SRM VALLIAMMAI ENGINEERING COLLEGE
(An Autonomous Institution)
SRM Nagar, Kattankulathur-603203

DEPARTMENT OF INFORMATION TECHNOLOGY

ACADEMIC YEAR: 2025-2026
ODD SEMESTER

(REGULATION - 2019)

1908705 – FOSS AND CLOUD COMPUTING LABORATORY

LAB MANUAL

SEVENTH SEMESTER

B.Tech-Information Technology

Prepared By
Dr. D. Sridevi , Associate Professor /IT

Ms. D. Nisha, Assistant Professor (Sr.G) / IT

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

1. To afford the necessary background in the field of Information Technology to deal with engineering problems to excel as engineering professionals in industries.
2. To improve the qualities like creativity, leadership, teamwork and skill thus contributing towards the growth and development of society.
3. To develop ability among students towards innovation and entrepreneurship that caters to the needs of Industry and society.
4. To inculcate and attitude for life-long learning process through the use of information technology sources.
5. To prepare then to be innovative and ethical leaders, both in their chosen profession and in other activities.

PROGRAMME OUTCOMES (POs)

After going through the four years of study, Information Technology Graduates will exhibit ability to:

PO#	Graduate Attribute	Programme Outcome
1	Engineering knowledge	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems.
2	Problem analysis	Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3	Design/development of solutions	Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations.
4	Conduct investigations of complex problems	Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5	Modern tool usage	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modeling to complex engineering activities, with an understanding of the limitations.
6	The engineer and society	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7	Environment and sustainability	Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8	Ethics	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
9	Individual and team work	Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings
10	Communication	Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations,and give and receive clear instructions
11	Project management and finance	Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments
12	Life-long learning	Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

PROGRAM SPECIFIC OUTCOMES (PSOs)

By the completion of Information Technology program the student will have following Program specific outcomes

1. Design secured database applications involving planning, development and maintenance using state of the art methodologies based on ethical values.

2. Design and develop solutions for modern business environments coherent with the advanced technologies and tools.

3. Design, plan and setting up the network that is helpful for contemporary business environments using latest hardware components.

4. Planning and defining test activities by preparing test cases that can predict and correct errors ensuring a socially transformed product catering all technological needs

OBJECTIVES

❖ To learn and use version control systems

❖ To develop web applications in cloud

❖ To learn and work with virtual machine

❖ To learn the design and development process involved in creating a cloud based application

❖ To learn to implement and use parallel programming using Hadoop

LIST OF EXPERIMENTS

1. Use gcc to compile c-programs. Split the programs to different modules and create an application using make command.

2. Use version control systems command to clone, commit, push, fetch, pull, checkout, reset, and delete repositories.

3. Install Virtual box/VMware Workstation with different flavours of linux or windows OS on top of windows7 or8.

4. Install a C compiler in the virtual machine created using virtual box and execute Simple Programs

5. Install Google App Engine. Create hello world app and other simple web applications using python/java.

6. Use GAE launcher to launch the web applications.

7. Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.

8. Find a procedure to transfer the files from one virtual machine to another virtual machine.

9. Find a procedure to launch virtual machine using try stack (Online Open stack Demo Version)

10. Install Hadoop single node cluster and run simple applications like word count.

TOTAL: 45 PERIODS

OUTCOME

At the end of the course, learners will be able to:

- ❖ Configure various virtualization tools such as Virtual Box, VMwareworkstation.
- ❖ Design and deploy a web application in a PaaS environment.
- ❖ Learn how to simulate a cloud environment to implement new schedulers.
- ❖ Install and use a generic cloud environment that can be used as a private cloud.
- ❖ Install and use Hadoop.

LIST OF EQUIPMENT FOR A BATCH OF 30 STUDENTS

HARDWARE

- ✓ Standalone desktops 30 Nos.

SOFTWARE

- ✓ **Openstack**
- ✓ **Hadoop**
- ✓ **Eucalyptus or Open Nebula or equivalent**

COURSE OUTCOMES

1908705.1	Configure various virtualization tools such as Virtual Box, VMware workstation
1908705.2	Design and deploy a web application in a PaaS environment.
1908705.3	Learn how to simulate a cloud environment to implement new schedulers.
1908705.4	Install and use a generic cloud environment that can be used as a private cloud.
1908705.5	Install and use Hadoop

CO-PO MATRIX

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
1908705.1	3				3							
1908705.2	2		3		3							
1908705.3	1			2		2						
1908705.4			3		3							
1908705.5	1				2	2						
Average	**2**		**3**	**2**	**3**	**2**						

CO-PSO MATRIX

	PSO1	PSO2	PSO3	PSO4
1908705.1		2		
1908705.2		3	2	
1908705.3			2	
1908705.4		2	2	
1908705.5		2		
Average		**2**	**2**	

EVALUATION PROCEDURE FOR EACH EXPERIMENTS

S.No	Description	Mark
1.	Aim & Pre-Lab discussion	20
2.	Observation	20
3.	Conduction and Execution	30
4.	Output & Result	10
5.	Viva	20
	Total	**100**

INTERNAL ASSESSMENT FOR LABORATORY

S.No	Description	Mark
1.	Observation	10
2.	Record	5
3.	Model Test	5
	Total	**20**

SOFTWARE DESCRIPTION

✓ **Openstack**

OpenStack software controls large pools of compute, storage, and networking resources throughout a datacenter, managed through a dashboard or via the OpenStack API. OpenStack works with popular enterprise and open source technologies making it ideal for heterogeneous infrastructure.

✓ **Hadoop**

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

✓ **Eucalyptus or Open Nebula or equivalent**

OpenNebula is a cloud computing platform for managing heterogeneous distributed data center infrastructures. The OpenNebula platform manages a data center's virtual infrastructure to build private, public and hybrid implementations of infrastructure as a service.

HARDWARE

✓ Standalone desktops 30Nos.

Ex No: 1 INTRODUCTION TO LINUX OPERATING SYSTEM AND COMMANDS

Aim:

To study about Linux operating system and its commands

Description of Linux:

- Linux is a community of open-source Unix like operating systems that are based on the Linux Kernel. It initially released by **Linus Torvalds** on September 17, 1991. It is a free and open-source operating system and the source code can be modified and distributed to anyone commercially or non commercially under the GNU General Public License.

- Initially, Linux is created for personal computers gradually it is used in other machines like servers, mainframe computers, supercomputers, etc. Nowadays, Linux is also used in embedded systems like routers, automation controls, televisions, digital video recorders, video game consoles, smart watches, etc.

- The biggest success of Linux is Android(operating system) it is based on the Linux kernel that is running on smartphones and tablets. Due to android Linux has the largest installed base of all general-purpose operating systems. Linux is generally packaged in a Linux distribution.

1. **Kernel:** Kernel is the core of the Linux based operating system. It virtualizes the common hardware resources of the computer to provide each process with its virtual resources. This makes the process seem as it is the sole process running on the machine. The kernel is also responsible for preventing and mitigating conflicts between different processes. Different types of the kernel are:
   - Monolithic Kernel
   - Hybrid kernels
   - Exo kernels
   - Microkernels

2. **System Library:** It is the special types of functions that are used to implement the functionality of the operating system.

3. **Shell:** It is an interface to the kernel which hides the complexity of the kernel's functions from the users. It takes commands from the user and executes the kernel's functions.
4. **Hardware Layer:** This layer consists all peripheral devices like RAM/ HDD/ CPU etc.
5. **System Utility:** It provides the functionalities of an operating system to the user.

Advantages of Linux

- The main advantage of Linux, is it is an open-source operating system, means the source code is easily available for everyone and you are allowed to contribute, modify and distribute the code to anyone without any permissions.
- In terms of security, Linux is more secure than any other operating system. It does not mean that Linux is 100 percent secure it has some malware for it but is less vulnerable than any other operating system. So, it does not require any anti-virus software.
- The software updates in Linux are easy and frequent.
- Various Linux distributions are available so that you can use them according to your requirements or according to your taste.
- Linux is freely available to use on the internet.
- It has large community support.
- It provides high stability. It rarely slows down or freezes and there is no need to reboot it after a short time.
- It maintain the privacy of the user.
- The performance of the Linux system is much higher than other operating systems. It allows a large number of people to work at the same time and it handles them efficiently.
- It is network friendly.
- The flexibility of Linux is high. There is no need to install a complete Linux suit you are allowed to install only required components.
- Linux is compatible with a large number of file formats.
- It is fast and easy to install from the web. It can also install in any hardware even in your old computer system.
- It performs all tasks properly even if it has limited space on the hard disk.

Disadvantages of Linux

- It is not much user-friendly. So, it may be confusing for beginners.
- It has small peripheral hardware drivers as compared to windows.

Linux Basic Commands

Let's start with some simple commands.

1) **Pwd** command

'pwd' command prints the absolute path to current working directory.

$ **pwd**

/home/Raghu

2) **cal** command

Displays the calendar of the current month.

$ **cal**

July 2012

Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 7

8 9 10 11 12 13 14

15 16 17 18 19 20 21

22 23 24 25 26 27 28

29 30 31

'cal ' will display calendar for the specified month and year.

$ **cal 08 1991**

August 1991

Su Mo Tu We Th Fr Sa

1 2 3

4 5 6 7 8 9 10

11 12 13 14 15 16 17

18 19 20 21 22 23 24

25 26 27 28 29 30 31

3) **echo** command

This command will echo whatever you provide it.

$ **echo** "linoxide.com"

linoxide.com

The 'echo' command is used to display the values of a variable. One such variable is 'HOME'.
To check the value of a variable precede the variable with a $ sign.

$ **echo $HOME**

/home/raghu

4) **date** command
Displays current time and date.
$ date
Fri Jul 6 01:07:09 IST 2012
If you are interested only in time, you can use 'date +%T' (in hh:mm:ss):
$ date +%T
01:13:14
5) tty command
Displays current terminal.
$ tty
/dev/pts/0
6) who am icommand
This command reveals the user who is currently logged in.

$ who am i
raghu

7) id command
This command prints user and groups (UID and GID) of the current user.

$ id
uid=1000(raghu) gid=1000(raghu)
groups=1000(raghu),4(adm),20(dialout),24(cdrom),46(plugdev),112(lpadmin),120(admin),122(s
ambashare)
By default, information about the current user is displayed. If another username is provided as an
argument, information about that user will be printed:

$ id root
uid=0(root) gid=0(root) groups=0(root)

8) clear command
This command clears the screen.

Help command
Nobody can remember all the commands. We can use help option from command like
9) help option
With almost every command, '--help' option shows usage summary for that command.

$ date --help

Usage: date [OPTION]... [+FORMAT] or: date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]] Display the current time in the given FORMAT, or set the system date.

10) What is command

This command gives a one line description about the command. It can be used as a quick reference for any command.

$ whatis date

date (1) - print or set the system date and time

$ whatis whatis

whatis (1) - display manual page descriptions

11) **Manual** Pages

'--help' option and 'whatis' command do not provide thorough information about the command. For more detailed information, Linux provides man pages and info pages. To see a command's manual page, man command is used.

$ **man date**

The man pages are properly documented pages. They have following sections:

NAME: The name and one line description of the command.

SYNOPSIS: The command syntax.

DESCRIPTION: Detailed description about what a command does.

OPTIONS: A list and description of all of the command's options.

EXAMPLES: Examples of command usage.

FILES: Any file associated with the command.

AUTHOR: Author of the man page

REPORTING BUGS: Link of website or mail-id where you can report any bug.

SEE ALSO: Any commands related to the command, for further reference.

With -k option, a search through man pages can be performed. This searches for a pattern in the name and short description of a man page.

$ **man -k gzip**

gzip (1) - compress or expandfiles

lz (1) - gunzips and shows a listing of a gzip'd tar'd archive

tgz (1) - makes a gzip'd tararchive

uz (1) - gunzips and extracts a gzip'd tar'd archive

zforce (1) - force a '.gz' extension on all gzip files

12) Info pages

Info documents are sometimes more elaborate than the man pages. But for some commands, info pages are just the same as man pages. These are like web pages. Internal links are present within the info pages. These links are called nodes. Info pages can be navigated from one page to another through these nodes.

