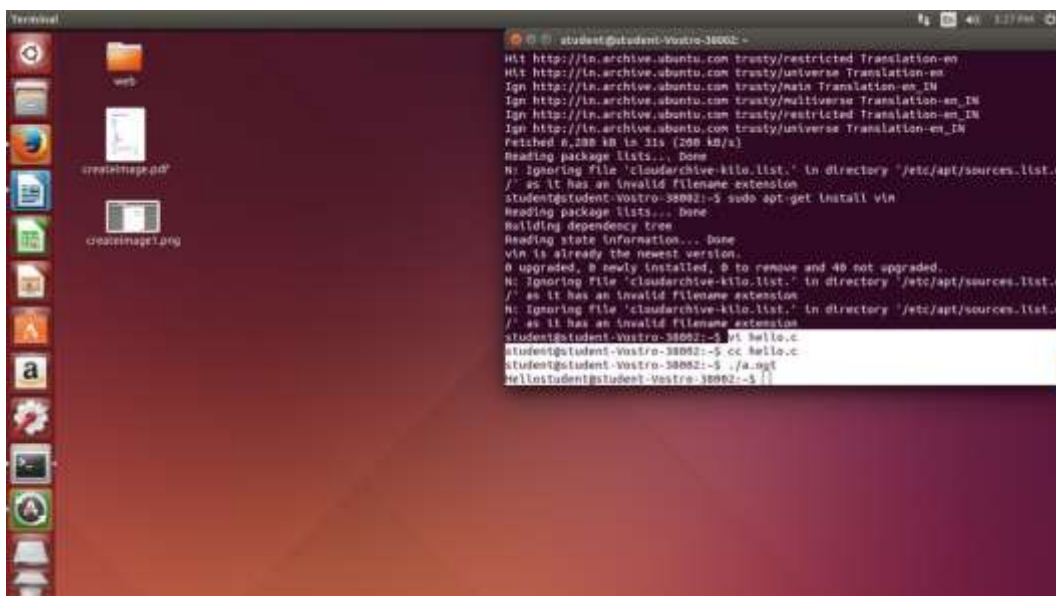
Step 5 – Compile and Execute C Program

cc hello.c

./a.out

OUTPUT

Hello



RESULT

Thus the procedure to install and run a C Compiler in Ubuntu is done successfully.

EX. NO: 4

DATE: SINGLE-NODE HADOOP INSTALLATION

AIM

To set-up one node Hadoop cluster

- System Update
- Install Java
- Add a dedicated Hadoop user
- Install SSH and setup SSH certificates
- Check if SSH works
- Install Hadoop
- Modify Hadoop config files
- Format Hadoop filesystem
- Start Hadoop
- Check Hadoop through web UI
- Stop Hadoop

THEORY

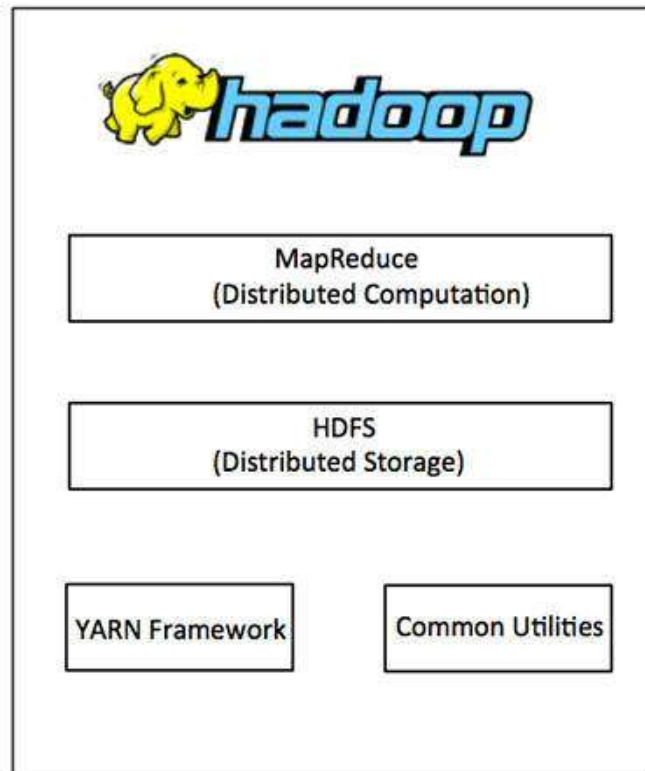
Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. A Hadoop frame-worked application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

HADOOP ARCHITECTURE

Hadoop framework includes following four modules:

- **Hadoop Common:** These are Java libraries and utilities required by other Hadoop modules. These libraries provides filesystem and OS level abstractions and contains the necessary Java files and scripts required to start Hadoop.
- **Hadoop YARN:** This is a framework for job scheduling and cluster resource management.
- **Hadoop Distributed File System (HDFS™):** A distributed file system that provides high-throughput access to application data.
- **Hadoop MapReduce:** This is YARN-based system for parallel processing of large data sets.

We can use following diagram to depict these four components available in Hadoop framework.



Hadoop Architecture

Since 2012, the term "Hadoop" often refers not just to the base modules mentioned above but also to the collection of additional software packages that can be installed on top of or alongside Hadoop, such as Apache Pig, Apache Hive, Apache HBase, Apache Spark etc.

PROCEDURE

Step 1 – System Update

\$ sudo apt-get update

```
ubuntu@ubuntu-VirtualBox: ~  
ubuntu@ubuntu-VirtualBox:~$ sudo apt-get update  
[sudo] password for ubuntu:  
Hit http://in.archive.ubuntu.com wily InRelease  
Get:1 http://security.ubuntu.com wily-security InRelease [65.9 kB]  
Get:2 http://in.archive.ubuntu.com wily-updates InRelease [65.9 kB]  
Get:3 http://security.ubuntu.com wily-security/main Sources [53.8 kB]  
Hit http://in.archive.ubuntu.com wily-backports InRelease  
Get:4 http://security.ubuntu.com wily-security/restricted Sources [2,854 B]  
Get:5 http://security.ubuntu.com wily-security/universe Sources [13.9 kB]  
Get:6 http://security.ubuntu.com wily-security/multiverse Sources [2,784 B]  
Get:7 http://security.ubuntu.com wily-security/main amd64 Packages [172 kB]  
Get:8 http://security.ubuntu.com wily-security/restricted amd64 Packages [10.9 kB]  
Get:9 http://security.ubuntu.com wily-security/universe amd64 Packages [56.2 kB]  
Get:10 http://security.ubuntu.com wily-security/multiverse amd64 Packages [6,248 B]  
Get:11 http://security.ubuntu.com wily-security/main i386 Packages [169 kB]  
Get:12 http://security.ubuntu.com wily-security/restricted i386 Packages [10.8 kB]  
100% [Waiting for headers] [Waiting for headers] 73.8 kB/s 0s
```

Step 2 – Install Java and Set JAVA_HOME

//This first thing to do is to setup the webupd8 ppa on your system. Run the following command and proceed.

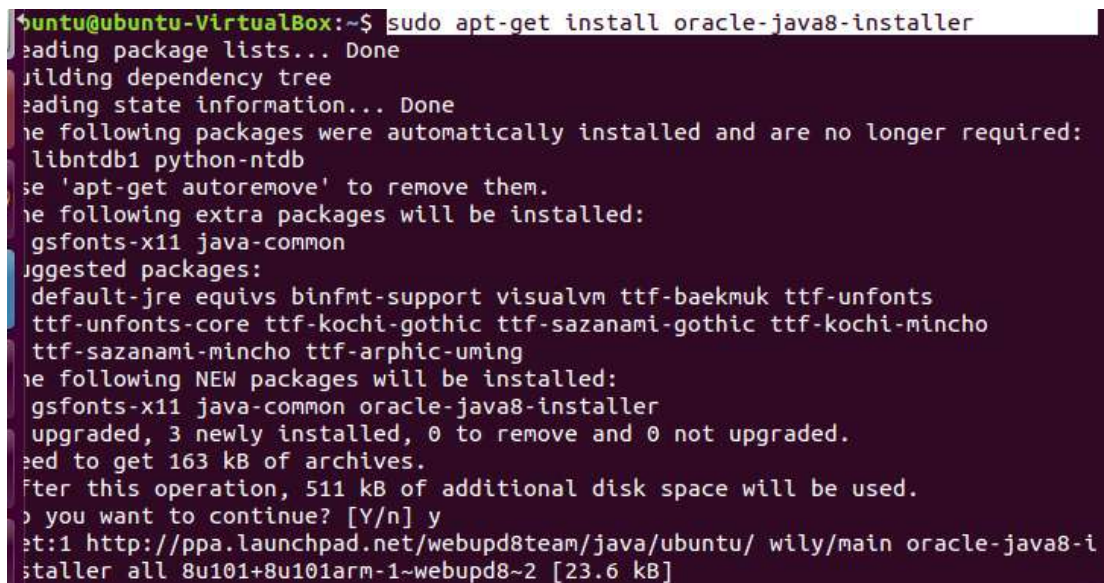
```
$ sudo apt-add-repository ppa:webupd8team/java
```

```
$ sudo apt-get update
```

//After setting up the ppa repository, update the package cache as well.

//Install the Java 8 installer

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

A terminal window screenshot showing the command 'sudo apt-get install oracle-java8-installer' being executed. The output shows the package lists being read, a dependency tree being built, and state information being read. It lists packages that were automatically installed and are no longer required (libntdb1, python-ntdb) and suggests removing them with 'apt-get autoremove'. It then lists extra packages to be installed (gsfonts-x11, java-common) and suggested packages (default-jre, equivs, binfmt-support, visualvm, etc.). It shows that 3 new packages will be installed, including oracle-java8-installer, and that 163 kB of archives will be needed, with 511 kB of additional disk space used. It asks for confirmation to continue, and the user responds 'y'. Finally, it shows the source of the package: 'http://ppa.launchpad.net/webupd8team/java/ubuntu/ wily/main oracle-java8-installer all 8u101+8u101arm-1-webupd8-2 [23.6 kB]'.

```
ubuntu@ubuntu-VirtualBox:~$ sudo apt-get install oracle-java8-installer
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libntdb1 python-ntdb
Use 'apt-get autoremove' to remove them.
The following extra packages will be installed:
  gsfonts-x11 java-common
Suggested packages:
  default-jre equivs binfmt-support visualvm ttf-baekmuk ttf-unfonts
  ttf-unfonts-core ttf-kochi-gothic ttf-sazanami-gothic ttf-kochi-mincho
  ttf-sazanami-mincho ttf-arphic-uming
The following NEW packages will be installed:
  gsfonts-x11 java-common oracle-java8-installer
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 163 kB of archives.
After this operation, 511 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ppa.launchpad.net/webupd8team/java/ubuntu/ wily/main oracle-java8-i
nstaller all 8u101+8u101arm-1-webupd8-2 [23.6 kB]
```

// After the installation is finished, Oracle Java is setup. Run the java command again to check the version and vendor.

[or]

```
$ sudo apt-get install default-jdk
```

```
$ java -version
```

A terminal window screenshot showing the command 'java -version' being executed. The output shows the Java version as '1.8.0_101', the Java(TM) SE Runtime Environment (build 1.8.0_101-b13), and the Java HotSpot(TM) 64-Bit Server VM (build 25.101-b13, mixed mode).

```
ubuntu@ubuntu-VirtualBox:~$ java -version
java version "1.8.0_101"
Java(TM) SE Runtime Environment (build 1.8.0_101-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.101-b13, mixed mode)
```

Step 3 – Add a dedicated Hadoop user

```
$ sudo addgroup hadoop
```



```
ubuntu@ubuntu-VirtualBox:~$ sudo addgroup hadoop
Adding group `hadoop' (GID 1001) ...
Done.
```

\$ sudo adduser --ingroup hadoop hduser

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

// Add hduser to sudo user group

\$ sudo adduser hduser sudo

```
ubuntu@ubuntu-VirtualBox:~$ sudo adduser hduser sudo
Adding user `hduser' to group `sudo' ...
Adding user hduser to group sudo
Done.
ubuntu@ubuntu-VirtualBox:~$
```

Step 4 – Install SSH and Create Certificates

\$ sudo apt-get install ssh

```
ubuntu@ubuntu-VirtualBox:~$ sudo apt-get install ssh
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libntdb1 python-ntdb
Use 'apt-get autoremove' to remove them.
The following extra packages will be installed:
  libck-connector0 ncurses-term openssh-server openssh-sftp-server
  ssh-import-id
Suggested packages:
  rssh molly-guard monkeysphere
The following NEW packages will be installed:
  libck-connector0 ncurses-term openssh-server openssh-sftp-server ssh
  ssh-import-id
0 upgraded, 6 newly installed, 0 to remove and 8 not upgraded.
Need to get 661 kB of archives.
```

\$ su hduser

```
ubuntu@ubuntu-VirtualBox:~$ su hduser
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

\$ ssh-keygen -t rsa -P ""

```
hduser@ubuntu-VirtualBox:~$ ssh-keygen -t rsa -P ""
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hduser/.ssh/id_rsa):
Created directory '/home/hduser/.ssh'.
Your identification has been saved in /home/hduser/.ssh/id_rsa.
Your public key has been saved in /home/hduser/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:K/F5oNmAqhY02Axp0vzew4EnrN+UGgDTgxIiFPHpT7Q hduser@ubuntu-VirtualBox
The key's randomart image is:
+---[RSA 2048]-----+
|=@o                    |
|@.* .                  |
|=* * o                 |
|.o= *.+                |
|. .=.EooS              |
|...= *B +              |
| o. *+.= .             |
| o o .. .              |
|o                       |
+-----[SHA256]-----+
```

// Set Environmental variables

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

```
hduser@ubuntu-VirtualBox:~$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_k
eys
```

Step 5 – Check if SSH works

\$ ssh localhost


```
hduser@ubuntu-VirtualBox: ~  
hduser@ubuntu-VirtualBox:~$ ssh localhost  
The authenticity of host 'localhost (127.0.0.1)' can't be established.  
ECDSA key fingerprint is SHA256:+j+WF1JPs00vl5mgcc7v9A/rU8jVQEHE8WfLmt2aEo8.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.  
Welcome to Ubuntu 15.10 (GNU/Linux 4.2.0-41-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com/  
  
9 packages can be updated.  
9 updates are security updates.  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.
```

Step 6 – Install Hadoop

\$ wget https://archive.apache.org/dist/hadoop/core/hadoop-2.8.4/hadoop-2.8.4.tar.gz

// Extract Hadoop-2.7.2

\$ sudo tar xvzf hadoop-2.7.2.tar.gz

```
hduser@ubuntu-VirtualBox:~$ tar xvzf hadoop-2.7.2.tar.gz
```

// Create a folder 'hadoop' in /usr/local

\$ sudo mkdir -p /usr/local/hadoop

```
hduser@ubuntu-VirtualBox:~$ sudo mkdir -p /usr/local/hadoop  
[sudo] password for hduser:
```

// Move the Hadoop folder to /usr/local/hadoop

\$ sudo mv hadoop-2.7.2 /usr/local/hadoop

```
hduser@ubuntu-VirtualBox:~$ sudo mv hadoop-2.7.2 /usr/local/hadoop
```

// Assigning read and write access to Hadoop folder

\$ sudo chown -R hduser:hadoop /usr/local/hadoop