$ info date

Linux File system commands

13) Changing Directories Command

$ **cd [path-to-directory]**

Change the current working directory to the directory provided as argument. If no argument is given to 'cd', it changes the directory to the user's home directory. The directory path can be an absolute path or relative to current directory. The absolute path always starts with /. The current directory can be checked with 'pwd' command (remember?):

$ **pwd**

/home/raghu

$ cd /usr/share/

$ pwd

/usr/share

$ cd doc

$ pwd

/usr/share/doc

In the first 'cd' command, absolute path (/usr/share) is used, and with second command, relative path (doc) is used.

14) Listing File And Directories Command

$ **ls [files-or-directories]**

List files and/or directories.

If no argument is given, the contents of current directory are shown.

$ ls

example file1.txt file2.txt file3.txt

If a directory is given as an argument, files and directories in that directory are shown.

$ ls /usr

bin games include lib lib64 local sbin share src

'ls -l' displays a long listing of the files.

$ ls -l

total 4

drwxr-xr-x 2 raghu raghu 4096 2012-07-06 12:52 example

-rw-r--r-- 1 raghu raghu 0 2012-07-06 12:52 file1.txt

In this long listing, the first character is 'd' or '-'. It distinguishes between file types. The entries with a '-' (dash) are regular files, and ones with 'd' are directories. The next 9 characters are permissions ('rwxr-xr-x' in first listing). The number following the permissions is the link count.

$ ls -la odesk

total 16

drwxr-xr-x 4 raghu raghu 4096 2012-07-06 13:46 .

drwxr-xr-x 11 raghu raghu 4096 2012-07-06 13:15 ..

If you want to see the properties of a directory instead of the files contained in it, use -d (with -l) option:

$ ls -ld odesk/

drwxr-xr-x 4 raghu raghu 4096 2012-07-06 13:46 odesk/

Creating files and directories

15) mkdircommand

To create a directory, the 'mkdir' command is used.

$ **mkdir**example

$ ls -l

total 4

drwxr-xr-x 2 raghu raghu 4096 2012-07-06 14:09 example

16) **touch**command

For creating an empty file, use the touch command.

$ **touch** file1 file2 file3

$ ls -l

total 4

drwxr-xr-x 2 raghu raghu 4096 2012-07-06 14:09 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1

If a file already exists, touch will update its time stamp. There are a lot of other methods to create a new file, e.g. using a text editor like vi or gedit, or using redirection. Here is an example of creating a file using redirection:

$ **ls -l /usr > usrlisting**
$ ls -l
total 8
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 14:09 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file2
A file named usrlisting is created in this example.
Copy, move and remove commands
17) **copy**command
$cp source destination

Copy files and directories. If the source is a file, and the destination (file) name does not exit, then source is copied with new name i.e. with the name provided as the destination.

$ cp usrlisting listing_copy.txt
$ ls -l
total 12
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 14:09 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20file1
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20file2

If the destination is a directory, then the file is copied with its original name in that directory.

$ **cp listing_copy.txt example/**
$ ls -l xample/
total 4
-rw-r--r-- 1 raghu raghu 491 2012-07-06 16:07 listing_copy.txt

Multiple files can also be copied, but in that case, the last argument will be expected to be a directory where all the files are to be copied. And the rest of the arguments will be treated as file names.

$ cp file1 file2 example/

$ **ls -l xample/**
total 4
-rw-r--r-- 1 raghu raghu 0 2012-07-06 16:10file1
-rw-r--r-- 1 raghu raghu 0 2012-07-06 16:10file2
-rw-r--r-- 1 raghu raghu 491 2012-07-06 16:07listing_copy.txt

If a directory is to be copied, then it must be copied recursively with the files contained in it. To copy a directory recursively, use -r option with 'cp'command:

$ **cp -r example /tmp/expertslogin**/
$ ls -l /tmp/expertslogin
total 4
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 16:12 example

18) **Move** command
$ mv source destination
Move files or directories. The 'mv' command works like 'cp' command, except that the original file is removed. But, the mv command can be used to rename the files (or directories).

$ **mv listing_copy.txt usrcopy**
$ **ls -l**
total 12
drwxr-xr-x 2 raghu raghu 4096 2012-07-06 16:10 example
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1

Here, 'listing_copy.txt' is moved with the name 'usrcopy' in the same directory (or you can say that it has been renamed).

19) To remove or Delete
$ **rmdir**
'rmdir' command removes any empty directories, but cannot delete a directory if a file is present in it. To use 'rmdir' command, you must first remove all the files present in the directory you wish to remove (and possibly directories if any).

To remove files and directories
$ **rm files|directories**

A directory must be removed recursively with -r option.
$ **rm** file2
$ **rm** -r example/

$ ls -l

total 8

-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20file1

-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20file3

Here, the file named 'file2' is removed first, and then the directory 'example' is removed recursively. This can be seen in the output of 'ls -l' command where these two are no longer present.

Other file commands

20) File command

The file command determines the file type of a given file. For example:

$ **file /etc/passwd**

/etc/passwd: ASCII text

You can provide one or more than one file as an argument to the file command.

$ file td.c td.out ARP.java Screenshot.png StringTokenizing.class

idl.rar List.pdf

td.c: ASCII C program text, with CRLF line terminators

td.out: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.15, not stripped

ARP.java: ASCII Java program text, with CRLF line terminators

Screenshot.png: PNG image data, 1366 x 768, 8-bit/color RGB, non-interlaced

StringTokenizing.class: compiled Java class data, version 50.0 (Java 1.6)

idl.rar: RAR archive data, v1d, os: Win32

List.pdf: PDF document, version 1.4

21) stat command

To check the status of a file. This provides more detailed information about a file than 'ls -l' output.

$ **stat usrcopy**

File: `usrcopy'

Size: 491 Blocks: 8 IO Block: 4096 regular file

Device: 808h/2056d Inode: 149452 Links: 1

Access: (0644/-rw-r--r--) Uid: (1000/ raghu) Gid: (1000/ raghu)

Access: 2012-07-06 16:07:06.413522009 +0530

Modify: 2012-07-06 16:02:30.204152386+0530

Change: 2012-07-06 16:17:18.992559654+0530

22) **cat** command

The 'cat' command is actually a concatenator but can be used to view the contents of a file.

$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh

23) pagers

The cat command lists file as a whole. But if the file is big enough to fit into one screen, then we will be able to see only the last page of the file. The commands 'less' and 'more' display files one page at a time. So they are also called pagers. You can navigate through a file using arrow keys. To quit from a pager, hit 'q'.

24) **head** command

Displays the first few lines of a file. By default, the 'head' command displays the first 10 lines of a file. But with -n option, the number of lines to be viewed can be specified.

$ **head /etc/passwd**
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh

25) **tail** command

Similar to 'head'; the 'tail' command shows the last 10 lines by default, and -n option is available aswell.

$ **tail -n 4 /etc/passwd**
raghu:x:1000:1000:Raghu Sharma,,,:/home/raghu:/bin/bash
sshd:x:113:65534::/var/run/sshd:/usr/sbin/nologin
dictd:x:114:123:Dictd Server,,,:/var/lib/dictd:/bin/false
mysql:x:115:124:MySQL Server,,,:/nonexistent:/bin/false

26) **wc**command
Word count
This command counts lines, words and letters of the input given to it.

$ **wc /etc/passwd**
35 57 1698 /etc/passwd

The /etc/passwd file has 35 lines, 57 words, and 1698 letters present in it.

27) **Grep** command
The 'grep' command searches for a pattern in a file (or standard input). It supports regular expressions. It returns a line if it matches the pattern in that line. So, if we wish to find the lines containing the word 'nologin', we use 'grep' as follows:

$ **grep nologin /etc/passwd**
sshd:x:113:65534::/var/run/sshd:/usr/sbin/nologin

28) **ln**command
The ln command is used in linux to create links. Links are a kind of shortcuts to other files. The general form of command is:

$ **ln** TARGET LINK_NAME

There are two types of links, soft links and hard links. By default, hard links are created. If you want to create soft link, use -s option. In this example, both types of links are created for the file usrlisting.

$ **ln usrlisting hard_link**

$ **ln -s usrlisting soft_link**

$ **ls -l**
total 12
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file1
-rw-r--r-- 1 raghu raghu 0 2012-07-06 14:20 file3
Text Editors
29) Pico &Nano
'Pico' is a text editor in Linux. 'Nano' editor is inspired from 'pico'. They work almost the same. If the argument given as filename exists, then that file will be opened for editing in pico/nano. Otherwise, a new file with that name will be created. Let's create a new file named hello.txt:

$ pico hello.txt
GNU nano 2.2.6 File: hello.txt Modified
This file is edited with pico editor.

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text^T To Spell

Having made all the changes to the file, press 'ctrl+o' to write the changes to the file and 'ctrl+x'
to exit from the editor. There are a lot of functions available with this editor. The help menu can
be accessed with 'ctrl+g' keystrokes.

30) VIeditor
The VI stands for Visual editor; another text editor in Linux. This is a standard editor in many
Linux/Unix environments. This is the default editor that comes with many Linux distributions. It
might be possible that it is the only text editor available with your distro.

You can open a file with vi for editing using the following:

$ **vi hello.txt**

The vi editor has 3 modes in which it performs its functions. The default is COMMAND mode,
in which tasks like copy, paste, undo etc can be performed. You can change a mode from
command mode only (and come back to it). The second mode is the INSERT mode, in which
whatever key you type is treated as a character and will be loaded into the file buffer. To enter
this mode, press 'i' when in command mode.
The final mode is EX mode or last line mode. The changes made in the buffer can be saved or
discarded in this mode.

Hello world.
This file is edited using vi editor.
"hello.txt" 2 lines, 50 characters

Useful commands
31) **Alias** command
The 'alias' is another name for a command. If no argument is given, it shows current aliases.
Aliases can be used for short names of commands. For example, you might use the clear
command frequently. You can create an alias for it:
$ **alias c="clear"**

Next time you enter 'c ' on command line, your screen will get clear. Current aliases can be
checked with 'alias' command:
$ **alias**

alias alert='notify-send --urgency=low -i "$([$? = 0]&& echo terminal || echo error)"
"$(history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[;&|]\s*alert$//'\'')"'
alias c='clear'
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alF'
alias ls='ls --color=auto'

32) **w** command
w command is used to check which users are logged in to the system, and what command they
are executing at that particular time:

$ **w**
10:06:56 up 57 min, 3 users, load average: 0.04, 0.06, 0.09
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
root tty1 10:06 28.00s 1.02s 0.67s pager-s
raghu tty7 :0 09:19 57:33 1:22 0.20s gnome-session --session=classic-gnome
raghu pts/0 :0.0 09:34 0.00s 0.78s 0.00sw

It also shows the uptime, number of users logged in and load average of the system (in the first
line of output above).

33) **last** command
Displays information about the users who logged in and out of the system. The output of the last
command can be very large, so the following output has been filtered (through head) to display
the top 10 lines only:

$ **last**| head
root tty1 Mon Jul 9 10:06 still logged in
root tty1 Mon Jul 9 10:06 - 10:06 (00:00)
raghu pts/1 :0.0 Mon Jul 9 10:05 - 10:06 (00:00)
A similar command is 'lastb' that shows the last unsuccessful login attempts. But this command
must be run as root otherwise you would get an error saying permission denied.

$ **lastb**
raghu tty2 Mon Jul 9 10:16 - 10:16 (00:00)
UNKNOWN tty2 Mon Jul 9 10:15 - 10:15 (00:00)

ubuntu tty8 :1 Mon Jul 2 10:23 - 10:23 (00:00)

btmp begins Mon Jul 2 10:23:54 2012

34) **du**command

The du command determines disk usage of a file. If the argument given to it is a directory, then it will list disk usage of all the files and directories recursively under that directory:

$ **du etc/passwd**
4 /etc/passwd

$ **du hello/**
52hello/HelloApp
4hello/orb.db/logs
20 hello/orb.db
108 hello/

35) dfcommand
The df reports file system usage. For example:

$ **df**
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/sda7 10079084 7372872 2194212 78% /
none 1522384 768 1521616 1% /dev
none 1529012 252 1528760 1% /dev/shm
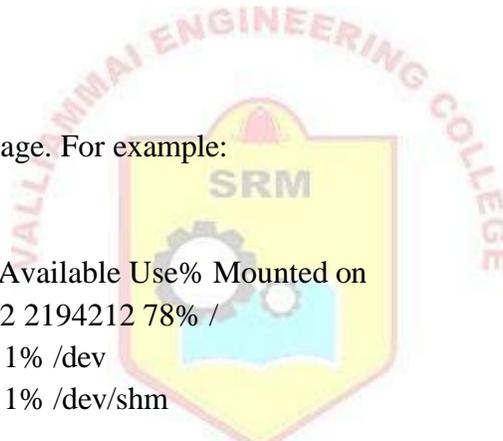
36) **fdisk**command

The fdisk is a tool for getting partition information, and for adding and removing partitions. The fdisk tool requires super user privileges. To list all the partitions of all the hard drives available:

$ **fdisk -l**

Disk /dev/sda: 320.1 GB, 320072933376 bytes
255 heads, 63 sectors/track, 38913 cylinders
Units = cylinders of 16065 * 512 = 8225280bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x396f396f

Device Boot Start End Blocks Id System

/dev/sda1 1 2611 20971520 7 HPFS/NTFS
/dev/sda2 2611 28720 209715200 7 HPFS/NTFS
/dev/sda3 * 28720 38914 81882113 5 Extended

The fdisk is an interactive tool to edit the partition table. It takes a device (hard disk) as an argument, whose partition table needs to be edited.

37) **netstat**command

The 'netstat' is a command used to check the network statistics of the system. It will list the current network connections, routing table information, interface statistics, masquerade connections and a lot more information.

$ **netstat | head**

Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags Type State I-Node Path
unix 13 [ ] DGRAM 8498 /dev/log
unix 2 [ ] DGRAM 6824 @/org/kernel/udev/udevd

38) historycommand

History command shows the commands you have entered on your terminal so far.

39) **passwd**command

To change your password with passwd command.

40) **Shutdown** Command

In Linux, you can use shutdown command to gracefully halt your system. Most commonly used command is shutdown -h now.

Viva questions

1. What is linux?
2. Name the come Creating files and directoriescommands
3. Explain about wordcount.
4. What is the use of head and tailcommands?
5. What ispwd?

Result:

Thus the introduction about Linux is studied.

Ex No: 2 BUILDING MODULAR PROGRAMS USING MAKE COMMAND

Aim:

To use gcc to compile c-programs. Split the programs to different modules and create an application using make command.

Procedure:

- A Monolithic Program
    - Consider this monolithic program Monolithic.c
    - Placing the code into one source file has a number of short comings:
        - It doesn't reflect the modularity of the design.
          There is no clear separation of the ADTs from each other and the main program.
        - There is no information hiding.
          The main program contains details of the Person Type representation which it doesn't need to know.
        - It doesn't scale. Storing a large program as a single file produces large files that are hard to navigate. Every time we make a change to one part, we are forced to recompile the whole program.
    - The mechanisms that enable us to reflect modularity in our code are *interface and implementation* and *separate compilation*.
- Interface and Implementation
    - A module's *interface* is a description of the data and functions that are visible to code external to the module. These descriptions are usually just the type signatures of the externally visible data and functions.
    - A module's *implementation* comprises the data structures and code that make the module work.
    - Separating the *interface* from the *implementation* is the way we reflect *abstraction*.
    - When we use a module, we need to know only what effects accessing the data and routines in its *interface* will have and we can ignore the details of how the module is *implemented*.
- A Modular Program
    - We can split the code above into 4 files as follows:
        - CString.h - defines the CString type
        - Person.h - defines the Person Typetype
        - PrettyPrint.h - defines the PRETTY_PRINTmacro
        - Person.c - implements the person manipulation
        - Modular.c – main program
    - This separates the interface of the ADT from the implementation.
    - The files can all be compiled together
        - gcc -o Modular Modular.cPerson.c
    - The "declare before main, define afterwards" style of programming makes the split easy to do later. Compare these two...

- ▪ Program Structure.c
- ▪ ProgramStructure2.c
- Separate Compilation

Program Structure for Multi-File Programs
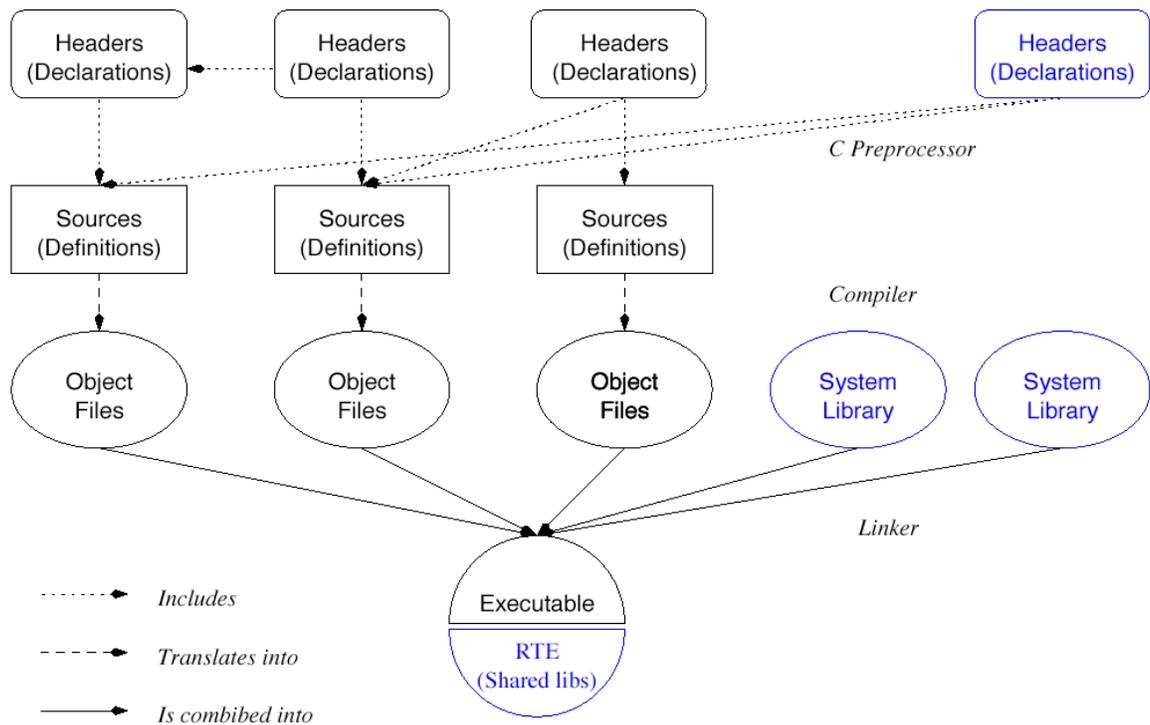
- o Separating the interface from the implementation means compiling multiple small files, and some code is compiled multiple times.
- o Some files need not be recompiled when changes are made.
- o The UNIX commands to compile and link separately are
  - ▪ gcc -c Source.c tocompile
  - ▪ gcc -o Executable Object1.o Object2.o ... to link
- o For the example above ... these are the file dependencies
- o The compiler has some options that support multi-file program development.
- make
  - o Large programs broken up may use many object and library files
  - o Changing a source file means updating the objects, libraries and executable that depend on the source file.
  - o The UNIX make utility keeps track of such dependencies, and recreates only the dependent files.
  - o To determine which source files are part of the *target* to be built, and which tools to use to build the target, make uses a file called Make file in the current directory.
    - ▪ A simple Make file for the modular program
    - ▪ Lines starting with a # are comments

- A Make file is a sequence of rules in the form
- target: dependencies
        action

    Note that action must be preceded by a tab character; this is a historical artifact that no one has bothered to fix.

- Actions are just standard UNIX commands that are fed to a shell
- If a dependency does not have a target entry, it is expected to exist as a result of external activity, e.g., editing

o Make files are processed from the top down, trying to create a target. By default the first target is used, but any target can be specified on the command line.

```
make(Target) {
   foreach Dependency do {
      if (IsATarget(Dependency)) {
         make(Dependency)
      }
   }
   if (!FileExists(Target) || AnyNewer(Dependencies,Target)) {
      DoActions(Target)
   }
}
```

o Make variables can hold values like environment variables. make automatically sets the value of certain variables like $@. Variables of this ilk include:
- $@ The file name of the target of the rule.
- $< The name of the first dependency.
- $^ The names of all the dependencies with spaces between them.
- $? The list of dependencies that are out of date.

o make has rules that it applies by default,specified in/usr/share/lib/make/make.rules. For example, make knows that to get from a .c file to a .o file, it must run the C compiler with the -c flag.
- Make file using defaults

o In addition to targets that build programs, make can be made to perform other housekeeping by specifying targets that are not necessarily programs. For example, a common target is

o clean:

o   rm -f *.o *.bak*~

    which causes make to delete object files and any backup files (*.bak and *~) created by other programs such as editors.

o makedepend ormkdep
- Examines each of the source files in its command line and looks for #include directives.

- Generates a list of source files that are needed by the compiler to produce the resulting target.
- Result is placed in .depend (FreeBSD) or appended to makefile(Linux)
    - o A more realistic Makefile for the example
- Creating alibrary
    - o .o files can be combined into libraries using the arutility.
    - o Library files are given a .aextension.
    - o The object files can be used by the compiler by specifying the location ofthe library file with the -L flag, and giving the library filename.
    - o Example
    - o prompt> gcc -c*.c
    - o prompt> ar -rv libmystuff.a*.o
    - o prompt>pwd
    - o /home/geoff/c
    - o prompt> gcc MyMain.c -L/home/geoff/c -lmystuff -o MyMain

Exercises

- Below is the content of a make file. Assume that the files part1.c and part3.c have just been modified. Write down the sequence of command activations in the correct order when make is run.

        all: part1.o part2.o
                    gcc part1.o part2.o -o whole

        part1.o: part1.c part3.o
                    gcc -c part1.c
                    gcc part1.o part3.o -o part3.out

        part2.o: part2.c
                    gcc -c part2.c

        part3.o: part3.c
                    gcc -c part3.c

Program

Monolithic.c

//
#include<stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_CSTRING 128

```c
typedef char CString[MAX_CSTRING];

typedef struct {
   CStringFamilyName;
   CStringGivenName;
} PersonType;

#definePRETTY_PRINT(This)printf("\n---------------\n%s\n------------------\n",(This))

//-----------------------------------------------------------------------------------------
void SetNames(CString NewFamilyName,CString NewGivenName,PersonType *APerson) {

   strcpy(APerson->FamilyName,NewFamilyName);
   strcpy(APerson->GivenName,NewGivenName);
}
//-----------------------------------------------------------------------------------------
void GetFullName(PersonType APerson,CString FullName) {

   strcpy(FullName,APerson.GivenName);
   strcat(FullName," ");
   strcat(FullName,APerson.FamilyName);
}
//-----------------------------------------------------------------------------------------
int main(void) {

   PersonType MyPerson;
   CStringInputFamilyName;
   CString InputGivenName;
   CString FullName;
   CStringOutputLine;

   printf("Please enter the given name and family name : ");
   scanf(" %s %s",InputGivenName,InputFamilyName);
   SetNames(InputFamilyName,InputGivenName,&MyPerson);
   GetFullName(MyPerson,FullName);
   sprintf(OutputLine,"The full name is %s",FullName);
   PRETTY_PRINT(OutputLine);
   return(EXIT_SUCCESS);
}
//-----------------------------------------------------------------------------------------
CString.h

#ifndef CSTRING_H
#define CSTRING_H
//-----------------------------------------------------------------------------------------
#define MAX_CSTRING 128
```

```
typedef char CString[MAX_CSTRING];
//----------------------------------------------------------------------------------------------
#endif



Person.h

#ifndef PERSON_H
#define PERSON_H
//----------------------------------------------------------------------------------------------
#include "CString.h"

typedef struct {
   CStringFamilyName;
   CStringGivenName;
} PersonType;
//----------------------------------------------------------------------------------------------
void SetNames(CString NewFamilyName,CString NewGivenName,PersonType *APerson);
void GetFullName(PersonType APerson,CString FullName);
//----------------------------------------------------------------------------------------------
#endif
\

PrettyPrint.h

#ifndef PRETTYPRINT_H
#define PRETTYPRINT_H
//----------------------------------------------------------------------------------------------
#definePRETTY_PRINT(This)printf("\n----------------\n%s\n------------------\n",(This))
//----------------------------------------------------------------------------------------------
#endif



Person.c

//----------------------------------------------------------------------------------------------
#include <string.h>
#include "CString.h"
#include "Person.h"
//----------------------------------------------------------------------------------------------
void SetNames(CString NewFamilyName,CString NewGivenName,
PersonType *APerson) {

//--- Comment
   strcpy(APerson->FamilyName,NewFamilyName);
   strcpy(APerson->GivenName,NewGivenName);
}
```

```
//-------------------------------------------------------------------------------
void GetFullName(PersonType APerson,CString FullName) {

    strcpy(FullName,APerson.GivenName);
    strcat(FullName," ");
    strcat(FullName,APerson.FamilyName);
}
//-------------------------------------------------------------------------------
Modular.c

//-------------------------------------------------------------------------------
#include <stdio.h>
#include <stdlib.h>
#include "CString.h"
#include "Person.h"
#include "PrettyPrint.h"
//-------------------------------------------------------------------------------
int main(void){
    PersonType MyPerson;
    CStringInputFamilyName;
    CString InputGivenName;
    CString FullName;
    CStringOutputLine;
    printf("Please enter the given name and family name : ");
    scanf(" %s %s",InputGivenName,InputFamilyName);
    SetNames(InputFamilyName,InputGivenName,&MyPerson);
    GetFullName(MyPerson,FullName);
    sprintf(OutputLine,"The full name is %s",FullName);
    PRETTY_PRINT(OutputLine);

    return(EXIT_SUCCESS);
}
//-------------------------------------------------------------------------------
```

Viva questions:

1. What is the use of make commands?
2. Explain about functions of make command.
3. What is compiler?
4. Why make file used?
5. Write the syntax to run the mak file.