```
hduser@ubuntu-VirtualBox:/usr/local/hadoop/hadoop-2.7.2$ sudo chown hduser:hadoo  
p -R /usr/local/hadoop  
hduser@ubuntu-VirtualBox:/usr/local/hadoop/hadoop-2.7.2$
```

Step 7 - Modify Hadoop config files

//Hadoop Environmental variable setting – The following files will be modified

1. `~/.bashrc`
2. `/usr/local/hadoop/hadoop-2.7.2/etc/hadoop/hadoop-env.sh`
3. `/usr/local/hadoop/hadoop-2.7.2/etc/hadoop/core-site.xml`
4. `/usr/local/hadoop/hadoop-2.7.2/etc/hadoop/hdfs-site.xml`
5. `/usr/local/hadoop/hadoop-2.7.2/etc/hadoop/yarn-site.xml`
6. `/usr/local/hadoop/hadoop-2.7.2/etc/hadoop/mapred-site.xml.template`

\$ sudo nano ~/.bashrc

// Add the following lines at the end of the file

```
export JAVA_HOME=/usr/lib/jvm/java-8-oracle
export HADOOP_HOME=/usr/local/hadoop/hadoop-2.7.2
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
export PATH=$PATH:/usr/local/hadoop/hadoop-2.7.2/bin
```

```
hduser@ubuntu-VirtualBox: ~
GNU nano 2.4.2      File: /home/hduser/.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/java-7-openjdk-amd64
export HADOOP_INSTALL=/usr/local/hadoop/hadoop-2.7.2
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

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

// Configure Hadoop Files

\$ cd /usr/local/hadoop/hadoop-2.7.2/etc/hadoop/

\$ sudo nano hadoop-env.sh

```
hduser@ubuntu-VirtualBox: /usr/local/hadoop/hadoop-2.7.2/etc/hadoop$ cd
hduser@ubuntu-VirtualBox: ~$ cd /usr/local/hadoop/hadoop-2.7.2/etc/hadoop
hduser@ubuntu-VirtualBox: /usr/local/hadoop/hadoop-2.7.2/etc/hadoop$ ls
capacity-scheduler.xml      httpfs-env.sh              mapred-env.sh
configuration.xml           httpfs-log4j.properties   mapred-queues.xml.template
container-executor.cfg      httpfs-signature.secret   mapred-site.xml.template
core-site.xml               httpfs-site.xml            slaves
hadoop-env.cmd              kms-acls.xml               ssl-client.xml.example
hadoop-env.sh               kms-env.sh                 ssl-server.xml.example
hadoop-metrics2.properties kms-log4j.properties      yarn-env.cmd
hadoop-metrics.properties  kms-site.xml               yarn-env.sh
hadoop-policy.xml           log4j.properties          yarn-site.xml
hdfs-site.xml               mapred-env.cmd
hduser@ubuntu-VirtualBox: /usr/local/hadoop/hadoop-2.7.2/etc/hadoop$ sudo nano ha
doop-env.sh
```

// Add following line in hadoop-env.sh – Set JAVA variable in Hadoop

```
# The java implementation to use.
export JAVA_HOME=/usr/lib/jvm/java-8-oracle
```

h

```
hduser@ubuntu-VirtualBox: /usr/local/hadoop/hadoop-2.7.2/etc/hadoop
GNU nano 2.4.2 File: hadoop-env.sh Modified

# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64

# The jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
# protocol. Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}

export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}

# Extra Java CLASSPATH elements. Automatically insert capacity-scheduler.
for f in $HADOOP_HOME/contrib/capacity-scheduler/*.jar; do
    if [ "$HADOOP_CLASSPATH" ]; then
        export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f
    else

```

// Create datanode and namenode

\$ sudo mkdir -p /usr/local/hadoop_tmp/hdfs/namenode

\$ sudo mkdir -p /usr/local/hadoop_tmp/hdfs/datanode

// Changing ownership to hadoop_tmp

\$ sudo chown -R hduser:hadoop /usr/local/hadoop_tmp

```
hduser@ubuntu-VirtualBox:~$ sudo mkdir -p /usr/local/hadoop_tmp/hdfs/namenode
hduser@ubuntu-VirtualBox:~$ sudo mkdir -p /usr/local/hadoop_tmp/hdfs/datanode
hduser@ubuntu-VirtualBox:~$ sudo chown hduser:hadoop -R /usr/local/hadoop_tmp
```

// Edit hdfs-site.xml

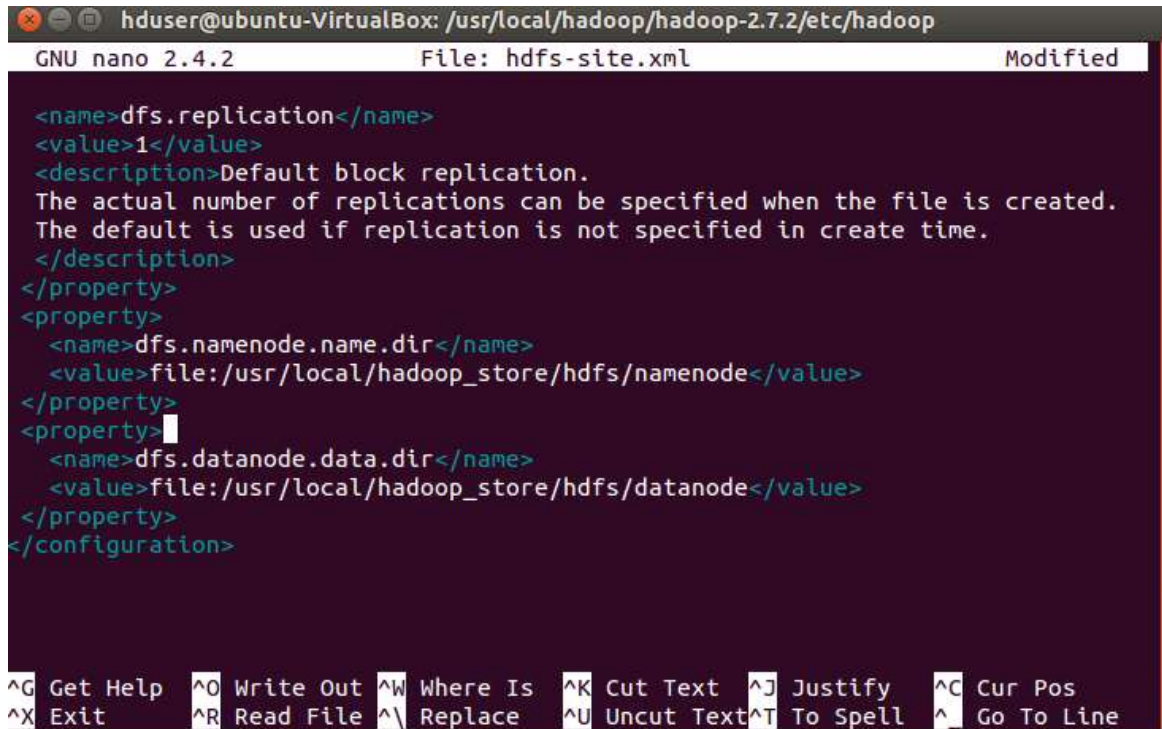
\$ sudo nano hdfs-site.xml

// Add the following lines between <configuration> </configuration>

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:/usr/local/hadoop_tmp/hdfs/namenode</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
```



```
<value>file:/usr/local/hadoop_tmp/hdfs/datanode</value>
</property>
</configuration>
```



```
hduser@ubuntu-VirtualBox: /usr/local/hadoop/hadoop-2.7.2/etc/hadoop
GNU nano 2.4.2 File: hdfs-site.xml Modified

<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_store/hdfs/namenode</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:/usr/local/hadoop_store/hdfs/datanode</value>
</property>
</configuration>

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

// Edit core-site.xml

\$ sudo nano core-site.xml

// Add the following lines between <configuration> </configuration>

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

// Edit yarn-site.xml

\$ sudo nano yarn-site.xml

// Add the following lines between <configuration> </configuration>

```
<configuration>
<property>
```

```
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
</property>
<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.Shuffle-Handler</value>
</property>
</configuration>
```

// Edit mapred-site.xml

```
$ cp /usr/local/hadoop/hadoop-2.7.2/etc/hadoop/mapred-site.xml.template
/usr/local/hadoop/hadoop-2.7.2/etc/hadoop/mapred-site.xml
```

```
hduser@ubuntu-VirtualBox:~$ cp /usr/local/hadoop/hadoop-2.7.2/etc/hadoop/mapred-
site.xml.template /usr/local/hadoop/hadoop-2.7.2/etc/hadoop/mapred-site.xml
```

\$ sudo nano mapred-site.xml

// Add the following lines between <configuration> </configuration>

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

Step 8 – Format Hadoop File System

```
$ cd /usr/local/hadoop/hadoop-2.7.2/bin
```

\$ hadoop namenode -format

```
hduser@ubuntu-VirtualBox:/usr/local/hadoop$ hadoop namenode -format
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

16/07/15 22:50:27 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
```

Step 9 - Start Hadoop

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

// Starting dfs services

\$ start-dfs.sh

```

hduser@ubuntu-VirtualBox: /usr/local/hadoop/hadoop-2.7.2/sbin$ start-dfs.sh
16/07/15 22:55:47 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop/hadoop-2.7.2/logs/hadoop-hduser-namenode-ubuntu-VirtualBox.out
localhost: starting datanode, logging to /usr/local/hadoop/hadoop-2.7.2/logs/hadoop-hduser-datanode-ubuntu-VirtualBox.out
Starting secondary namenodes [0.0.0.0]
The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
ECDSA key fingerprint is SHA256:+j+WF1JPs00vl5mgcc7v9A/rU8jVQEHE8WfLmt2aEo8.
Are you sure you want to continue connecting (yes/no)? yes
0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of known hosts
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/hadoop-2.7.2/logs/hadoop-hduser-secondarynamenode-ubuntu-VirtualBox.out

```

// Starting mapreduce services

\$ start-yarn.sh

```

hduser@ubuntu-VirtualBox: /usr/local/hadoop/hadoop-2.7.2/sbin$ start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/hadoop-2.7.2/logs/yarn-hduser-resourcemanager-ubuntu-VirtualBox.out
localhost: starting nodemanager, logging to /usr/local/hadoop/hadoop-2.7.2/logs/yarn-hduser-nodemanager-ubuntu-VirtualBox.out

```

\$ jps

```

hduser@ubuntu-VirtualBox: /usr/local/hadoop/hadoop-2.7.2/sbin$ jps
12425 SecondaryNameNode
12609 ResourceManager
12733 NodeManager
13131 Jps
12205 DataNode
12080 NameNode

```

Step 10 - Check Hadoop through web UI

Go to browser type <http://localhost:8088> – All Applications Hadoop Cluster

The screenshot shows the Hadoop web interface at `localhost:8088/cluster`. The page features the Hadoop logo and a sidebar with navigation links: Cluster, About, Nodes, Node Labels, Applications, NEW, NEW SAVING, SUBMITTED, ACCEPTED, RUNNING, FINISHED, FAILED, KILLED, Scheduler, and Tools. The main content area displays 'All Application' and 'Cluster Metrics'.

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VC Total
0	0	0	0	0	0 B	8 GB	0 B	0	8

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minim
Capacity Scheduler	[MEMORY]	<memory:1024, vC

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State
----	------	------	------------------	-------	-----------	------------	-------

Go to browser type <http://localhost:50070> – Hadoop Namenode

The screenshot shows the Hadoop Namenode web interface at `localhost:50070/dfshealth.html#tab-overview`. The page has a green header with the Hadoop logo and navigation links: Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. Below the header, there is a section for 'Hadoop, 2015'.

Step 11 - Stop Hadoop

`$ stop-dfs.sh`

`$ stop-yarn.sh`

RESULT

Thus the procedure to install single-node Hadoop is executed successfully.

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

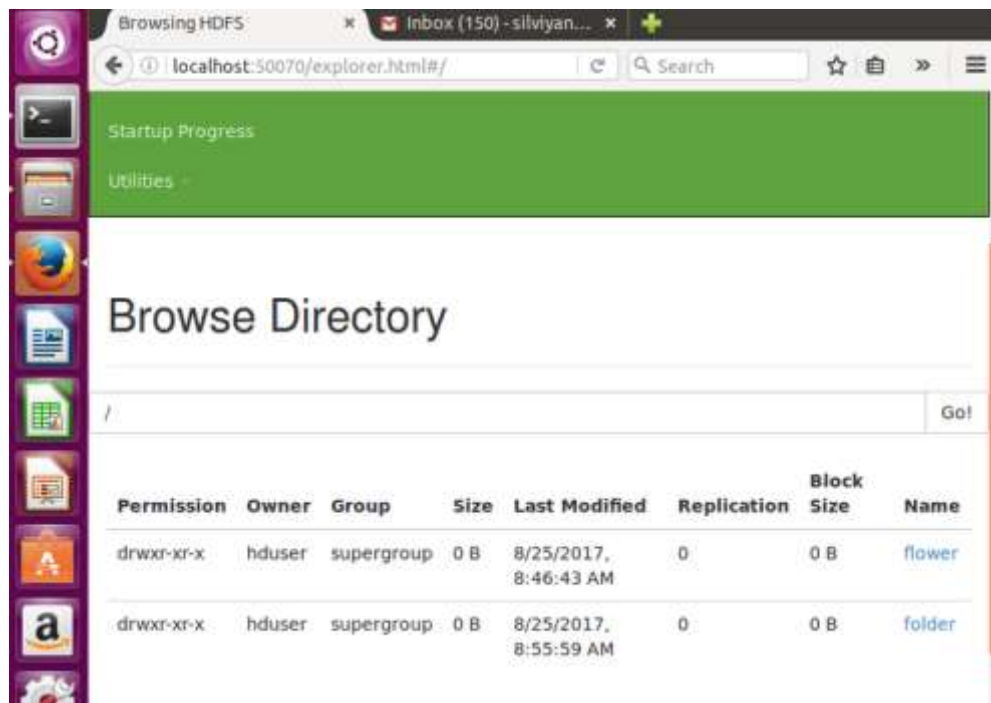
\$ su hduser

\$ 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
```

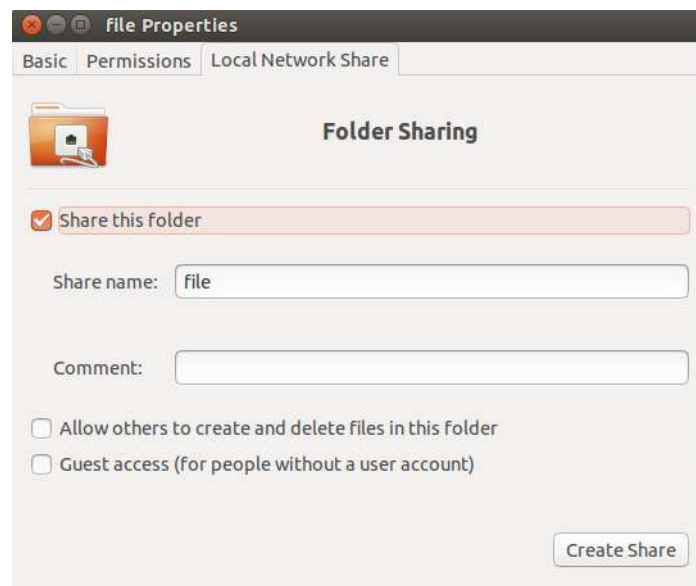
```
$ sudo apt-get install hadoop-hdfs-fuse
```

```
$ sudo mkdir -p <mount_point>
```

Sample :`sudo mkdir -p file`

```
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)

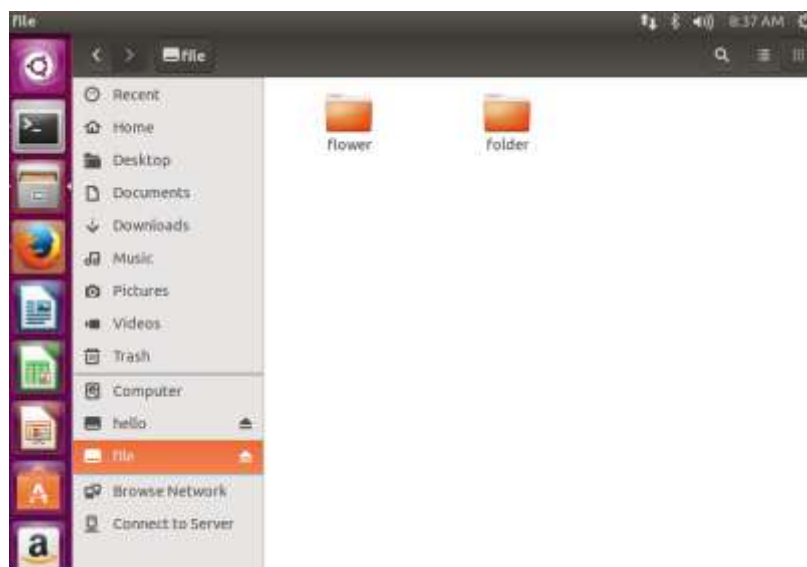
```
hduser@ubuntu:~$ cd /usr/local/hadoop/hadoop-2.7.2/etc/hadoop/
hduser@ubuntu:/usr/local/hadoop/hadoop-2.7.2/etc/hadoop$ sudo nano core-site.xml
[sudo] password for hduser:
```

```
<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

```
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.

EX. NO: 6

**DATE: INTERACTION WITH HADOOP API FOR ACCESSING HDFS FROM
LOCAL FILE SYSTEM**

AIM

To write a program to use the API's of Hadoop for copying File from Local File System to HDFS and to interact with it.

PROCEDURE

Step 1 – Open New Terminal

```
$ cd Desktop/  
  
$ mkdir inputdata  
  
$ cd inputdata/  
  
$ echo "Hai, Hello, How are you? How is your health?" >> file.txt  
  
$ cat >> file.txt
```

Step 2 – Download and open eclipse by creating workspace

Create a new java project.

Step 3 – Add jar to the project

You need to remove dependencies by adding jar files in the hadoop source folder. Now Click on **Project** tab and go to Properties. Under Libraries tab, click Add External JARs and select all the jars in the folder (click on 1st jar, and Press Shift and Click on last jar to select all jars in between and click ok)

/usr/local/hadoop/hadoop-2.7.2/share/hadoop/common and

/usr/local/hadoop/hadoop-2.7.2/share/hadoop/mapreduce folders.

Step 4 – Program

```
import org.apache.hadoop.conf.Configured;  
import org.apache.hadoop.util.Tool;  
import java.io.BufferedInputStream;  
import java.io.FileInputStream;  
import java.io.InputStream;  
import java.io.OutputStream;  
import org.apache.hadoop.conf.Configuration;  
import org.apache.hadoop.fs.FileSystem;  
import org.apache.hadoop.fs.Path;  
import org.apache.hadoop.io.IOUtils;  
import org.apache.hadoop.util.ToolRunner;  
public class HdfsWriter extends Configured implements Tool {  
    public static final String FS_PARAM_NAME = "fs.defaultFS";  
    public int run(String[] args) throws Exception {
```

```

    if (args.length < 2) {
        System.err.println("HdfsWriter /home/rec/Desktop/input/file.txt /hello");
        return 1;
    }
    String localInputPath = args[0];
    Path outputPath = new Path(args[1]);
    Configuration conf = getConf();
    System.out.println("configured filesystem = " + conf.get(FS_PARAM_NAME));
    FileSystem fs = FileSystem.get(conf);
    if (fs.exists(outputPath)) {
        System.err.println("output path exists");
        return 1;
    }
    OutputStream os = fs.create(outputPath);
    InputStream is = new BufferedInputStream(new FileInputStream(localInputPath));
    IOUtils.copyBytes(is, os, conf);
    return 0;
}
public static void main( String[] args ) throws Exception {
    int returnCode = ToolRunner.run(new HdfsWriter(), args);
    System.exit(returnCode);
}
}

```

Step 5 - Creatr JAR file

Now Click on the Run tab and click Run-Configurations. Click on New Configuration button on the left-top side and Apply after filling the following properties.

Step 6 - Export JAR file

Now click on File tab and select Export. under Java, select Runnable Jar.

In Launch Config – select the config fie you created in **Step 9** (WordCountConfig).

- Select an export destination (lets say desktop.)
- Under Library handling, select Extract Required Libraries into generated JAR and click Finish.
- Right-Click the jar file, go to Properties and under **Permissionstab**, Check Allow executing file as a program. and give Read and Write access to all the users

Step 7–Execute File

hadoop jar /home/rec/Desktop/HelloWrite.jar /home/rec/Desktop/input/file.txt /hello

Browsing HDFS - Mozilla Firefox

localhost:50070/explorer.html#

drwxr-xr-x	hduser	supergroup	0 B	8/23/2016, 12:42:01 AM	0	0 B	demo
drwxr-xr-x	hduser	supergroup	0 B	8/22/2016, 11:46:02 PM	0	0 B	flower
drwxr-xr-x	hduser	supergroup	0 B	8/22/2016, 11:58:20 PM	0	0 B	floweroutput
drwxr-xr-x	hduser	supergroup	0 B	8/4/2016, 11:37:37 PM	0	0 B	folder
drwxr-xr-x	hduser	supergroup	0 B	8/11/2016, 9:52:15 PM	0	0 B	grid
-rw-r--r--	hduser	supergroup	27 B	9/7/2016, 2:00:52 AM	1	128 MB	hai
-rw-r--r--	hduser	supergroup	27 B	9/1/2016, 11:35:32 PM	1	128 MB	hello
-rw-r--r--	hduser	supergroup	27 B	9/2/2016, 12:35:50 AM	1	128 MB	hello1
-rw-r--r--	hduser	supergroup	0 B	9/1/2016, 10:47:32 PM	1	128 MB	nydir
-rw-r--r--	hduser	supergroup	0 B	9/1/2016, 10:58:47 PM	1	128 MB	myjoe
-rw-r--r--	hduser	supergroup	0 B	9/1/2016, 10:54:32 PM	1	128 MB	mynancy
drwxr-xr-x	hduser	supergroup	0 B	8/11/2016, 9:54:38 PM	0	0 B	output
drwxr-xr-x	hduser	supergroup	0 B	9/2/2016, 12:26:10 AM	0	0 B	outtoday
-rw-r--r--	hduser	supergroup	27 B	9/2/2016, 12:36:11 AM	1	128 MB	ping
drwxr-xr-x	hduser	supergroup	0 B	8/11/2016, 11:54:23 PM	0	0 B	project
drwxr-xr-x	hduser	supergroup	0 B	8/22/2016, 11:16:36 PM	0	0 B	recant
drwx---	hduser	supergroup	0 B	8/4/2016, 11:40:37 PM	0	0 B	tmp
drwxr-xr-x	hduser	supergroup	0 B	8/5/2016, 12:46:00 AM	0	0 B	today

RESULT

Thus the program to use the API's of Hadoop for copying File from Local File System to HDFS is executed and output verified successfully.

EX. NO: 7

DATE: DEMONSTRATION OF MAP AND REDUCE TASKS

AIM

To write a wordcount program to demonstrate the use of Map and Reduce tasks.

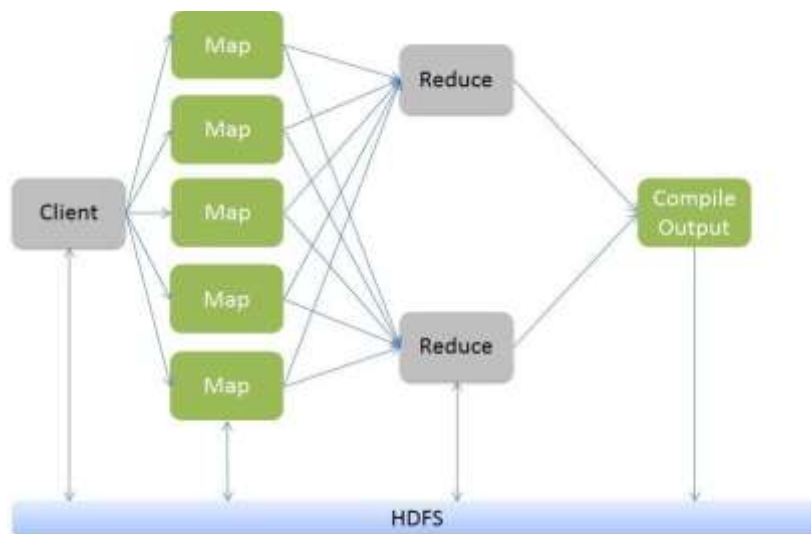
THEORY

Hadoop MapReduce is a programming paradigm at the heart of Apache Hadoop for providing massive scalability across hundreds or thousands of Hadoop clusters on commodity hardware. The MapReduce model processes large unstructured data sets with a distributed algorithm on a Hadoop cluster.

The term MapReduce represents two separate and distinct tasks Hadoop programs perform-Map Job and Reduce Job. Map job scales takes data sets as input and processes them to produce key value pairs. Reduce job takes the output of the Map job i.e. the key value pairs and aggregates them to produce desired results. The input and output of the map and reduce jobs are stored in HDFS.

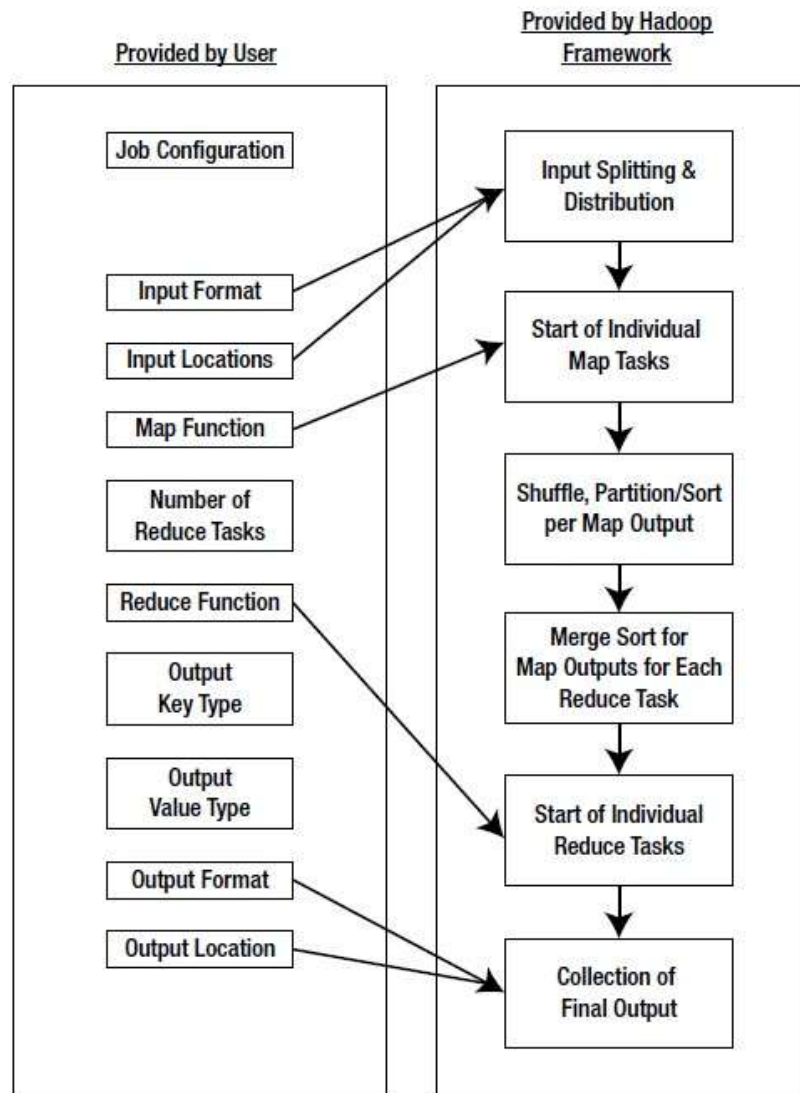
MAPREDUCE

Hadoop MapReduce (Hadoop Map/Reduce) is a software framework for distributed processing of large data sets on computing clusters. It is a sub-project of the Apache Hadoop project. Apache Hadoop is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models. MapReduce is the core component for data processing in Hadoop framework. In layman's term Mapreduce helps to split the input data set into a number of parts and run a program on all data parts parallel at once. The term MapReduce refers to two separate and distinct tasks. The first is the map operation, takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). The reduce operation combines those data tuples based on the key and accordingly modifies the value of the key.



MAPREDUCE ARCHITECTURE

The figure shown below illustrates the various parameters and modules that can be configured during a MapReduce operation:



MapReduce Architecture

JobConf is the framework used to provide various parameters of a MapReduce job to the Hadoop for execution. The Hadoop platform executes the programs based on configuration set using JobConf. The parameters being Map Function, Reduce Function, combiner, Partitioning function, Input and Output format. Partitioner controls the shuffling of the tuples when being sent from Mapper node to Reducer nodes. The total number of partitions done in the tuples is equal to the number of reduce nodes. In simple terms based on the function output the tuples are transmitted through different reduce nodes.

Input Format describes the format of the input data for a MapReduce job. Input location specifies the location of the datafile. Map Function/ Mapper convert the data into key value pair. For example let's consider daily temperature data of 100 cities for the past 10 years. In this the map function is written such a way that every temperature being mapped to the corresponding

city. Reduce Function reduces the set of tuples which share a key to a single tuple with a change in the value. In this example if we have to find the highest temperature recorded in the city the reducer function is written in such a way that it return the tuple with highest value i.e: highest temperature in the city in that sample data.

The number of Map and Reduce nodes can also be defined. You can set Partitioner function which partitions and transfer the tuples which by default is based on a hash function. In other words we can set the options such that a specific set of key value pairs are transferred to a specific reduce task. For example if your key value consists of the year it was recorded, then we can set the parameters such that all the keys of specific year are transferred to a same reduce task. The Hadoop framework consists of a single master and many slaves. Each master has JobTracker and each slave has TaskTracker. Master distributes the program and data to the slaves. Task tracker, as the name suggests keep track of the task directed to it and relays the information to the JobTracker. The JobTracker monitors all the status reports and re-initiates the failed tasks if any.

Combiner classes are run on map task nodes. It takes the tuples emitted by Map nodes as input. It basically does reduce operation on the tuples emitted by the map node. It is like a pre-reduce task to save a lot of bandwidth. We can also pass global constants to all the nodes using 'Counters'. They can be used to keep track of all events in map and reduce tasks. For example we can pass a counter to calculate the statistics of an event beyond a certain threshold.

PROCEDURE

Step 1 - Open Terminal

```
$ su hduser
```

Password:

Step 2 - Start dfs and mapreduce services

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

```
$ start-dfs.sh
```

```
$ start-yarn.sh
```

```
$ jps
```

Step 3 - Check Hadoop through web UI

```
// Go to browser type http://localhost:8088 – All Applications Hadoop Cluster
```

```
// Go to browser type http://localhost:50070 – Hadoop Namenode
```

Step 4 – Open New Terminal

```
$ cd Desktop/
```

```
$ mkdir inputdata
```

```
$ cd inputdata/
```

```
$ echo "Hai, Hello, How are you? How is your health?" >> hello.txt
```

```
$ cat >> hello.txt
```

Step 5 – Go back to old Terminal

```
$ hadoop fs -mkdir /folder
```

```
$ hadoop fs -copyFromLocal /home/hduser/Desktop/inputdata/hello.txt /folder
```

// Check in hello.txt in Namenode using Web UI

Step 6 – Download and open eclipse by creating workspace

Create a new java project.