Result:

Thus the program use **make** command executed and answer id verified.

Ex No:3 CONTROL SYSTEMS COMMAND TO CLONE, COMMIT, PUSH, FETCH, PULL, CHECKOUT, RESET,AND DELETE

Aim:

Use version control systems command to clone, commit, push, fetch, pull, checkout, reset, and delete repositories.

Description:

Git is a *version control system (software)* and **GitHub** is a *source code hosting service*. Git is a versioncontrolsystemfortrackingchangesincomputerfilesandcoordinatingworkonthosefiles among multiple people.

Lets Now, Understand the working of **Git**
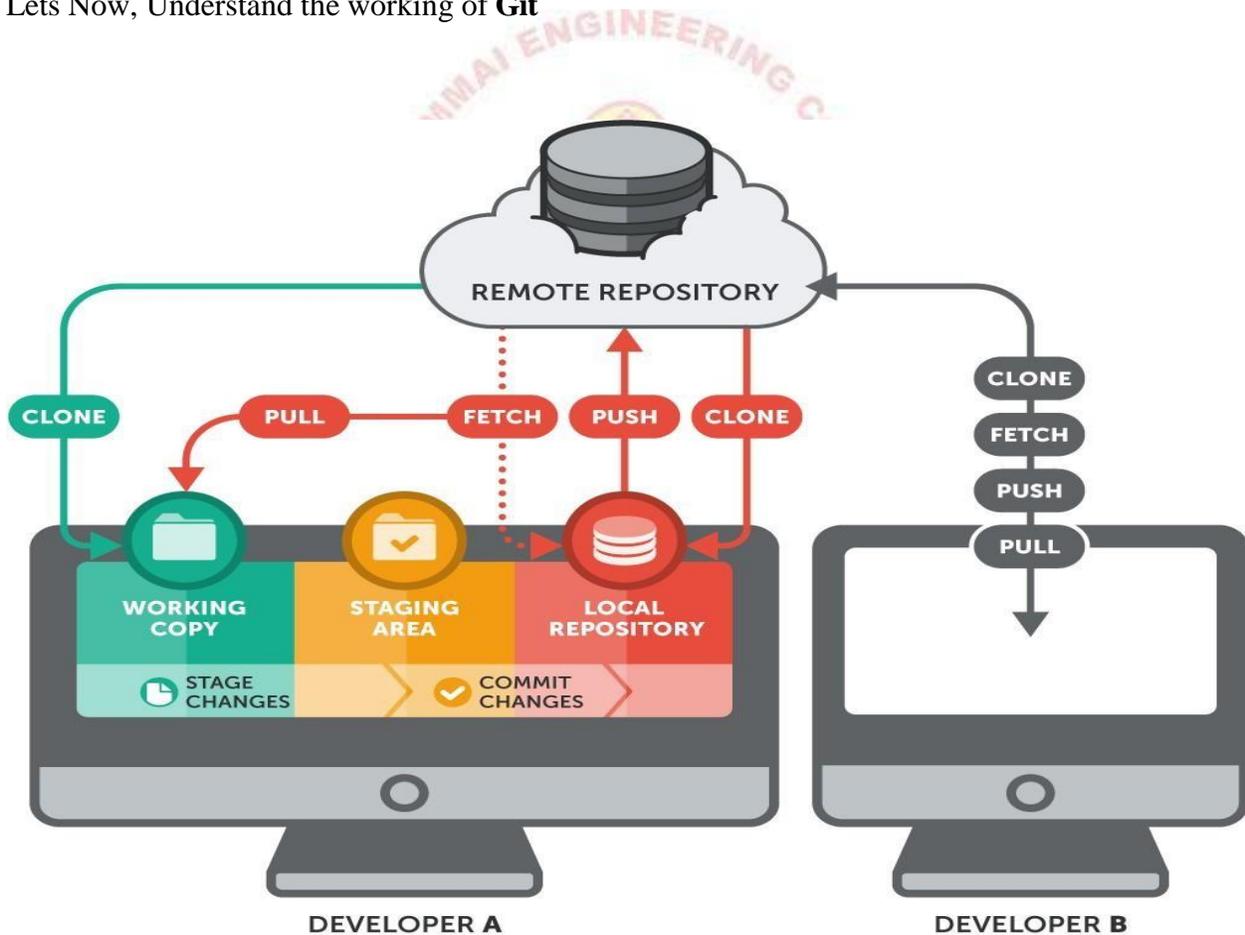
A file in git goes through the following stages:
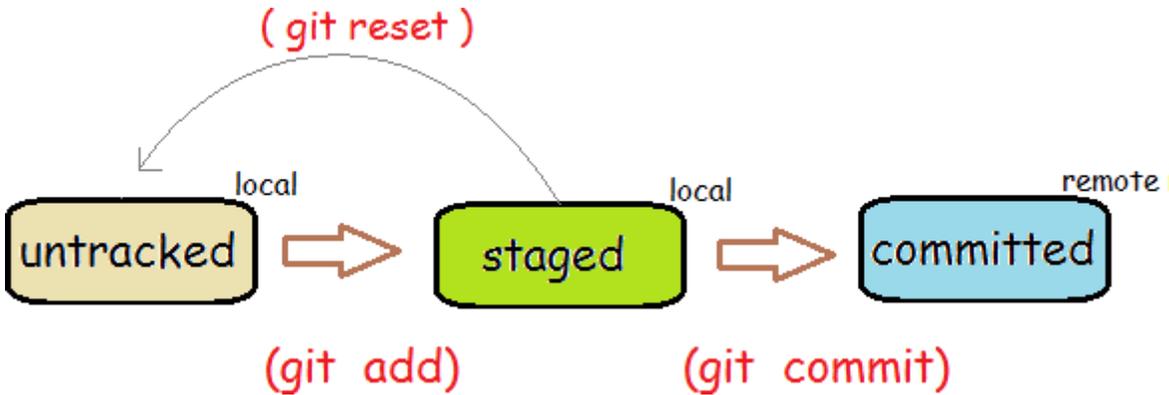
Procedure

Setting Up Git

You need to setup Git on your local machine, as follows:

1. Download &Install:

    o   For Windows and Mac, download the installer from http://git-scm.com/downloads and run the downloaded installer.
    o   For Ubuntu, issue command "sudo apt-get install git".
    For Windows, use the "Git Bash" command shell bundled with Git Installer to issue commands. For Mac/Ubuntu, use the "Terminal".

2. Customize   Git:
    Issue "git config" command (for Windows, run "Git Bash" from the Git installed directory. For Ubuntu/Mac, launch a "Terminal"):

3. // Set up your username and email (to be used in labeling your commits)

4. $ git config --global user.name"your-name"
    $ **git config --global user.email "*your-email@youremail.com*"**
    The settings are kept in "<GIT_HOME>/etc/gitconfig" (of the GIT installed directory) and "<USER_HOME>/.gitconfig"   (of   the   user's   home   directory.
    You can issue "git config --list" to list the settings:
    $ **git config --list**
    user.email=xxxxxx@xxxxxx.com
    user.name=xxxxxx

Git Commands

Git provides a set of simple, distinct, standalone commands developed according to the "Unix toolkit" philosophy - build small, interoperable tools.

To issue a command, start a "Terminal" (for Ubuntu/Mac) or "Git Bash" (for Windows):

$ **git<command><arguments>**

The commonly-used commands are:

1. **init**, **clone**, **config**: for starting a Git-managed project.

2. **add**, **mv**, **rm**: for staging filechanges.
3. **commit**, **rebase**, **reset,tag**:
4. **status,log,diff,grep,show**: show status
5. **checkout,branch,merge,push,fetch,pull**

Help and Manual

The best way to get help these days is certainly *googling*.

To get help on Git commands:

$ **git help** *<command>*
// or
$ **git***<command>* **--help**

The GIT manual is bundled with the software (under the "doc" directory), and also available online @ http://git-scm.com/docs.

Initialize a new Git Repo (git init)
To manage a project under Git, run "git init" at the project *root* directory (i.e., "hello-git") (via "Git Bash" for Windows, or "Terminal" for Ubuntu/Mac):

// Change directory to the project directory
$ **cd /***path-to***/hello-git**

// Initialize Git repo for this project
$ **git init**
Initialized empty Git repository in /*path-to*/hello-git/.git/

$ **ls -al**
drwxr-xr-x   1xxxxx   xxxxx   4096 Sep 14 14:58 .git
-rw-r--r--   1xxxxx   xxxxx   426 Sep 14 14:40 Hello.class
-rw-r--r--   1xxxxx   xxxxx   142 Sep 14 14:32 Hello.java
-rw-r--r--   1xxxxx   xxxxx   66 Sep 14 14:33 README.md

A hidden sub-directory called ".git" will be created under your project *root* directory (as shown in the above "ls -a" listing), which contains ALL Git related data.

Git Storage Model

The local repo after "git init" is empty. You need to explicitly deposit files into the repo. Before we proceed, it is important to stress that Git manages changes to files between so-called commits. In other words, it is a version control system that allows you to keep track of the file changes at the commits.

Staging File Changes for Tracking (git add <file>...)
Issue a "git status" command to show the status of the files:

$ **git status**
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)
     Hello.class
     Hello.java
     README.md
nothing added to commit but untracked files present (use "git add" to track)

By default, we start on a *branch* called "master". We will discuss "branch"later.
In Git, the files in the working tree are either *untracked* or *tracked*. Currently, all 3 files
are *untracked*. To stage a new file for tracking, use "git add <file>..."command.

```
// Add README.mdfile
$ git addREADME.md

$ git status
On branch master
Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
     new file:README.md

Untrackedfiles:
  (use "git add <file>..." to include in what will be committed)
     Hello.class
     Hello.java

// You can use wildcard * in the filename
// Add all Java source files into Git repo
$ git add *.java

// You can also include multiple files in the "git add"
// E.g.,
// git add Hello.java README.md

$ git status
On branch master
Initial commit

Changes to becommitted:
  (use "git rm --cached <file>..." tounstage)
     new file:Hello.java
     new file:README.md

Untracked files:
```

   (use "git add <file>..." to include in what will be committed)
     Hello.class

The command "git add <file>..." takes one or more filenames or pathnames with possibly wildcards pattern. You can also use "git add ." to add all the files in the current directory (and all sub-directories). But this will include "Hello.class", which we do not wish to be tracked.

When a new file is added, it is *staged* (or *index*ed, or *cached*) in the *staging area* (as shown in the GIT storage model), but NOT yet *committed*.

Git uses two stages to commit file changes:

1. "git add <file>" to stage file changes into the *staging area*, and
2. "git commit" to commit ALL the file changes in the *staging area* to the *local repo*.