Step 7 – Add jar to the project

You need to remove dependencies by adding jar files in the hadoop source folder. Now Click on **Project** tab and go to Properties. Under Libraries tab, click Add External JARs and select all the jars in the folder (click on 1st jar, and Press Shift and Click on last jar to select all jars in between and click ok)

/usr/local/hadoop/hadoop-2.7.2/share/hadoop/common and

/usr/local/hadoop/hadoop-2.7.2/share/hadoop/mapreduce folders.

Step -8 – WordCount Program

Create 3 java files named

- **WordCount.java**
- **WordCountMapper.java**
- **WordCountReducer.java**

WordCount.java

```
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;

import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;
import org.apache.hadoop.io.Text;

public class WordCount extends Configured implements Tool {

    @Override
    public int run(String[] arg0) throws Exception {
        // TODO Auto-generated method stub
        if(arg0.length<2)
```

```

        {
            System.out.println("check the command line arguments");
        }
        JobConf conf=new JobConf(WordCount.class);
        FileInputFormat.setInputPaths(conf, new Path(arg0[0]));
        FileOutputFormat.setOutputPath(conf, new Path(arg0[1]));
        conf.setMapperClass(WordMapper.class);
        conf.setReducerClass(WordReducer.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        JobClient.runJob(conf);

        return 0;
    }
    public static void main(String args[]) throws Exception
    {
        int exitcode=ToolRunner.run(new WordCount(), args);
        System.exit(exitcode);
    }
}

```

WordCountMapper.java

```

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.Mapper;

public class WordCountMapper extends MapReduceBase implements
Mapper<LongWritable,Text,Text,IntWritable>
{
    @Override
    public void map(LongWritable arg0, Text arg1, OutputCollector<Text, IntWritable>
arg2, Reporter arg3)
        throws IOException {
        // TODO Auto-generated method stub

        String s=arg1.toString();
        for(String word:s.split(" "))
        {
            arg2.collect(new Text(word),new IntWritable(1));
        }
    }
}

```

WordCountReducer.java

```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.io.Text;

public class WordCountReducer implements Reducer<Text,IntWritable,Text,IntWritable> {
    @Override
    public void configure(JobConf arg0) {
        // TODO Auto-generated method stub
    }
    @Override
    public void close() throws IOException {
        // TODO Auto-generated method stub
    }
    @Override
    public void reduce(Text arg0, Iterator<IntWritable> arg1, OutputCollector<Text,
IntWritable> arg2, Reporter arg3)
        throws IOException {
        // TODO Auto-generated method stub
        int count=0;
        while(arg1.hasNext())
        {
            IntWritable i=arg1.next();
            count+=i.get();
        }
        arg2.collect(arg0,new IntWritable(count));
    }
}
```

Step 9 - Creatr JAR file

Now Click on the Run tab and click Run-Configurations. Click on New Configuration button on the left-top side and Apply after filling the following properties.

Step 10 - Export JAR file

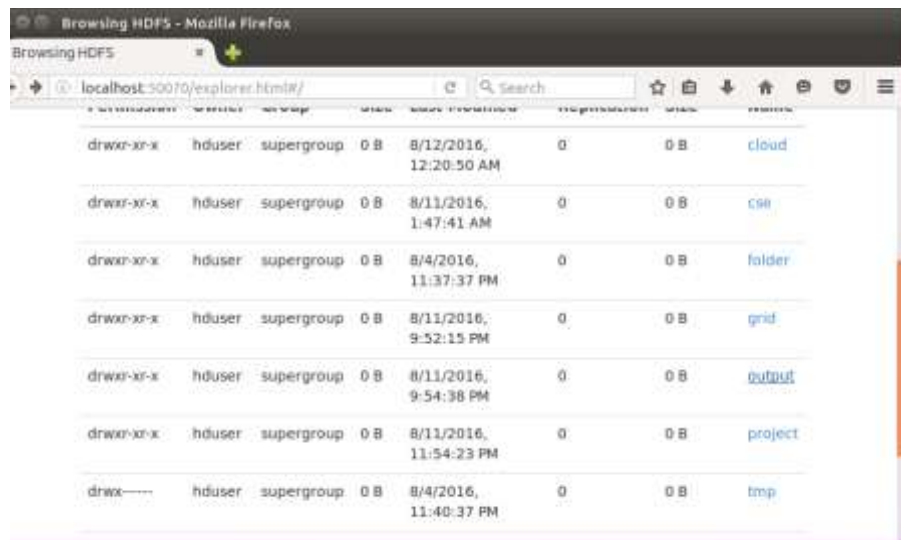
Now click on File tab and select Export. under Java, select Runnable Jar.

In Launch Config – select the config fie you created in **Step 9** (WordCountConfig).

- Select an export destination (lets say desktop.)
- Under Library handling, select Extract Required Libraries into generated JAR and click Finish.
- Right-Click the jar file, go to Properties and under **Permission**stab, Check Allow executing file as a program. and give Read and Write access to all the users

Step 11 – Go back to old Terminal for Execution of WordCount Program

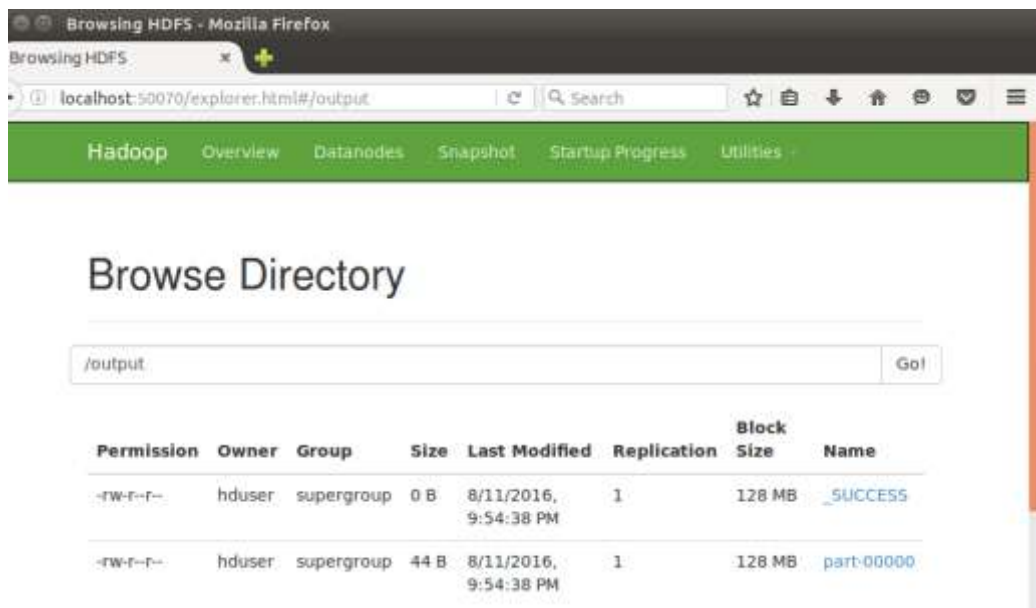
\$hadoop jar wordcount.jar/usr/local/hadoop/input/usr/local/hadoop/output



Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	hduser	supergroup	0 B	8/12/2016, 12:20:50 AM	0	0 B	cloud
drwxr-xr-x	hduser	supergroup	0 B	8/11/2016, 1:47:41 AM	0	0 B	c98
drwxr-xr-x	hduser	supergroup	0 B	8/4/2016, 11:37:37 PM	0	0 B	folder
drwxr-xr-x	hduser	supergroup	0 B	8/11/2016, 9:52:15 PM	0	0 B	grid
drwxr-xr-x	hduser	supergroup	0 B	8/11/2016, 9:54:38 PM	0	0 B	output
drwxr-xr-x	hduser	supergroup	0 B	8/11/2016, 11:54:23 PM	0	0 B	project
drwxr-xr-x	hduser	supergroup	0 B	8/4/2016, 11:40:37 PM	0	0 B	tmp

Step 12 – To view results in old Terminal

\$hdfs dfs -cat /usr/local/hadoop/output/part-r-00000



Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	hduser	supergroup	0 B	8/11/2016, 9:54:38 PM	1	128 MB	_SUCCESS
-rw-r--r--	hduser	supergroup	44 B	8/11/2016, 9:54:38 PM	1	128 MB	part-00000

Step 13 - To Remove folders created using hdfs

\$ hdfs dfs -rm -R /usr/local/hadoop/output

RESULT

Thus the program to write a wordcount program to that demonstrates the use of Map and Reduce tasks is done and output is verified successfully.

EX. NO: 8

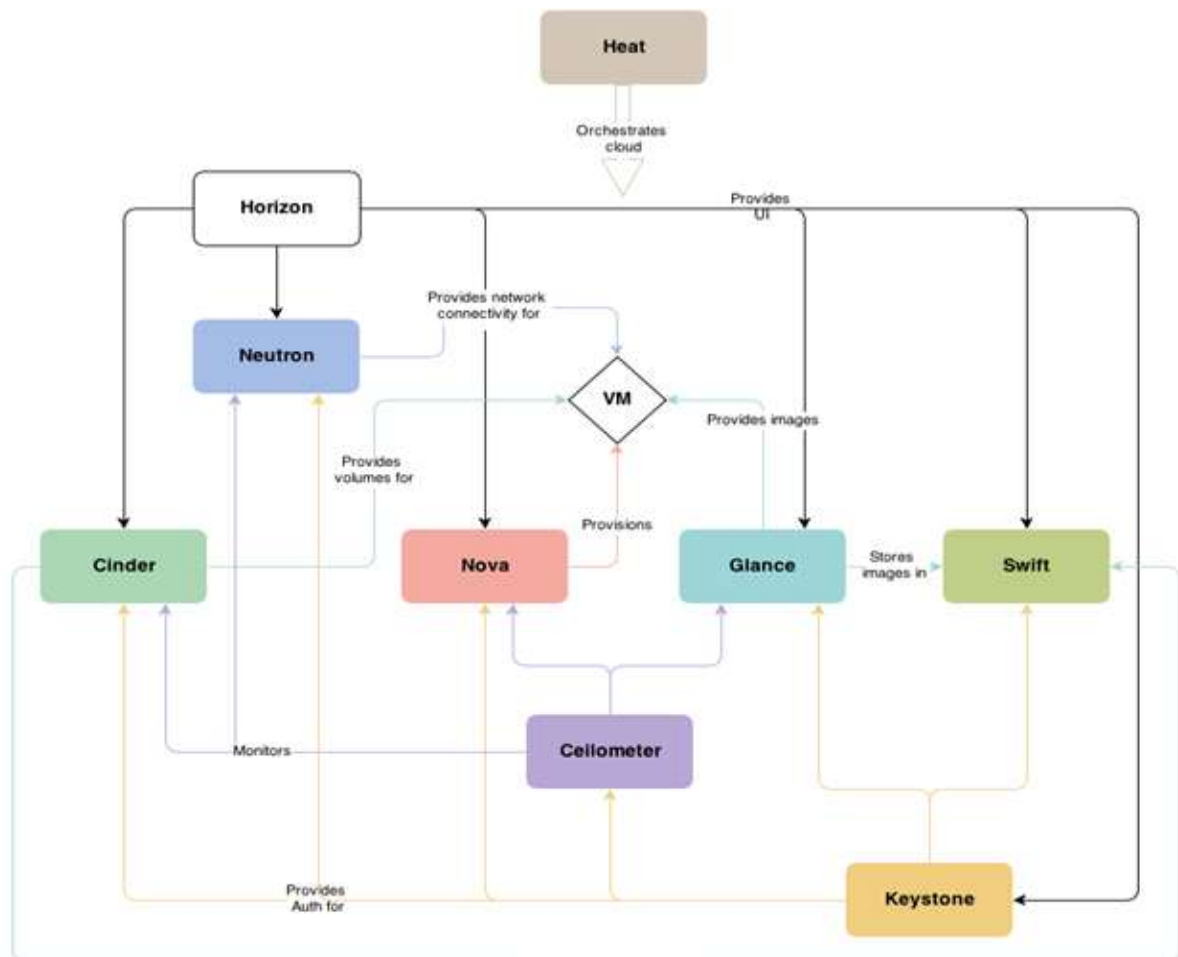
DATE: CREATION AND MANAGEMENT OF INSTANCE IN OPENSTACK

AIM

- a) To install Single node OpenStack-devstack.
- b) Creation, Management and Termination of Instances

THEORY

OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a data center. All of the above components are managed through a dashboard which gives administrators control while empowering their users to provision resources through a web interface. More specifically though, it is a collection of open source software that allows us to perform certain functions on the cloud. OpenStack was a pilot project launched by Rackspace and NASA which was founded in July 2010. The purpose behind the project was to provide open source software that enables any organization to create and offer cloud computing services running on standardized hardware.



OpenStack Architecture

Just like as the Linux distribution has multiple flavors that are supported by different foundations like RedHat and SUSE, it is believed that in the near future there is a strong possibility that OpenStack will have distributions, as well by the leading players and contributors to the open source project, including: RedHat, Ubuntu, and more.

OpenStack has a modular architecture with various code names for its components:

- Nova (Compute)
- Swift (Object Storage)
- Cinder (Block Storage)
- Glance (Image Service)
- Neutron (Networking)
- Horizon (Dashboard)
- Ceilometer (Telemetry)
- Heat (Orchestration)

PRODEEDURE

a) To install Single node OpenStack-devstack.

On execution of following commands, the openstack commands like nova, neutron, cinder, glance, keystone, glance, swift, horizon, celimeter would be installed. At the end of installation, IP will be generated.

```
root@ubuntu: sudo apt-get update
```

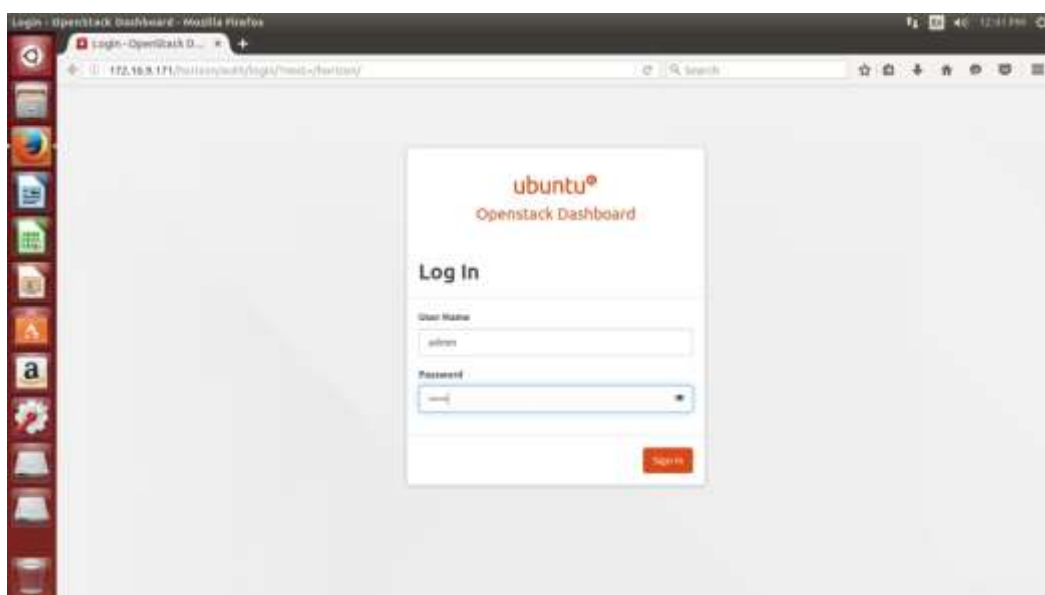
```
root@ubuntu: sudo apt-get install git
```

```
root@ubuntu: git clone https://git.openstack.org/openstack-devstack/devstack
```

```
root@ubuntu: cd devstack
```

```
root@ubuntu: ls
```

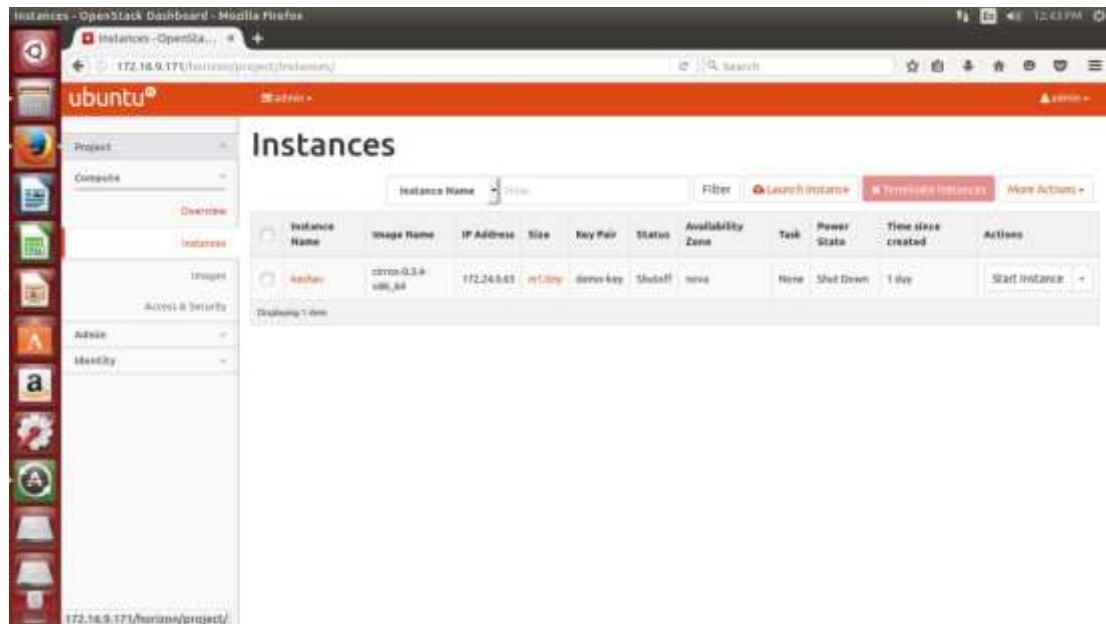
```
root@ubuntu: ./stack.sh
```



Openstack Dashboard

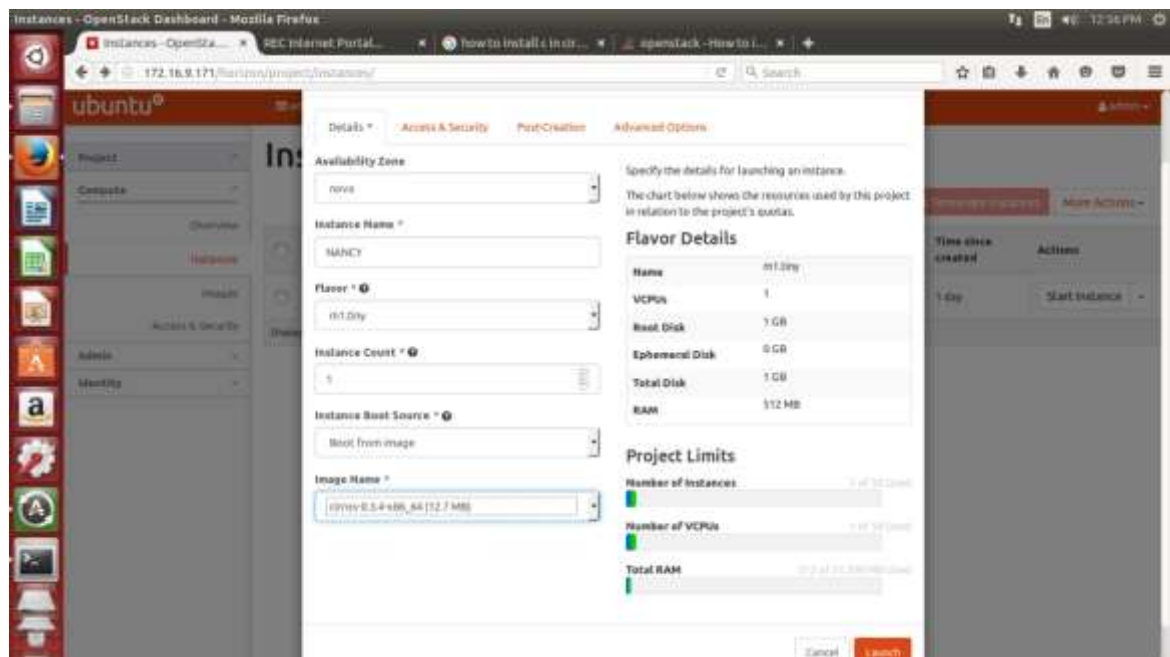
b) Creation, Management and Termination of Instances

Step: 1–Click Compute -> Instances -> Launch Instance



Step: 2 – On clicking Launch Instance, the following dialog appears, which shows number of Instances, VCPUs, and RAM.

- Give an instance name.
- Select Flavor and Instance Count based on requirement
- Select Boot source -> Boot from image
- Launch Instance.



Step: 3 – Instance in running condition

The screenshot displays the OpenStack Horizon dashboard interface. The top navigation bar includes the 'ubuntu' logo and a 'admin' user profile. The left sidebar contains a navigation menu with options like Project, Admin, System, Overview, Hypervisors, Host Aggregate, Instances (highlighted), Flavors, Images, Defaults, Metadata Definitions, System Information, and Identity. The main content area is titled 'Instances' and features a table listing the current instances. A green notification box in the top right corner states: 'Success: The instance is preparing the live migration to host "student-Metro-39002"'. The table has columns for Project, Host, Name, Image Name, IP Address, Size, Status, Task, Power State, Time since created, and Actions. Two instances are listed: one in a 'Migrating' state and another in an 'Active' state, both with a 'Running' power state. The bottom of the image shows a Windows taskbar with various application icons and a system clock indicating 16:57 on 09-09-2019.

Project	Host	Name	Image Name	IP Address	Size	Status	Task	Power State	Time since created	Actions
admin	student-Metro-39002	CSE	cirros-0.3.4-x86_64	172.24.0.69	101 MB	Migrating	Migrating	Running	45 minutes	Edit instance
admin	student-Metro-39002	keystone	cirros-0.3.4-x86_64	172.24.0.63	101 MB	Active	None	Running	5 days, 22 hours	Edit instance

RESULT

Thus the procedure to install Single node OpenStack-devstack, Creation, Management and Termination of Instances is completed successfully.

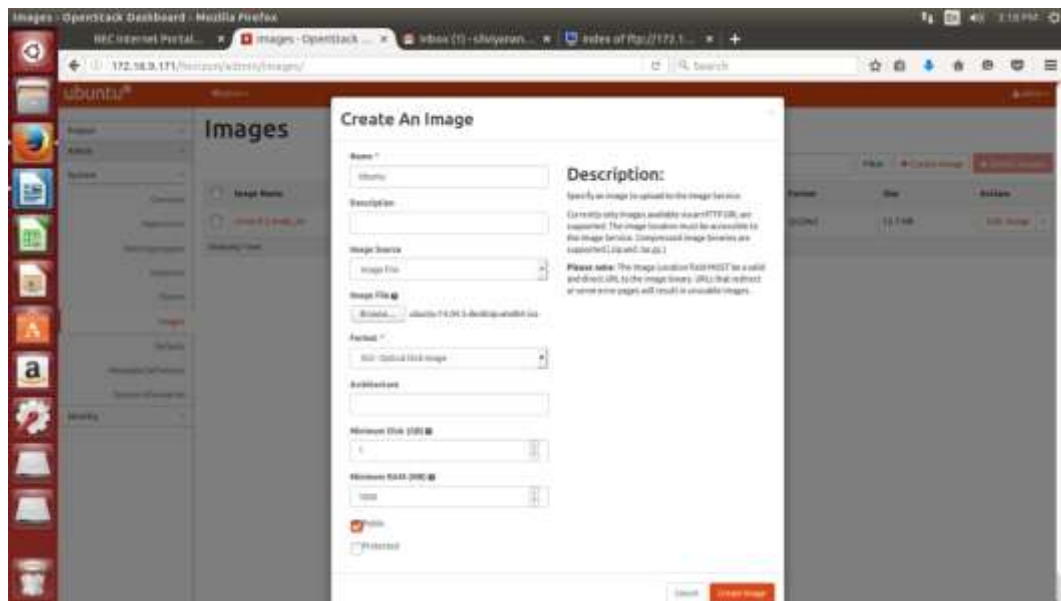
EX. NO: 9

DATE: OPENSTACK - ADD AND DELETE IMAGE AND VOLUME

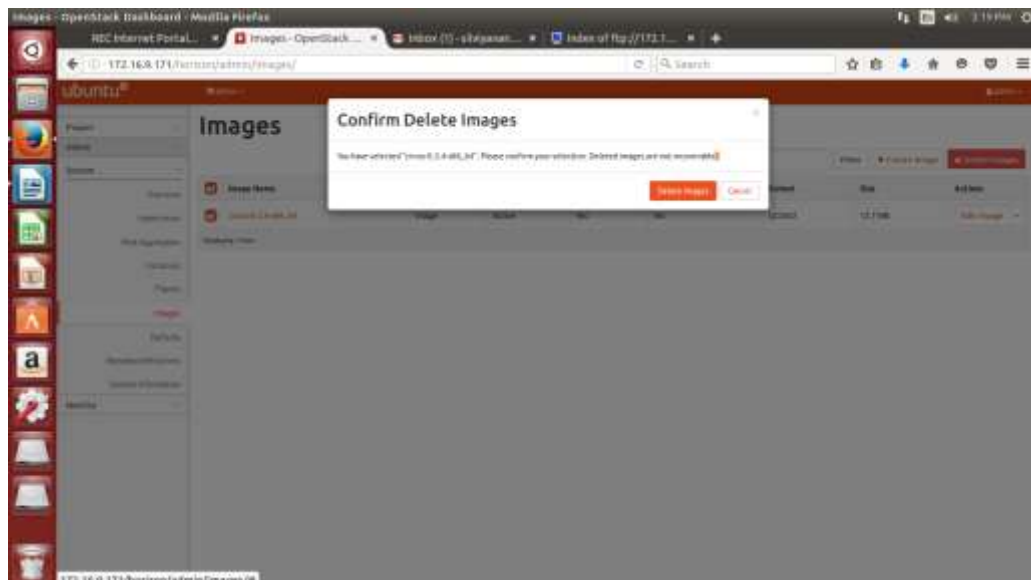
AIM

- a) To Upload and Deletion of images.
- b) Adding Volume

Step: 1 –Click System -> Images -> Create Image -> Add details.



Step: 2 - Click System -> Images -> delete Image.



b) Adding Volume

Step 1 - Log in to the dashboard.

Step 2 - Select the appropriate project from the drop down menu at the top left.

Step 3 - On the Project tab, open the Compute tab and click Volumes category.

Step 4 - Click Create Volume.

In the dialog box that opens, enter or select the following values.

Volume Name: Specify a name for the volume.

Description: Optionally, provide a brief description for the volume.

Volume Source: Select one of the following options:

- No source, empty volume: Creates an empty volume. An empty volume does not contain a file system or a partition table.
- Image: If you choose this option, a new field for Use image as a source displays. You can select the image from the list.
- Volume: If you choose this option, a new field for Use volume as a source displays. You can select the volume from the list. Options to use a snapshot or a volume as the source for a volume are displayed only if there are existing snapshots or volumes.

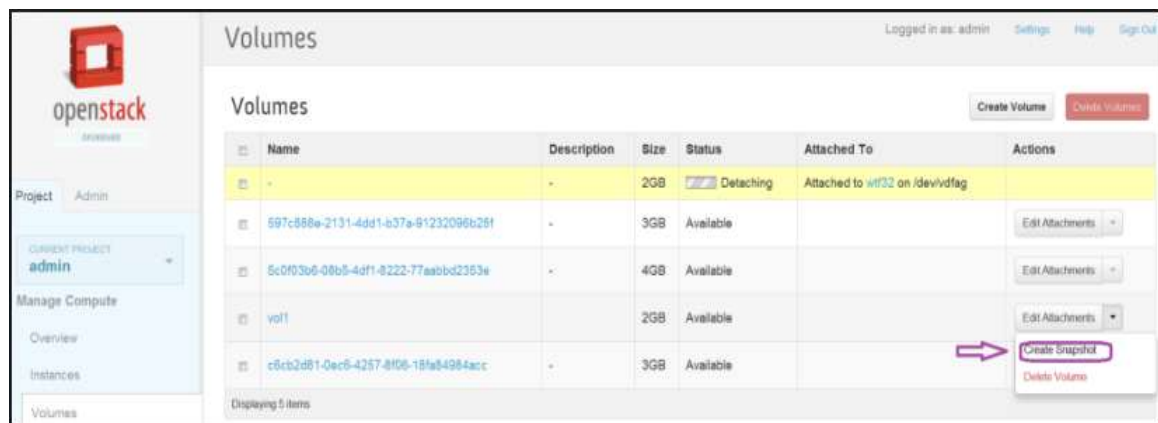
Type: Leave this field blank.

Size (GB): The size of the volume in gibibytes (GiB).

Availability Zone: Select the Availability Zone from the list. By default, this value is set to the availability zone given by the cloud provider (for example, us-west or apac-south). For some cases, it could be nova.

Step 5 - Click Create Volume.

The dashboard shows the volume on the Volumes tab.



RESULT

Thus the procedure to Upload and Deletion of images and adding a volume is completed successfully.

EX. NO: 10

DATE: STORAGE VIRTUALIZATION USING FREEBSD

AIM

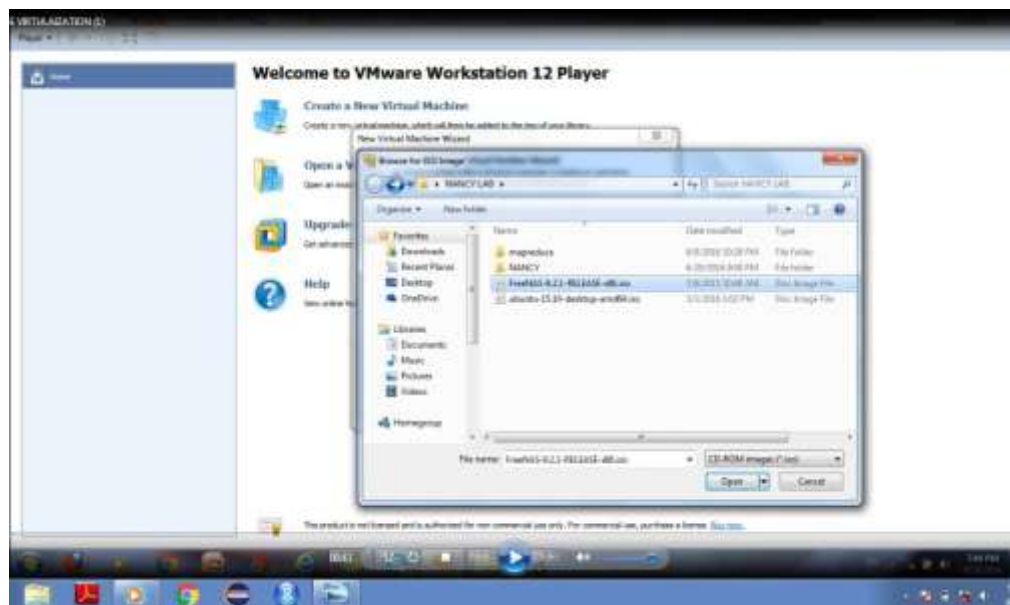
To install storage controller and interact with it using FreeBSD.