The staging area allows you to group related file changes and commit them together.

Committing File Changes (git commit)

The "git commit" command commits ALL the file changes in the *staging area*. Use a -m option to provide a *message* for the commit.

$ **git commit -m "First commit"** // -m to specify the commit message
[master (root-commit) 858f3e7] first commit
 2 files changed, 8 insertions(+)
 create mode 100644 Hello.java
 create mode 100644 README.md

// Check the status
$ **git status**
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Hello.class
nothing added to commit but untracked files present (use "git add" to track)

Commit ALL staged file changes via "git commit":

$ **git commit -m "Second commit"**
[master 96efc96] Second commit
 1 file changed, 1 insertion(+)

$ **git status**
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Hello.class
nothing added to commit but untracked files present (use "git add" to track)

Once the file changes are committed, it is marked as *unmodified* in the staging area (not shown in "Changes to be committed").

Both "git diff" and "git diff --staged" return empty output, signaling there is no "upstaged" and "staged" changes.

More on "git checkout" and Detached HEAD

"git checkout" can be used to checkout a branch, a commit, or files. The syntaxes are:

$ **git checkout <branch-name>**
$ **git checkout <commit-name>**
$ **git checkout <commit-name><filename>**

When you checkout a commit, Git switches into so-called "Detached HEAD" state, i.e., the HEAD detached from the tip of a branch. Suppose that you continue to work on the detached HEAD on commit-5, and wish to merge the commit-5 back to master. You checkout the master branch, but there is no branch name for your to reference the commit-5!!!

In Summary, you can use "git checkout <commit-name>" to inspect a commit. BUT you should always work on a branch, NOT on a detached HEAD.

More on "git reset" and "git reset --hard"

$ **git reset <file>**
  // Unstage the changes of <file> from staging area,
  // not affecting the working tree.

$ **git reset**
  // Reset the staging area
  // Remove all changes (of all files) from staging area,
  // not affecting the working tree.

$ **git reset --hard**
  // Reset the staging area and working tree to match the
  // recent commit (i.e., discard all changes since the
  // last commit).

$ **git reset <commit-name>**
  // Move the HEAD of current branch to the given commit,
  // not affecting the working tree.

$ **git reset --hard <commit-name>**
  // Reset both staging area and working tree to the given
  // commit, i.e., discard all changes after that commit.

[TODO] Diagram

[TODO] --soft option

Summary of Work Flows

Setting up GIT and "Edit/Stage/Commit/Push" Cycle
Step 1: Install GIT.
- For Windows and Mac, download the installer from http://git-scm.com/downloads and run the downloaded installer.

- For Ubuntu, issue command "sudo apt-get install git".

For Windows, use "git-bash" command shell provided by Windows installer to issue command.
For Mac/Ubuntu, use "Terminal".

Step 2: Configuring GIT:

```
// Setup your username and email to be used in labeling commits
$ git config --global user.email "your-email@yourmail.com"
$ git config --global user.name "your-name"
```

Step 3: Set up GIT repo for a project. For example, we have a project called "olas1.1" located at "/usr/local/olas/olas1.1".

```
$ cd /usr/local/olas/olas1.1

// Initialize the GIT repo
$ git init

$ ls -al
   // Check for ".git" directory
```

Create a "README.md" (or "README.textile" if you are using Eclipse's WikiText in "textile" markup) under your project directory to describe the project.

Step 4: Start "Edit/Stage/Commit/Push" cycles.
Create/Modify files. Stage files into the staging area via "git add <file>".

```
// Check the status
$ git status
......

// Add files into repo
$ git add README.md
$ git add www
......

// Check the status
$ git status
......
```

Step 5: Create a ".gitignore" (in the project base directory) to exclude folders/files from being tracked by GIT. Check your "git status" output to decide which folders/files to be ignored.

For example,

```
# ignore files and directories beginning with dot
.*

# ignore directories beginning with dot (a directory ends with a slash)
.*/
```

```
# ignore these files and directories
www/test/
```

```
www/.*
www/.*/
```

The trailing slash indicate directory (and its sub-directories and files).

If you want the ".gitignore" to be tracked (which is in the ignore list):

```
$ git add -f .gitignore
    // -f to override the .gitignore
```

Step 6: Commit.

```
$ git status
......

// Commit with a message
$ git commit -m "Initial Commit"
......

$ git status
......
```

Step 7: Push to the Remote Repo (for backup, version control, and collaboration).
You need to first create a repo (says olas) in a remote GIT host, such as GitHub or BitBucket.
Take note of the remote repo URL, e.g., https://*username*@*hostname*.org/*username*/olas.git.

```
$ cd /path-to/local-repo

// Add a remote repo name called "origin" mapped to the remote URL
$ git remote add origin https://hostname/username/olas.git

// Push the "master" branch to the remote "origin"
// "master" is the default branch name of your local repo after init.
$ git push origin master
```

Check the remote repo for the files committed.

Step 8: Work on the source files, make changes, commit and push to remote repo.

```
// Check the files modified
$ git status
......

// Stage for commit the modified files
$ git add ....
......

// Commit (with a message)
$ git commit -m "commit-message"

// Push to remote repo
```

```
$ git push origin master
```

Step 9: Create a "tag" (for version number).

```
// Tag a version number to the current commit
$ git tag -a v1.1 -m "Version 1.1"
    // -a to create an annotated tag, -m to provide a message

// Display all tags
$ git tag
......

// Push the tags to remote repo
// ("git push -u origin master" does not push the tags)
$ git push origin --tags
```

.
Fetch/Merge Changes from remote (gitfetch/merge)

The "git fetch" command imports commits from a remote repo to your local repo, without updating your local working tree. This gives you a chance to review changes before updating (merging into) your working tree. The fetched objects are stored in remote branches, that are differentiated from the local branches.

```
$ cd /path-to/working-directory

$ git fetch <remote-name>
  // Fetch ALL branches from the remote repo to your local repo

$ git fetch <remote-name><branch-name>
  // Fetch the specific branch from the remote repo to your local repo

// List the local branches
$ git branch
* master
  devel
  // * indicates current branch

// List the remote branches
$ git branch -r
  origin/master
  origin/devel

// You can checkout a remote branch to inspect the files/commits.
// But this put you into "Detached HEAD" state, which prevent you
// from updating the remote branch.

// You can merge the fetched changes into local repo
$ git checkout master
```

// Switch to "master" branch of local repo
$ **git merge origin/master**
// Merge the fetched changes from stored remote branch to local

git pull

As a short hand, "git pull" combines "git fetch" and "git merge" into one command, for convenience.

$ **git pull <remote-name>**
// Fetch the remote's copy of the current branch and merge it
// into the local repo immediately, i.e., update the working tree

// Same as
$ **git fetch <remote-name><current-branch-name>**
$ **git merge <remote-name><current-branch-name>**

$ **git pull --rebase <remote-name>**
// linearize local changes after the remote branch.

The "git pull" is an easy way to *synchronize* your local repo with origin's (or upstream) changes (for a specific branch).

Pushing to Remote Repo(revision)

The "git push <remote-name><branch-name>" is the counterpart of "git fetch", which exports commits from local repo to remote repo.

$ **git push <remote-name><branch-name>**
// Push the specific branch of the local repo

$ **git push <remote-name> --all**
// Push all branches of the local repo

$ **git push <remote-name> --tag**
// Push all tags
// "git push" does not push tags

$ **git push -u <remote-name><branch-name>**
// Save the remote-name and branch-name as the
// reference (or current) remote-name and branch-name.
// Subsequent "git push" without argument will use these references.

REFERENCES & RESOURCES
1. GIT mother site @ http://git-scm.com and GIT Documentation @http://git-scm.com/doc.

Viva questions:

1. What is version control commands?
2. What is GIT?
3. Where we use the commit?
4. Why should we use clone?
5. What is push and pull commands?

Result:

Thus the control commands are executed in GIT.

Ex No : 4 INSTALL VIRTUALBOX/VMWARE WORKSTATION

Aim:

To Install Virtualbox/VMware Workstation with different flavours of linux or windows OS on top of windows7 or 8.

Procedure:

Before downloading and installing VMware Workstation:

Ensure that you are using a supported guest operating system

Downloading VMware Workstation

To download VMware Workstation:
 Navigate to the VMware Workstation Download Center.

1. Based on your requirements, click **Go to Downloads** for VMware Workstation for Windows or VMware Workstation for Linux.
2. Click **Download Now**.
3. If prompted, log in to your My VMware profile. If you do not have a profile, create one. For more information, see How to create a My VMware profile(2007005).
4. Ensure that your profile is complete and enter all mandatory fields. For more information, see How to update your My VMware profile(2086266).
5. Review the End User License Agreement and click **Yes**.
6. Click **Download Now**.

 If the installer fails to download during the download process:

- Delete the cache in your web browser. For more information, see:

  - Mozilla Firefox: How to clear the Firefox cache
  - Google Chrome: Delete your cache and other browser data
  - Microsoft Internet Explorer: How to delete the contents of the Temporary Internet Files folder
- Disable the pop-up blocker in your web browser. For more information, see:

  - Mozilla Firefox: How do I disable a Pop-up blocker?
  - Google Chrome: Manage pop-ups
  - Microsoft Internet Explorer: How to turn Internet Explorer Pop-up Blocker on or off on a Windows XP SP2-basedcomputer
- Download using a different web browser application.
- Disable any local firewall software.

- Restart the virtual machine.
- Download the installer from a different computer or network.

Installing VMware Workstation
Notes:

- You must have only one VMware Workstation installed at a time. You must uninstall previous version of VMware Workstation before installing a new version.
- If the installer reports an error when you run it, you must verify the download. For more information, see Verifying the integrity of downloaded installer files(1537).

To install VMware Workstation on a Windows host:

1. Log in to the Windows host system as the Administrator user or as a user who is a member of the local Administrators group.
2. Open the folder where the VMware Workstation installer was downloaded. The default location is the **Downloads** folder for the user account on the Windows host.

   Note: The installer file name is similar to VMware-workstation-full-*xxxx-xxxx*.exe, where *xxxx-xxxx* is the version and build numbers.

3. Right-click the installer and click **Run asAdministrator**.
4. Select a setupoption:

   - **Typical**: Installs typical Workstation features. If the Integrated Virtual Debugger for Visual Studio or Eclipse is present on the host system, the associated Workstation plug-ins are installed.
   - **Custom**: Lets you select which Workstation features to install and specify where to install them. Select this option if you need to change the shared virtual machines directory, modify the VMware Workstation Server port, or install the enhanced virtual keyboard driver. The enhanced virtual keyboard driver provides better handling of international keyboards and keyboards that have extra keys.

   5. Follow the on-screen instructions to finish the installation.
   6. Restart the host machine.

To install VMware Workstation on a Linux host:

Note: VMware Workstation for Linux is available as a .bundle download in the VMware Download Center. The Linux bundle installer starts a GUI wizard on most Linux distributions. In some Linux distributions, the bundle installer starts a command-line wizard instead of a GUI wizard.

1. Log in to the Linux host with the user account that you plan to use with VMware Workstation.

2. Open a terminal interface. For more information, see Opening a command or shell prompt (1003892).
3. Change to root. For example:

su root

Note: The command that you use depends on your Linux distribution and configuration.
4. Change directories to the directory that contains the VMware Workstation bundle installer file. The default location is the **Download** directory.
5. Run the appropriate Workstation installer file for the host system.

For example:

sh VMware-workstation-Full-*xxxx-xxxx.architecture*.bundle [--

*option]*Where:

- *xxxx-xxxx* is the version and build numbers
- architecture is i386 or x86_64
- *option* is a command line option.
- This table describes the command line options:

Option	Description
--gtk	**Opens the GUI-based VMware installer, which is the default option.**
--console	**Use the terminal for installation.**
--custom	**Use this option to customize the locations of the installation directories and set the hard limit for the number of open file descriptors.**
--regular	**Shows installation questions that have not answered before or are required. This is the default option.**
-- ignore- errors **or** **-I**	**Allows the installation to continue even if there is an error in one of the installer scripts. Because the section that has an error does not complete, the component might not be properly configured.**
--required	**Shows the license agreement only and then proceeds to install Workstation**

6. Accept the license agreement.

> **Note**: If you are using the --console option or installing VMware Workstation on a Linux host that does not support the GUI wizard, press Enter to scroll through and read the license agreement or type q to skip to the yes/no prompt.