PROCEDURE

To install storage controller and interact with it using FreeBSD.

Step: 1 - Open VMware and select “Create a new Virtual Machine” and increase the RAM size to 4GB

Step: 2 - In the pop-up box select the installer disc image file (iso) option and browse the file FREENAS.iso and click next.



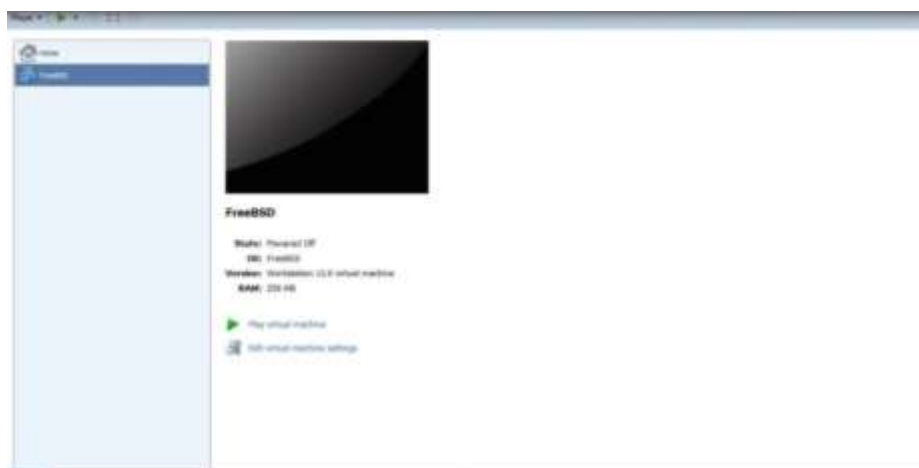
Step:3 - In the next page of the wizard, provide a name for the virtual machine and select the location at which it has to be installed and click next.



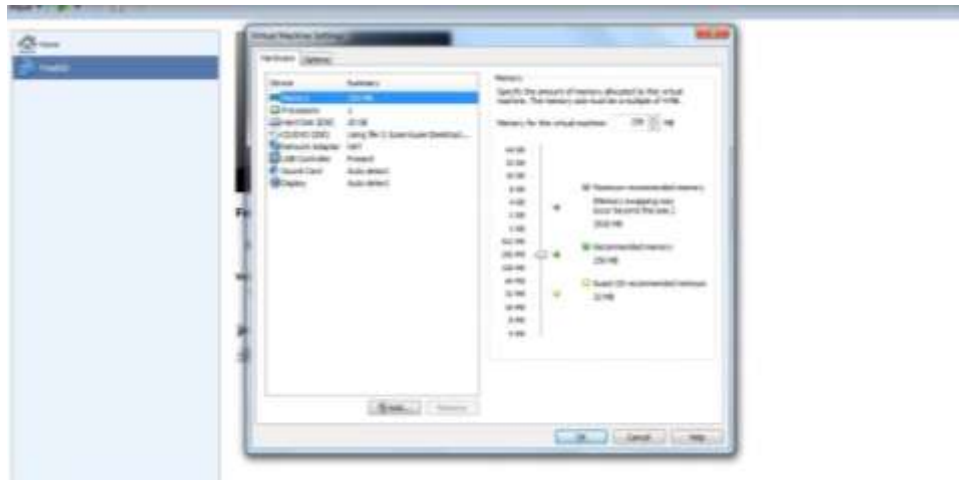
Step: 4 - In the next page, provide the maximum disc capacity as 20 GB and select store virtual disc as single file option and then click next. Then click finish.



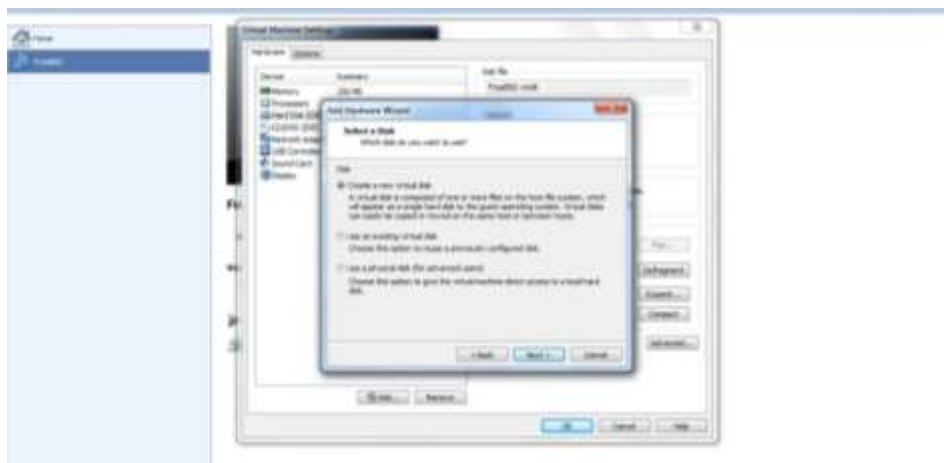
Step: 5 - A new virtual machine with the provided name would appear in the VMware home screen. Click on edit virtual machine settings.



Step: 6 - Select hard Disk from the virtual machine settings dialog box. Select add button. In the Add Hardware wizard select the hardware type as hard disk and click next. Select the virtual disk type as IDE and click next.

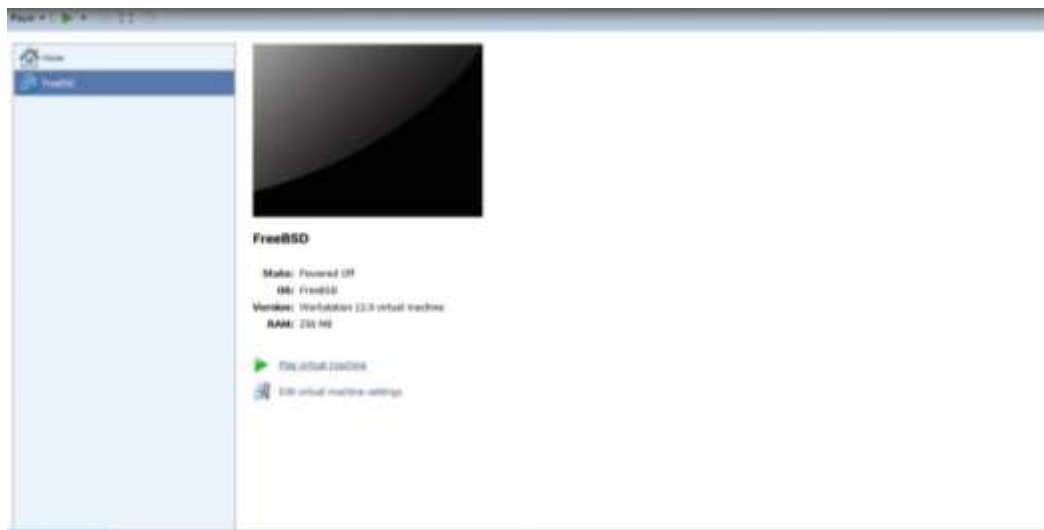


Step: 7 - In the next page select the Create a new virtual disk option and click next. Provide the disk capacity as 20GB and select store virtual disk as single file option and click next. Specify the disk file and click finish.



Step: 8 - Close the virtual machine settings dialog box.

Step: 9 - Select play virtual machine option available in the home screen of the VMware.



Step: 10- Proceed with the normal installation wizard of FREENAS. Wait until the FREENAS gets installed.

```

Root@RHEL6-Root7777:~# dd if=/dev/zero of=/dev/sda
1+0 records in
1+0 records out
512000 bytes transferred in 0.000000 seconds (512000000 bytes/sec)
Root@RHEL6-Root7777:~# fdisk /dev/sda
fdisk: warning: 1.44TiB partition table detected, but will not use it.
Command (m for help): n

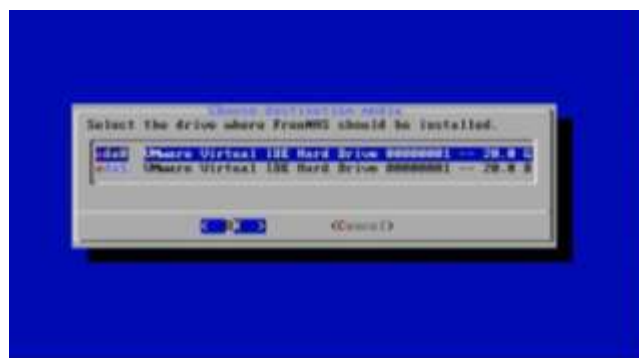
Partition type:
   0 = primary   1 = extended   2 = logical
   3 = raid      4 = unknown   5 = extended LBA
   6 = reserved  7 = reserved
Selected: 0
Partition number (1-4): 1
Starting point for partition (1-1024): 1
Ending point for partition (1-1024): 1024
Partition 1: type 0, starting at 1, ending at 1024, size 1024 sectors (512 KiB)
Command (m for help): n

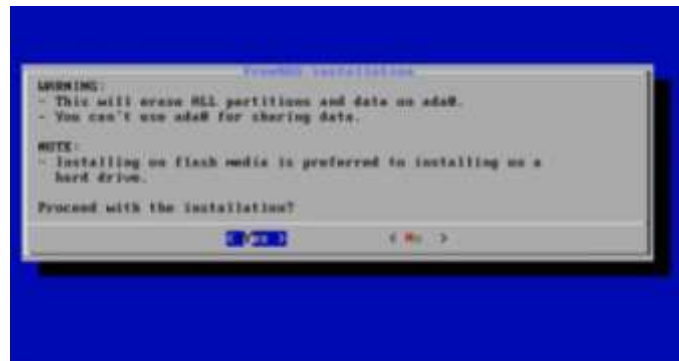
Partition type:
   0 = primary   1 = extended   2 = logical
   3 = raid      4 = unknown   5 = extended LBA
   6 = reserved  7 = reserved
Selected: 0
Partition number (1-4): 2
Starting point for partition (1-1024): 1024
Ending point for partition (1-1024): 2048
Partition 2: type 0, starting at 1024, ending at 2048, size 1024 sectors (512 KiB)
Command (m for help): n

Partition type:
   0 = primary   1 = extended   2 = logical
   3 = raid      4 = unknown   5 = extended LBA
   6 = reserved  7 = reserved
Selected: 0
Partition number (1-4): 3
Starting point for partition (1-1024): 2048
Ending point for partition (1-1024): 3072
Partition 3: type 0, starting at 2048, ending at 3072, size 1024 sectors (512 KiB)
Command (m for help): n

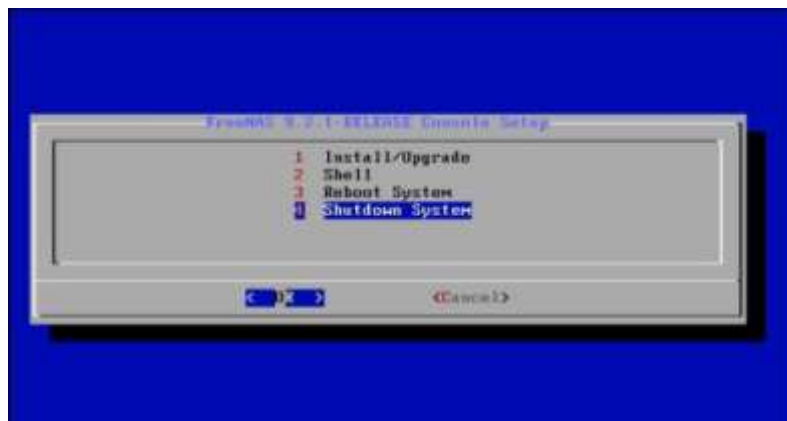
Partition type:
   0 = primary   1 = extended   2 = logical
   3 = raid      4 = unknown   5 = extended LBA
   6 = reserved  7 = reserved
Selected: 0
Partition number (1-4): 4
Starting point for partition (1-1024): 3072
Ending point for partition (1-1024): 4096
Partition 4: type 0, starting at 3072, ending at 4096, size 1024 sectors (512 KiB)
Command (m for help): w
The partition table is now stored in the MBR and sectors 1-4096 are now used.
Root@RHEL6-Root7777:~#

```

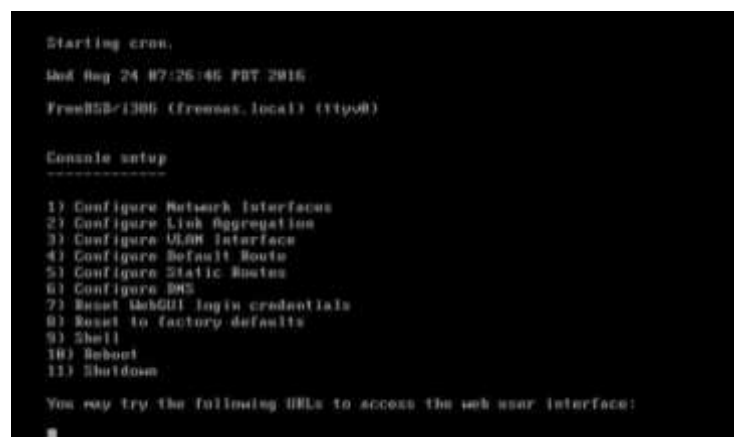




Step: 11 - After successful installation of FREENAS, reboot the VMware.



Step: 12– IP will be generated.



Step: 13- Open web Browser and type the IP address 192.168.170.128 to access the FREENAS dashboard. Enter the new password and log in.



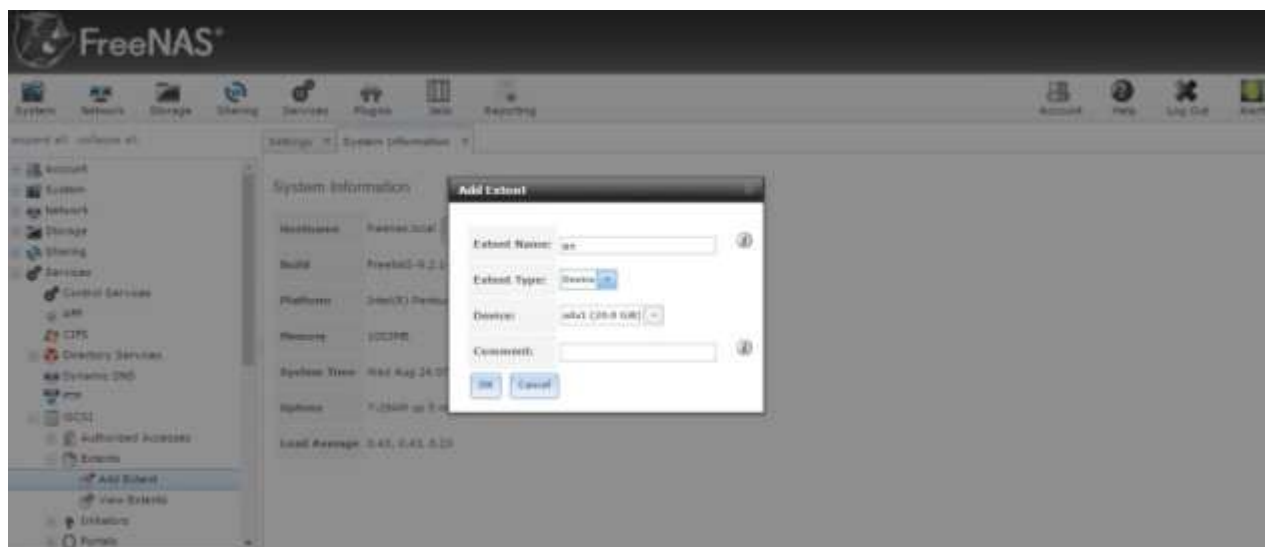
Step: 14- In the FREENAS dashboard page, find the menu services in the left pane.



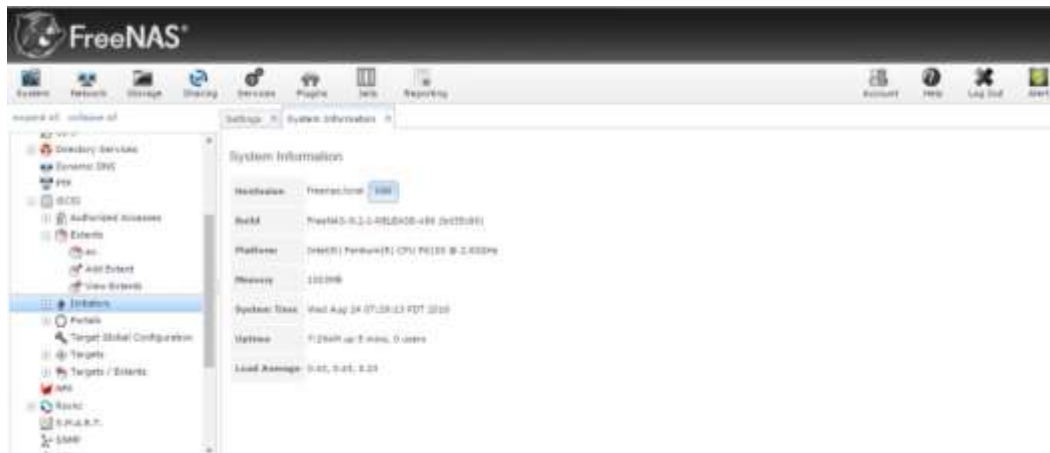
Step: 15- Select services->iscsi->Extents->Add extents.



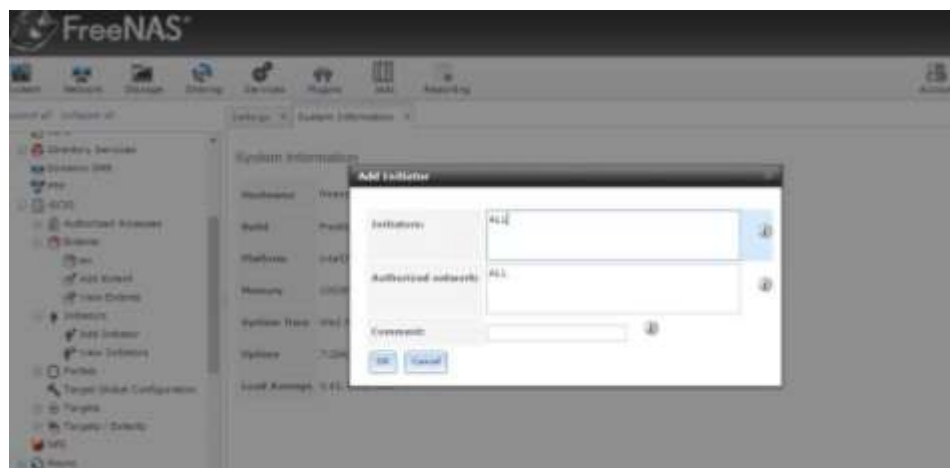
Step: 16 - In the Add Extent dialog box enter the extent name as en and select the extent type as Device and select ada1(20GB) from the device drop down list box and click ok.



Step: 17- Select services->iscsi->Initiators->Add initiators.



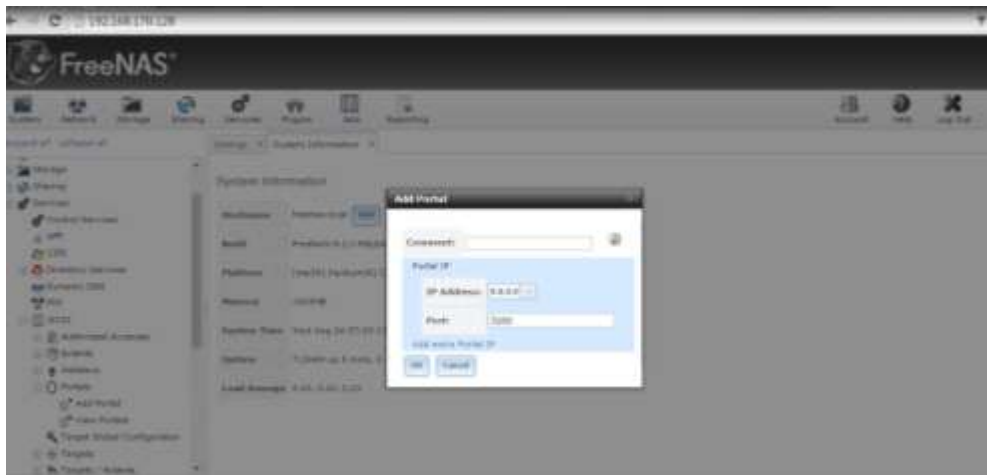
Step: 18- In the Add Initiators dialog box enter ALL in the initiators box and ALL in Authorised network box and click ok.



Step: 19- Select services->iscsi->Portals->Add portals.



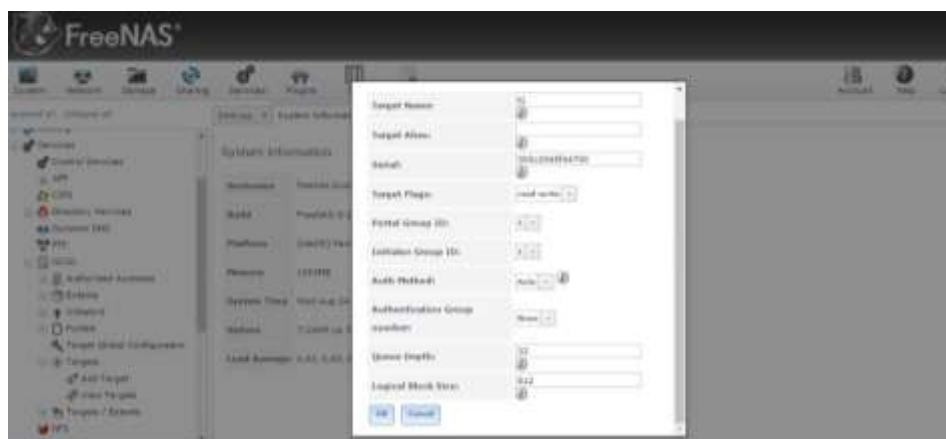
Step: 20- In the Add portal dialog box select 0.0.0.0 from the IP Address drop down list box and enter 3260 as Port then click ok.



Step: 21- Select services->iscsi->Targets->Add Target.



Step: 22- In the Add Target dialog box, Enter the target name as t1, select portal Group Id as 1 and Initiator group Id as 1 and then click ok.



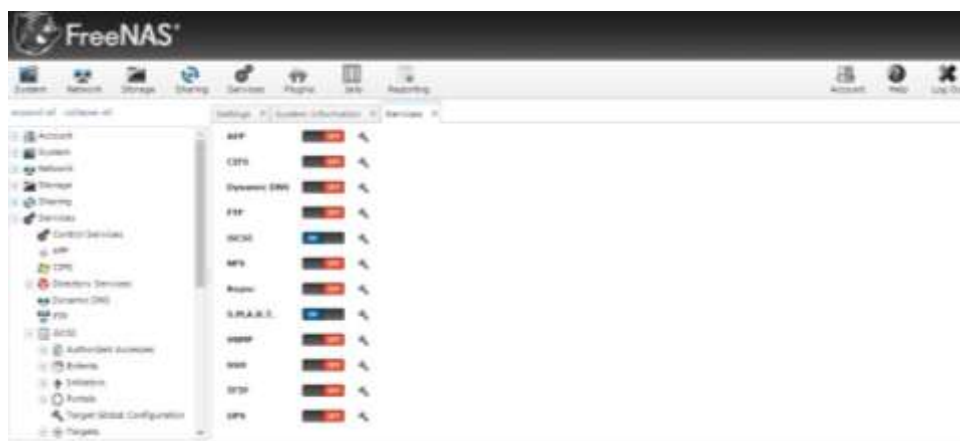
Step: 23- Select services->iscsi->Targets/Extents->Add Target/Extent.



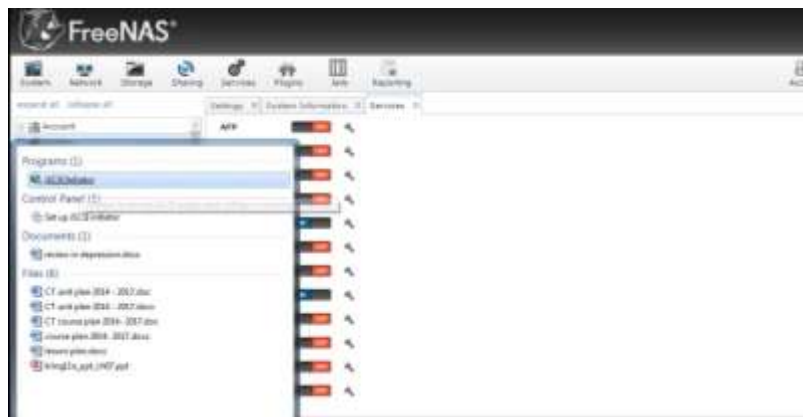
Step: 24- In the Add Target/Extent dialog box, Select t1 in the Target list box and en in the Extent list box and click ok.



Step: 25- From the menu bar select services and turn ON iSCSI and S.M.A.R.T.

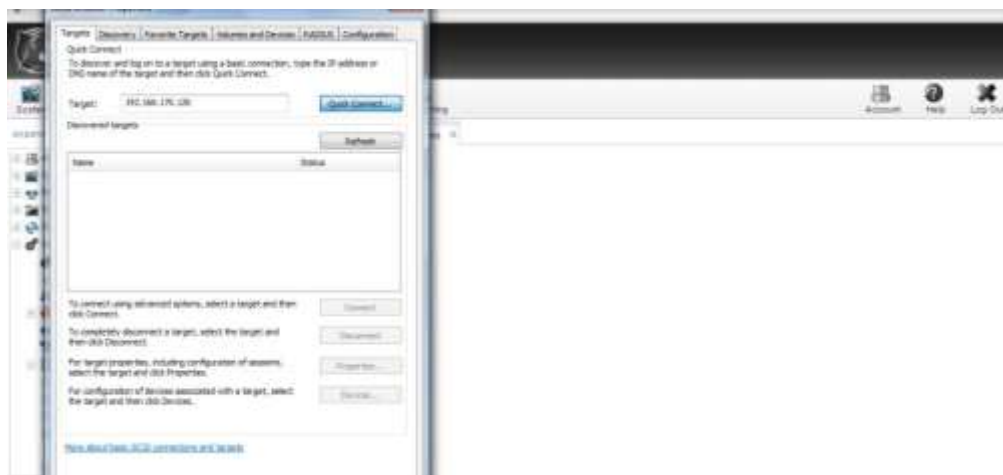


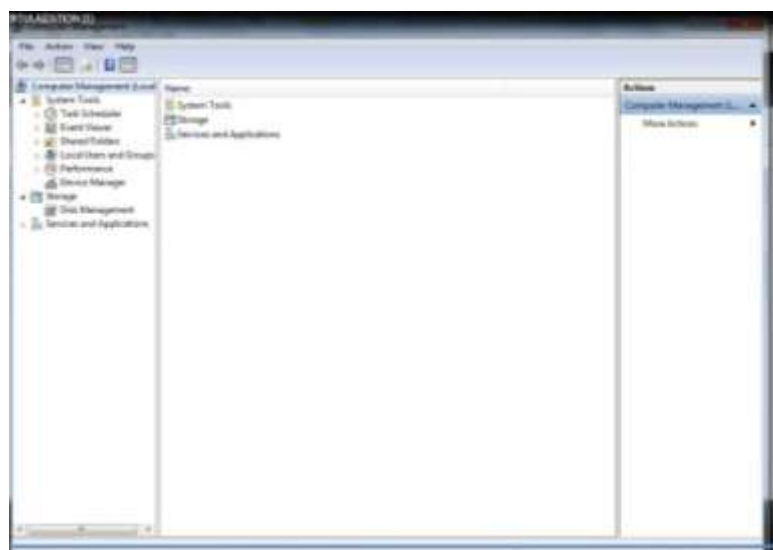
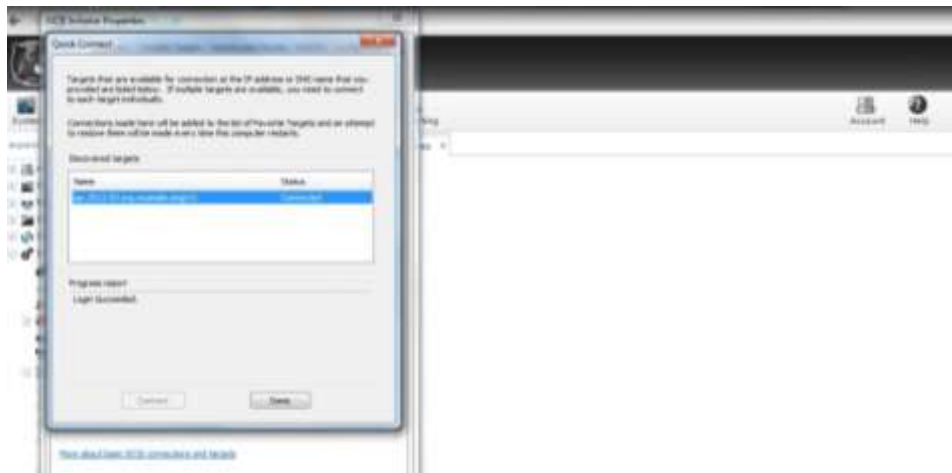
Step: 26- Search for the program iSCSI initiator and click yes in the pop-up box.



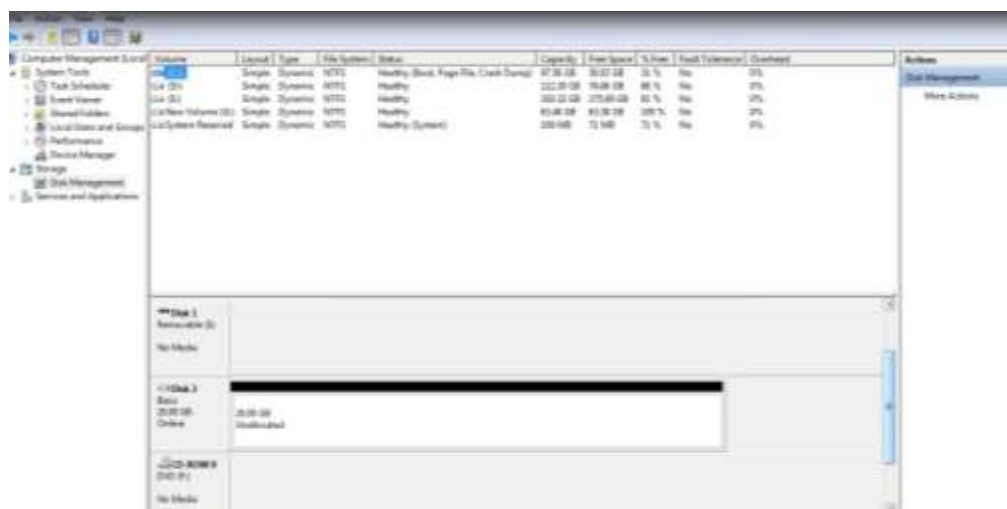
Step: 27- In the iSCSI initiator properties dialog box , enter the IP Address 192.168.170.128 in the Target box and click Quick connect. After successful connection the Quick connect dialog box appears with the status as connected. Click Done and close the iSCSI properties window.