7. Follow the on-screen instructions or prompts to finish the installation.
8. Restart the Linux host.

After installation

On Windows host systems:

- The installer creates a desktop shortcut, a quick launch shortcut, or a combination of these options in addition to a Start Menu item.
- To start VMware Workstation on a Windows host system, select **Start** >**Programs** >**VMwareWorkstation**.

On Linux host systems:

- VMware Workstation can be started from the command line on all Linux distributions.
- On some Linux distributions, VMware Workstation can be started in the GUI from the **System Tools** menu under **Applications**.
- To start VMware Workstation on a Linux host system from the command line, run the vmware & command in a terminal window. For more information, see Opening a command or shell prompt(1003892).

For example:

/usr/bin/vmware &

When you start the Workstation for the first time, Workstation prompts you to accept the **End User License Agreement**. After you start Workstation, the Workstation window opens.

To install VM on windows go to web site

https://www.iitk.ac.in/nt/faq/vbox.htm

Simple Steps to a New Virtual Machine

By default, the new virtual machine uses an IDE disk for Windows 95, Windows 98, Windows Me, Windows XP, Windows Server 2003, NetWare and FreeBSD guests. The default for other guest operating systems is a SCSI disk.

Follow these steps to create a virtual machine using a virtual disk.

1. Start VMware Workstation.

Windows hosts: Double-click the VMware Workstation icon on your desktop or use the **Start** menu (**Start** >**Programs** >**VMware** >**VMware Workstation**).
Linux hosts: In a terminal window, enter the command
vmware &

2. If this is the first time you have launched VMware Workstation and you did not enter the serial number when you installed the product (an option available on a Windows host), you are prompted to enter it. The serial number is on the registration card in your package. Enter your serial number and click **OK**.

The serial number you enter is saved and VMware Workstation does not ask you for it again. For your convenience, VMware Workstation automatically sends the serial number to the VMware Web site when you use certain Web links built into the product (for example, **Help** >**VMware software on the Web** >**Register Now!** and **Help** >**VMware on the Web** >**Request Support**). This allows us to direct you to the correct Web page to register and get support for your product.

3. **Linux hosts:** If this is the first time you have launched VMware Workstation, a dialog box asks if you want to rename existing virtual disks using the new .vmdk extension. Click **OK** to search all local drives on the host computer and make this change. (On Windows hosts, you have a chance to rename virtual disk files when you are installing VMware Workstation.)

The converter also renames the files that store the state of a suspended virtual machine, if it finds them. It changes the old .std file extension to .vmss. However, it is best to resume and shut down all suspended virtual machines before you upgrade to Workstation 4.

Besides renaming files, the converter updates the corresponding virtual machine configuration files so they identify the virtual disks using the new filenames.

If you store your virtual disk files or suspended state files on a Windows XP or Windows Server 2003 host - or if you may do so in the future - it is important to convert the filenames to avoid conflicts with the System Restore feature of Windows XP and Windows Server 2003.

Linux Hosts: One Chance to Rename Disk Files
The Rename Virtual Disks dialog box appears only once. If you click **Cancel**, you will not have another opportunity to update the filenames and configuration files automatically.

4. Start the New Virtual Machine Wizard.
When you start VMware Workstation, you can open an existing virtual machine or create a new one. Choose **File** >**New** >**New Virtual Machine** to begin creating your virtual machine.

5. The New Virtual Machine Wizard presents you with a series of screens that you navigate using the Next and Prev buttons at the bottom of each screen. At each screen, follow the instructions, then click **Next** to proceed to the next screen.

6. Select the method you want to use for configuring your virtual machine.
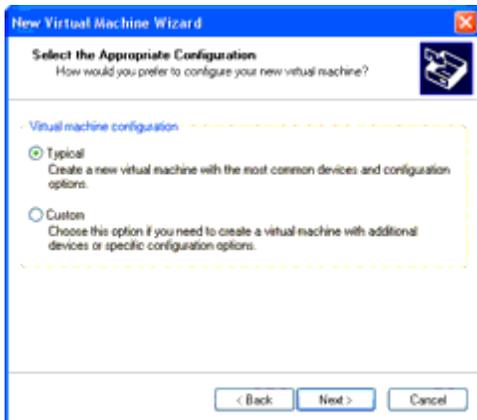
If you select **Typical**, the wizard prompts you to specify or accept defaults for

- o The guest operating system
- o The virtual machine name and the location of the virtual machine's files
- o The network connection type
  If you select **Custom**, you also can specify how to set up your disk - create a new virtual disk, use an existing virtual disk or use a physical disk - and specify the settings needed for the type of disk you select.
  Select **Custom** if you want to
- o Make a virtual disk larger or smaller than 4GB
- o Store your virtual disk's files in a particular location
- o Use an IDE virtual disk for a guest operating system that would otherwise have a SCSI virtual disk created by default
- o Allocate all the space for a virtual disk at the time you create it
- o Choose whether to split a virtual disk into 2GBfiles
- o Use a physical disk rather than a virtual disk (for expert users)
- o Set memory options that are different from the defaults.

7. Select a guest operating system.
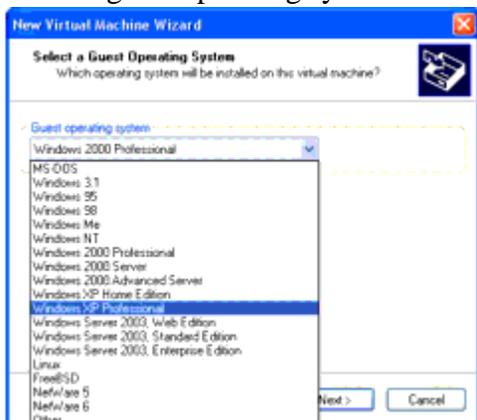
This screen asks which operating system you plan to install in the virtual machine. The New Virtual Machine Wizard uses this information to select appropriate default values, such as the amount of memory needed. The wizard also uses this information when naming associated virtual machine files.

If the operating system you are using is not listed, select **Other**.

The remaining steps assume you plan to install a Windows XP Professional guest operating system. You can find detailed installation notes for this and other guest operating systems in the *VMware Guest Operating System Installation Guide*, available from the VMware Web site or from the Help menu.

8. Select a name and folder for the virtual machine.
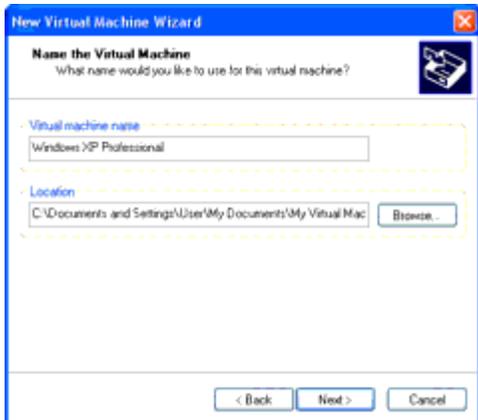
The name specified here is used if you add this virtual machine to the VMware Workstation Favorites list. This name is also used as the name of the folder where the files associated with this virtual machine are stored.

Each virtual machine should have its own folder. All associated files, such as the configuration file and the disk file, are placed in this folder.

Windows hosts: On Windows 2000, Windows XP and Windows Server 2003, the default folder for this Windows XP Professional virtual machine is C:\Documents and Settings\<username>\My Documents\My Virtual Machines\Windows XP Professional. On Windows NT, the default folder is C:\WINNT\Profiles\<username>\Personal\My Virtual Machines\Windows XP Professional.
Linux hosts: The default location for this Windows XP Professional virtual machine is <homedir>/vmware/winXPPro, where <homedir> is the home directory of the user who is currently logged on.
Virtual machine performance may be slower if your virtual hard disk is on a network drive. For best performance, be sure the virtual machine's folder is on a local drive. However, if other users need to access this virtual machine, you should consider placing the virtual machine files in a location that is accessible to them. For more information, see Sharing Virtual Machines with OtherUsers.

9. Configure the networking capabilities of the virtual machine.
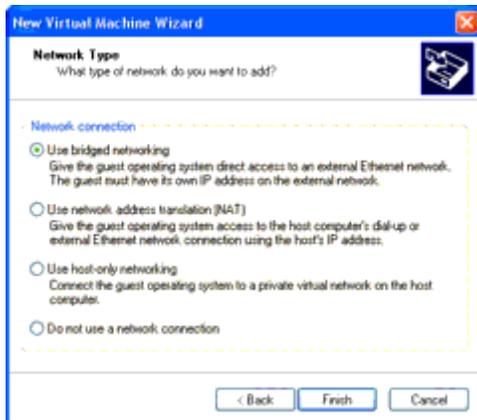
If your host computer is on a network and you have a separate IP address for your virtual machine (or can get one automatically from a DHCP server), select **Use bridged networking**.

If you do not have a separate IP address for your virtual machine but you want to be able to connect to the Internet, select **Use network address translation (NAT)**. NAT is useful if you have a wireless network adapter on a Linux host (as bridged networking on wireless network adapters is supported only on Windows hosts). It also allows for the sharing of files between the virtual machine and the host operating system.

For more details about VMware Workstation networking options, see Networking.

10. If you selected **Typical** as your configuration path, click **Finish** and the wizard sets up the files needed for your virtual machine.

    If you selected **Custom** as your configuration path, continue with the steps for configuring a disk for your virtual machine.

11. Select the disk you want to use with the virtual machine.
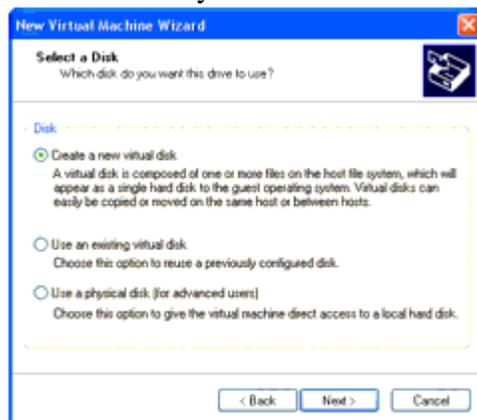
Select **Create a new virtual disk**.

Virtual disks are the best choice for most virtual machines. They are quick and easy to set up and can be moved to new locations on the same host computer or to different host computers. By default, virtual disks start as small files on the host computer's hard drive, then expand as needed - up to the size you specify in the next step. The next step also allows you to allocate all the disk space when the virtual disk is created, if you wish.

To use an existing operating system on a physical hard disk (a "raw" disk), readConfiguringaDual-BootComputerforUsewithaVirtualMachine.Toinstallyour

guest operating system directly on an existing IDE disk partition, read the reference note Installing an Operating System onto a Raw Partition from a Virtual Machine.
Caution: Raw disk configurations are recommended only for expert users.
Caution: If you are using a Windows Server 2003, Windows XP or Windows 2000 host, see Do Not Use Windows 2000, Windows XP and Windows Server 2003 Dynamic Disks as Raw Disks.
To install the guest operating system on a raw IDE disk, select **Existing IDE Disk Partition**. To use a raw SCSI disk, add it to the virtual machine later with the Virtual Machine Control Panel. Booting from a raw SCSI disk is not supported. For a discussion of some of the issues involved in using a raw SCSI disk, see Configuring Dual-orMultiple-Boot SCSI Systems to Run with VMware Workstation on a Linux Host.

12. Specify the capacity of the virtualdisk.
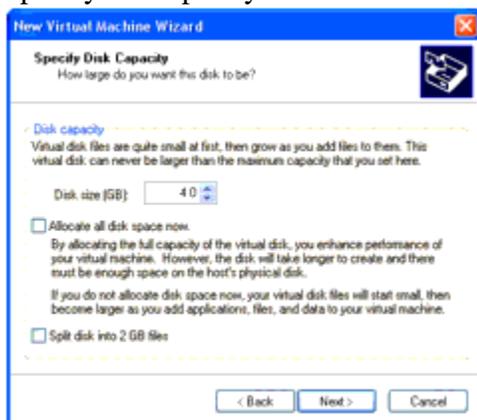
Enter the size of the virtual disk that you wish to create.

If you wish, select **Allocate all disk space now**.
Allocating all the space at the time you create the virtual disk gives somewhat better performance, but it requires as much disk space as the size you specify for the virtual disk.