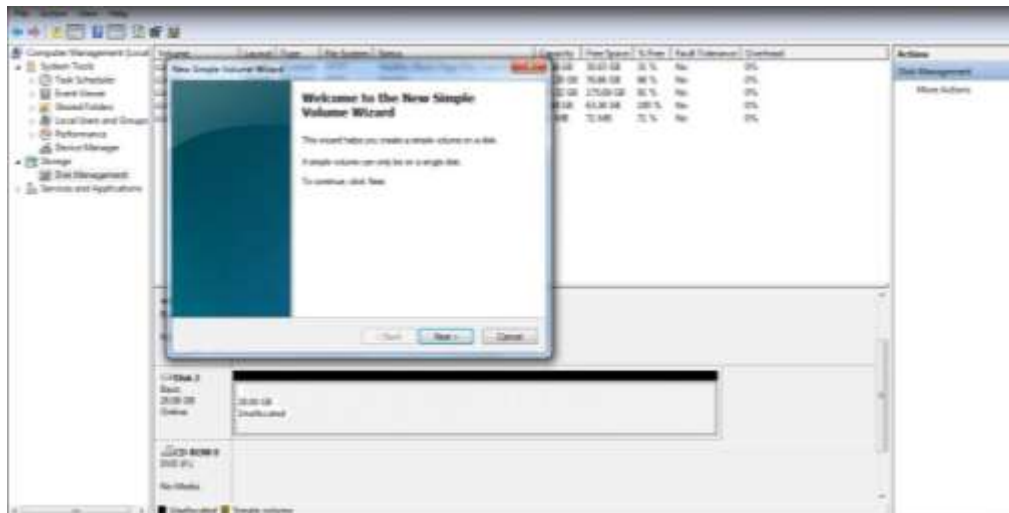
Step: 28- Open Computer Management by right clicking on My computer and selecting Manage. Select Disk Management from the right pane of Computer Management.



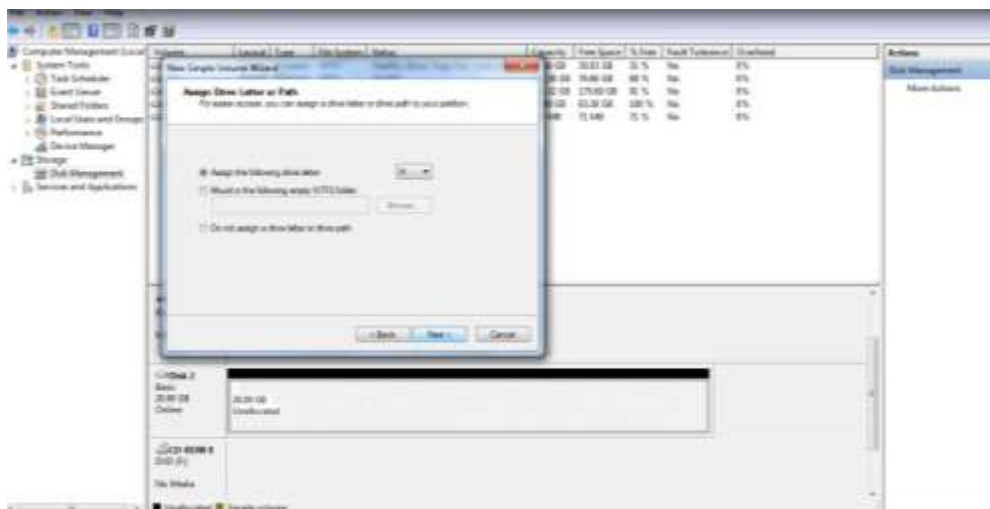


Step: 29- In the disk volume information page right click on the 20GB unallocated Disk and select New Simple Volume.

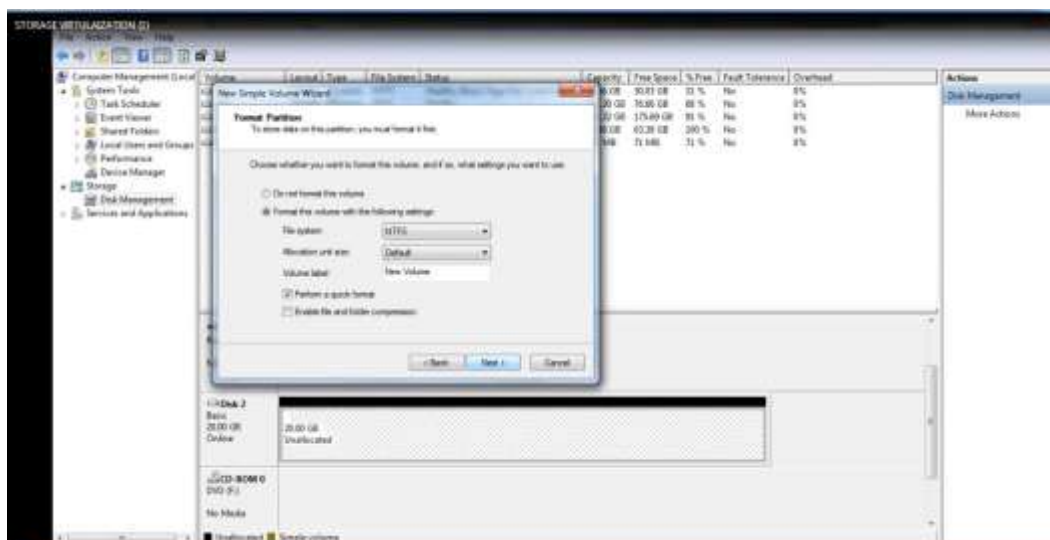


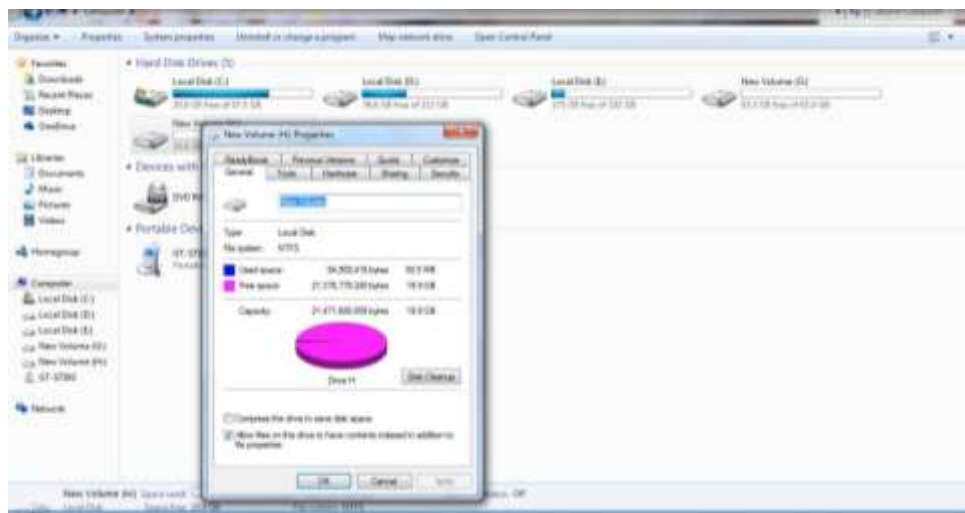
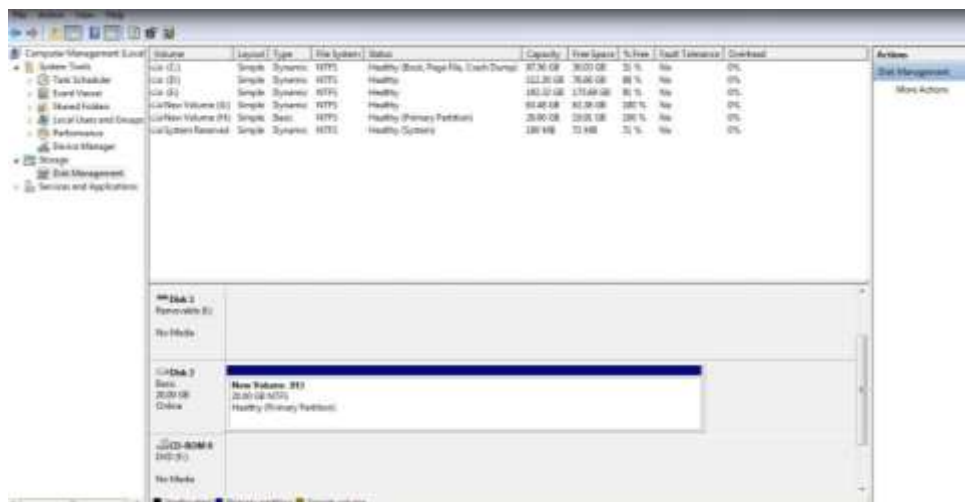
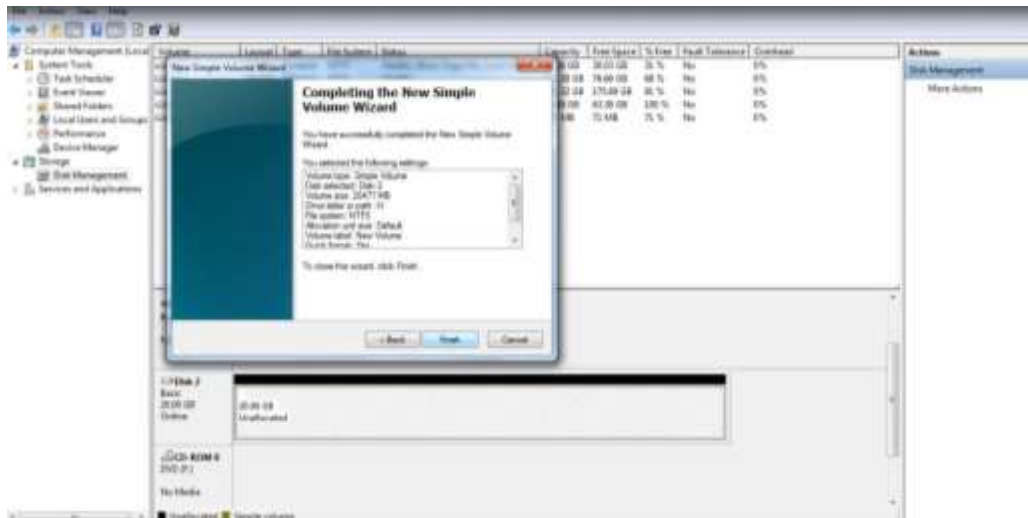


Step: 30- In the New Simple Volume wizard click next and select The Drive letter to be allocated as H and click next.



Step: 31- In the Format partition page select Format Partition with the following Settings option with file system as NTFS and check perform Quick format option and click next. Click finish.





RESULT

Thus the procedure to install storage controller and interaction is done successfully using FreeBSD.

BASIC UNDERSTANDING ON GRID COMPUTING

Grid Computing

Computational Grid is a collection of distributed, possibly heterogeneous resources which can be used as an ensemble to execute large-scale applications

- Computational Grid also called metacomputer
- Term computational grid comes from an analogy with the electric power grid:
 - Electric power is ubiquitous
 - Don't need to know the source (transformer, generator) of the power or the power company that serves it

Grid applications include

Distributed Supercomputing

- Distributed Supercomputing applications couple multiple computational resources - supercomputers and/or workstations
- Distributed supercomputing applications include SFExpress (large-scale modeling of battle entities with complex interactive behavior for distributed interactive simulation), Climate Modeling (modeling of climate behavior using complex models and long time-scales)

High-Throughput Applications

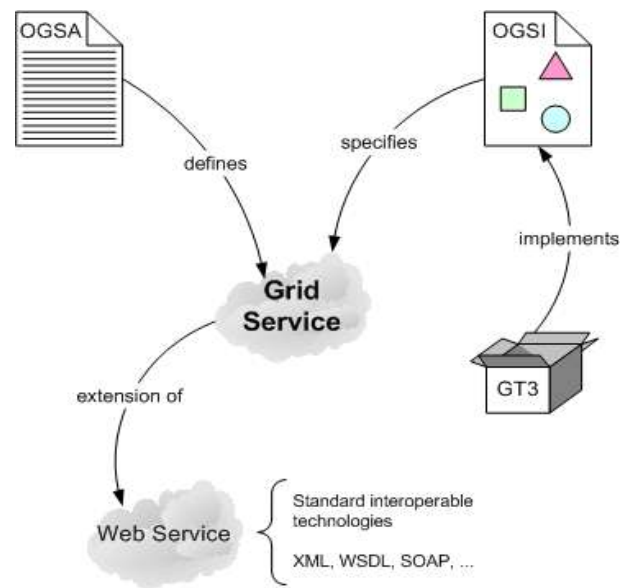
- Grid used to schedule large numbers of independent or loosely coupled tasks with the goal of putting unused cycles to work
- High-throughput applications include RSA keycracking, seti@home (detection of extra-terrestrial communication)

Data-Intensive Applications

- Focus is on synthesizing new information from large amounts of physically distributed data
- Examples include NILE (distributed system for high energy physics experiments using data from CLEO), SAR/SRB applications, digital library applications

GLOBUS TOOLKIT

The Globus Toolkit is a software toolkit that allows us to program grid-based applications. The third and latest version of the toolkit is based on something called Grid Services. Before defining Grid Services, we're going to see how Grid Services are related to a lot of acronyms you've probably heard (OGSA, OGSF for example), but aren't quite sure what they mean exactly. The following diagram summarizes the major players in the Grid Service world:



Grid Service

Grid Services are defined by OGSA. The Open Grid Services Architecture (OGSA) aims to define a new common and standard architecture for grid-based applications. Right at the center of this new architecture is the concept of a Grid Service. OGSA defines what Grid Services are, what they should be capable of, what types of technologies they should be based on, but doesn't give a technical and detailed specification (which would be needed to implement a Grid Service).

Grid Services are specified by OGSi. The Open Grid Services Infrastructure is a formal and technical specification of the concepts described in OGSA, including Grid Services. The Globus Toolkit 3 is an implementation of OGSi. GT3 is a usable implementation of everything that is specified in OGSi (and, therefore, of everything that is defined in OGSA). Grid Services are based on Web Services. Grid Services are an extension of Web Services. Grid Services provide some cool features which are not available with standard Web Services.

1. Grid Services are Stateful unlike Web Services which are stateless.
2. Grid Services implement Factory Pattern. (Since you have already studied Patterns I won't describe it further).
3. Two implementation approaches: A Grid Service can be implemented either by inheriting from a skeleton class or by using a delegation model, where incoming calls are delegated to a series of classes called operation providers.
4. Lifecycle management: Grid Services provide the necessary tools, such as callback functions during special moments in a Grid Service's lifetime (creation time, destruction time, etc.), to effectively manage its lifecycle (for example, to make Grid Services persistent).
5. Service Data: A Grid Service can have a set of associated service data that describes it. This is not to be confused with WSDL, which describes details like methods, protocols, etc. Service data is particularly useful to index Grid Services according to their characteristics and capabilities.
6. Notifications: We can configure a Grid Service to be a notification source, and certain clients to be notification sinks (or subscribers). This means that if a change occurs in the Grid Service, that change is notified to all the subscribers.