If you do not select this option, the virtual disk's files start small and grow as needed, but they can never grow larger than the size you set here.

You can set a size between 2GB and 256GB for a SCSI virtual disk or 128GB for an IDE virtual disk. The default is 4GB.

You may also specify whether you want the virtual disk created as one large file or split into a set of 2GB files.

Make the Virtual Disk Big Enough
The virtual disk should be large enough to hold the guest operating system and all of the software that you intend to install, with room for data and growth.

You cannot change the virtual disk's maximum capacity later.

You can install additional virtual disks using the Virtual Machine Control Panel

For example, you need about 500MB of actual free space on the file system containing the virtual disk to install Windows Me and popular applications such as Microsoft Office inside the virtual machine. You can set up a single virtual disk to hold these files. Or you can split them up - installing the operating system on the first virtual disk and using a second virtual disk for applications or data files.

13. Specify the location of the virtual disk's files.

If a SCSI virtual disk is created by default and you want to use a virtual IDE disk instead, or if you want to specify which device node should be used by your SCSI or IDE virtual disk, click **Advanced**.
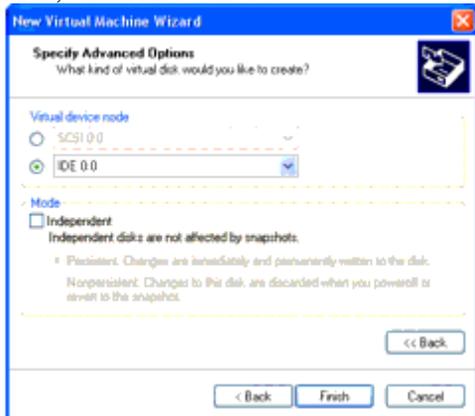
On the advanced settings screen, you can also specify a disk mode. This is useful in certain special-purpose configurations in which you want to exclude disks from the snapshot. For more information on the snapshot feature, see Using the Snapshot. Normal disks are included in the snapshot. In most cases, this is the setting you want.

Independent disks are not included in the snapshot.

Caution: The independent disk option should be used only by advanced users who need it for special-purpose configurations.
You have the following options for an independent disk:

   o **Persistent** - changes are immediately and permanently written to thedisk.

- o **Nonpersistent** - changes to the disk are discarded when you power off or revert to the snapshot.

    When you have set the filename and location you want to use and have made any selections you want to make on the advanced settings screen, click **Finish**.

14. Click **Finish**. The wizard sets up the files needed for your virtual machine.

Viva Questions:

1. What is VMWARE?
2. What is the use of virtual machine?
3. How can we create the VM?
4. How will you allocate the memory to VM?
5. How will you make the virtual block?

Result:

Thus the virtual machine is created.

Ex No 5: INSTALL A C COMPILER IN THE VIRTUAL MACHINE AND EXECUTE A SAMPLE PROGRAM.

Aim :

To install a C compiler in the virtual machine and execute a sample program.

Procedure:

step1:
Install the centos or ubuntu in the open nebula as per previous commands.

Step 2:
Login into the VM of installed OS.

Step 3:
If it is ubuntu then, for gcc installation
$ sudo add-apt-repository ppa:ubuntu-toolchain-r/test
$ sudo apt-get update
$ sudo apt-get install gcc-6 gcc-6-base

Step 4:
Write a sample program like
Welcome.cpp
```
#include<iostream.h>
using namespace std;
int main()
{
cout<<"Hello world";
return 0;
}
```

Step 5:
First we need to compile and link our program. Assuming the source code is saved in a file welcome.cpp, we can do that using GNU C++ compiler g++, for example
g++ -Wall -o welcome welcome.cpp

And output can be executed by ./welcome
https://www.iitk.ac.in/nt/faq/vbox.htm

Viva Questions:

1. What is virtual machine?
2. What is compiler?
3. How will you run the c compiler?

4. What is GCC compiler?
5. Explain the uses of VM.

Result:
    Thus the GCC compiler has been successfully installed and executed a sample program.

Ex No: 6 INSTALL GOOGLE APP ENGINE

Aim :

To Install Google App Engine. Create hello world app and other simple web applications using python/java.

Procedure:

Use **Eclipse** to create a **Google App Engine** (GAE) **Java** project (hello world example), run it locally, and deploy it to Google App Engine account.

Tools used:

1. JDK1.6
2. Eclipse 3.7 + Google Plugin for Eclipse
3. Google App Engine Java SDK 1.6.3.1

Note
GAE supports Java 1.5 and 1.6.
P.S Assume JDK1.6 and Eclipse 3.7 are installed.

1. Install Google Plugin for Eclipse

Read this guide – how to install Google Plugin for Eclipse. If you install the Google App Engine Java SDK together with "**Google Plugin for Eclipse**", then go to step 2, Otherwise, get The Google App Engine Java SDK and extract it.

2. Create New Web Application Project
In Eclipse toolbar, click on the Google icon, and select "**New Web Application Project…**"

Figure – New Web Application Project
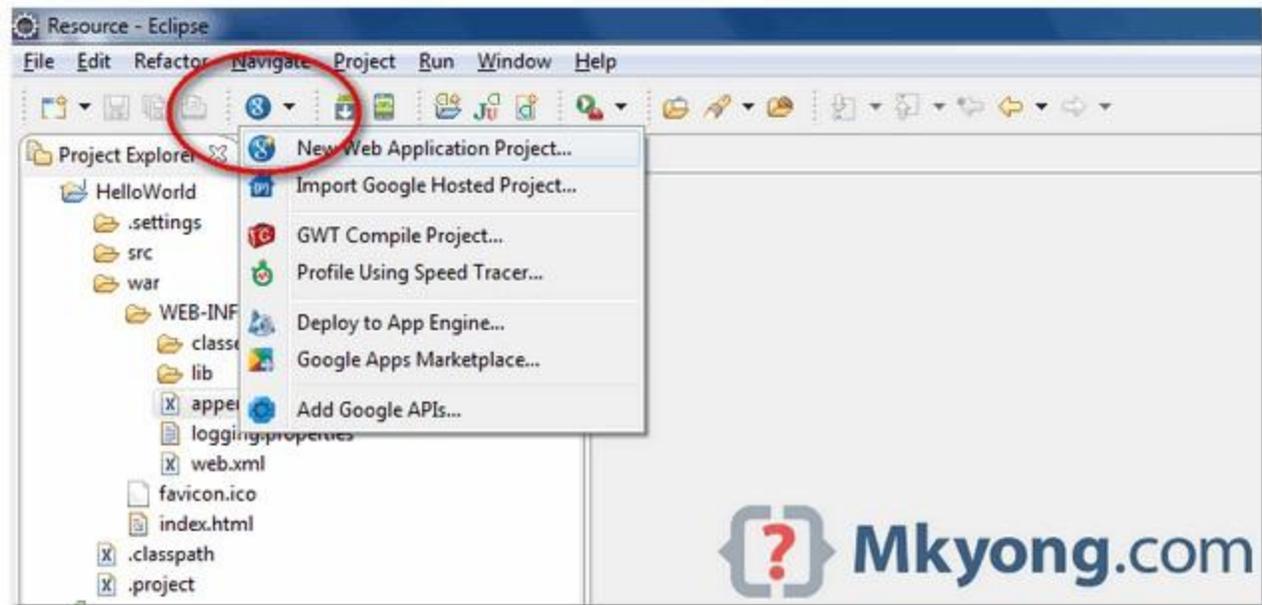
*Figure – Deselect the "**Google Web ToolKit**", and link your GAE Java SDK via the "**configure SDK**" link.*
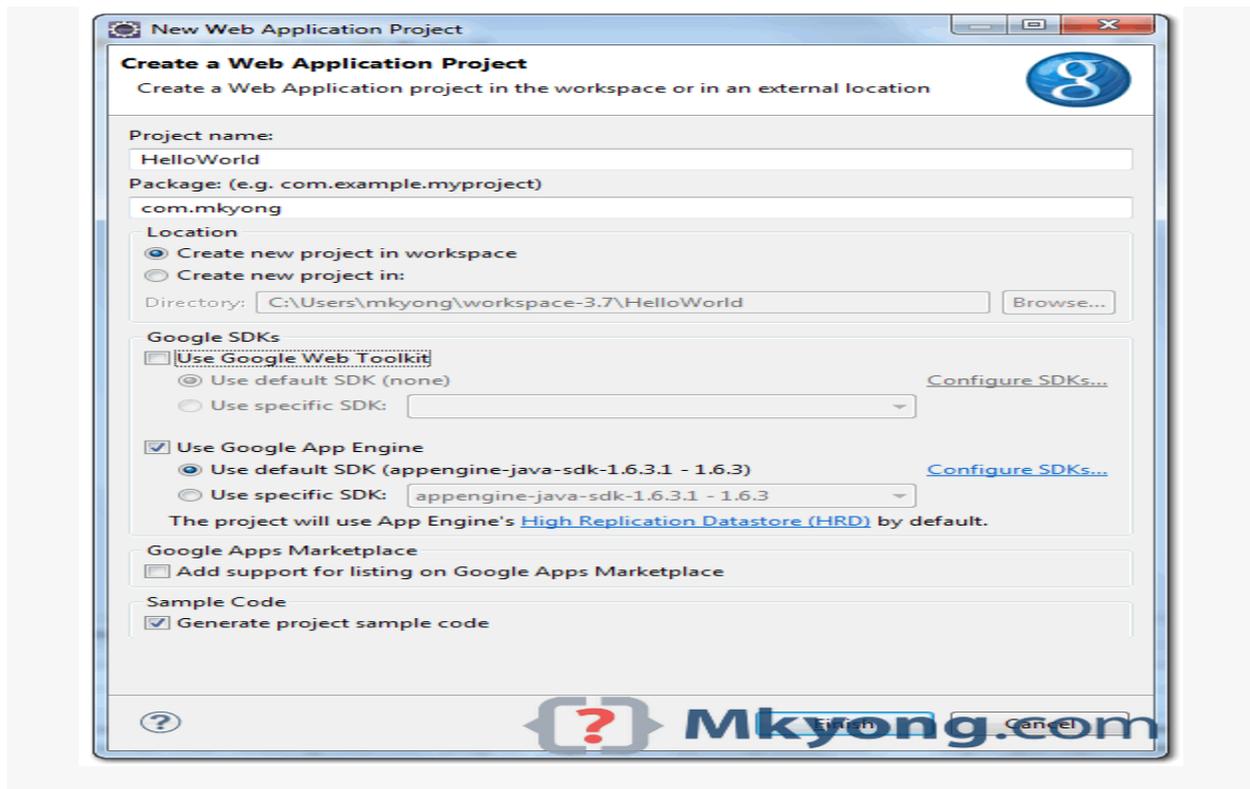
Click finished, Google Plugin for Eclipse will generate a sample project automatically.

3. HelloWorld
Review the generated project directory.

Nothing special, a standard Java web project structure.

```
HelloWorld/
  src/
    ...Java source code...
    META-INF/
      ...other configuration...
  war/
    ...JSPs, images, data files...
    WEB-INF/
      ...app configuration...
      lib/
        ...JARs for libraries...
      classes/
        ...compiled classes...
```
Copy

The extra is this file "appengine-web.xml", Google App Engine need this to run and deploy the application.

File : appengine-web.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
  <application></application>
  <version>1</version>

  <!-- Configure java.util.logging -->
  <system-properties>
    <property name="java.util.logging.config.file" value="WEB-INF/logging.properties"/>
  </system-properties>

</appengine-web-app>
```
Copy

4. Run itlocal
Right click on the project and run as "**Web Application**".

Eclipse console :

```
//...
INFO: The server is running at http://localhost:8888/
30 Mac 2012 11:13:01 PM com.google.appengine.tools.development.DevAppServerImpl start
INFO: The admin console is running at http://localhost:8888/_ah/admin
```
Copy

Access URL http://localhost:8888/, see output
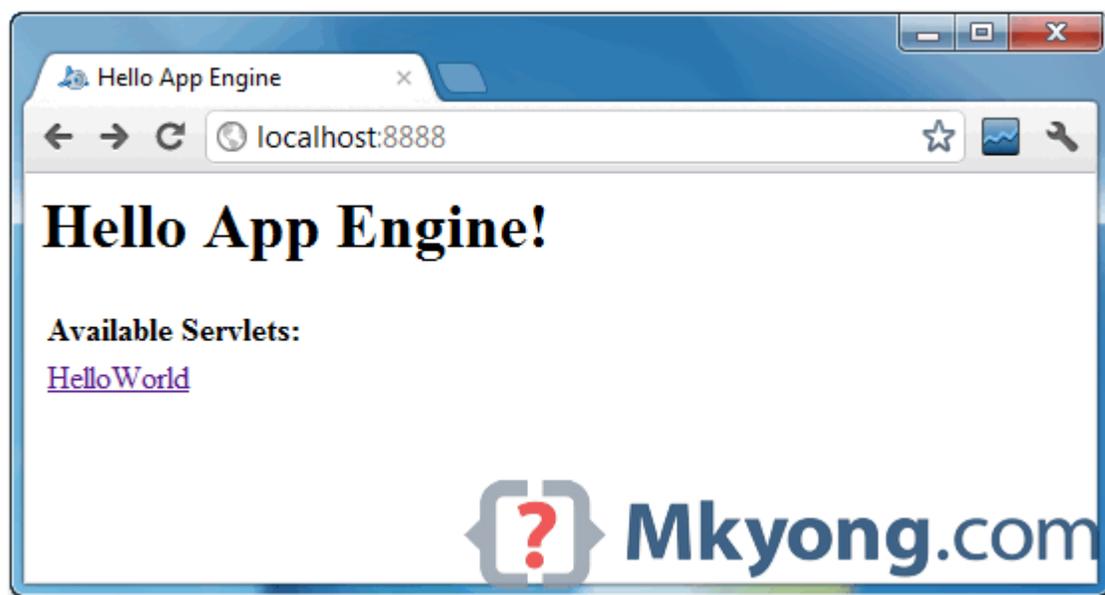
and also the hello world servlet – http://localhost:8888/helloworld
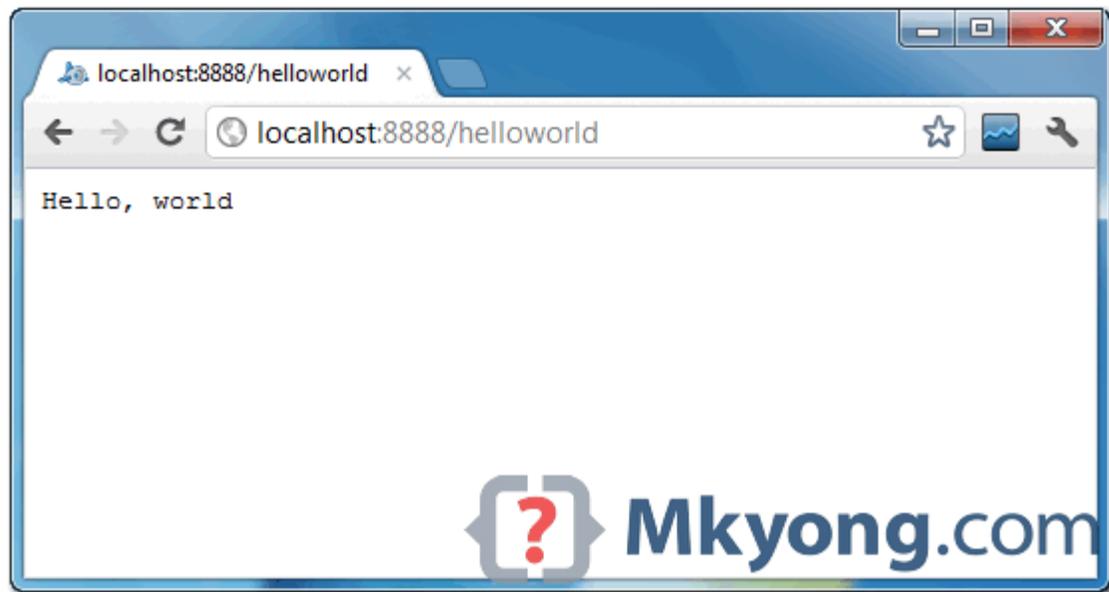
5. Deploy to Google App Engine

Register an account on https://appengine.google.com/, and create an application ID for your web application.

In this demonstration, I created an application ID, named "mkyong123", and put it in appengine-web.xml.

File : appengine-web.xml

```
<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
 <application>mkyong123</application>
 <version>1</version>

 <!-- Configure java.util.logging -->
 <system-properties>
   <property name="java.util.logging.config.file" value="WEB-INF/logging.properties"/>
 </system-properties>

</appengine-web-app>
```
Copy

To deploy, see following steps:

Figure 1.1 – Click on GAE deploy button on the toolbar.
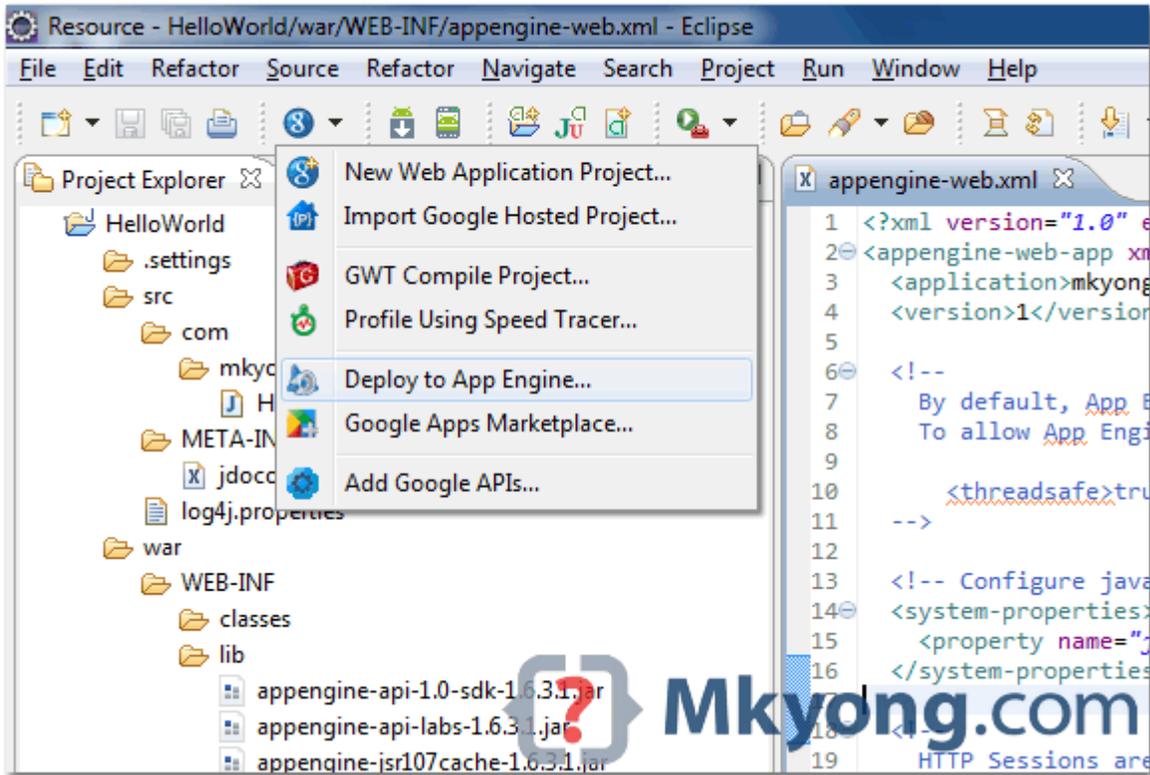
Figure 1.2 – Sign in with your Google account and click on the Deploy button.
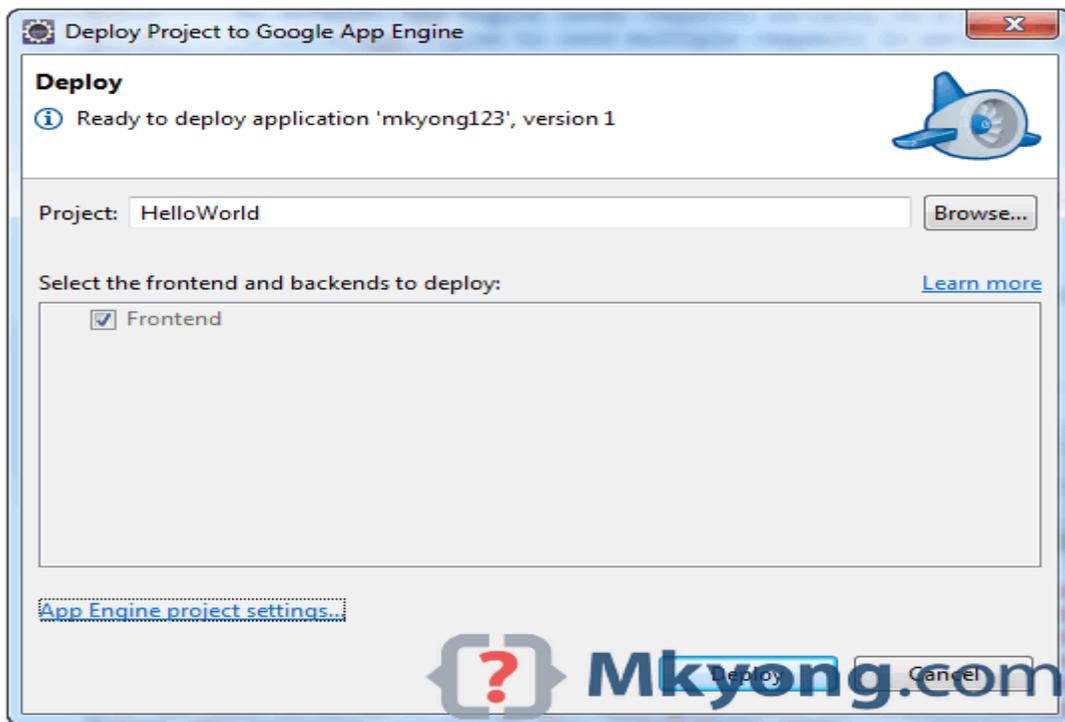
Figure 1.3 – If everything is fine, the hello world web application will be deployed to this URL –
http://mkyong123.appspot.com/

Done.

References

1. Google App Engine – Getting Started:Java
2. Google app engine Python hello world example using Eclipse

Viva Questions:

1. What is GAE?
2. What is ASP?
3. What is JSP?
4. Explain the procedure to create the GAE
5. What is SDK?

Result:

Thus the GAE is installed and executed the hello world application.

Ex No: 7 HOSTING A STATIC WEBSITE ON GOOGLE APP ENGINE

Aim:

To use GAE launcher to launch the web applications**.**

Procedure:

You can use Google App Engine to host a static website. Static web pages can contain client-side technologies such as HTML, CSS, and JavaScript. Hosting your static site on App Engine can cost less than using a traditional hosting provider, as App Engine provides a free tier.

Sites hosted on App Engine are hosted on the REGION_ID.r.appspot.com subdomain, such as [my-project-id].uc.r.appspot.com. After you deploy your site, you can map your own domain name to your App Engine-hosted website.

Before you begin

Before you can host your website on Google App Engine:

1. Create a new Cloud Console project or retrieve the project ID of an existing project to use: Go to the Project page

   Tip: You can retrieve a list of your existing project IDs with the gcloud command line tool.
2. Install and then initialize the Google Cloud SDK:

   Download the SDK

   Creating a website to host on Google App Engine

   Basic structure for the project

   This guide uses the following structure for the project:

- app.yaml: Configure the settings of your App Engine application.

- www/: Directory to store all of your static files, such as HTML, CSS, images, and JavaScript.

- css/: Directory to store style sheets.

- style.css: Basic style sheet that formats the look and feel of your site.

- images/: Optional directory to store images.

- index.html: An HTML file that displays content for your website.

- js/: Optional directory to store JavaScript files.

- Other asset directories.

Creating the app.yaml file

The app.yaml file is a configuration file that tells App Engine how to map URLs to your static files. In the following steps, you will add handlers that will load www/index.html when someone visits your website, and all static files will be stored in and called from the www directory.

Create the app.yaml file in your application's root directory:

1. Create a directory that has the same name as your project ID. You can find your project ID in the Console.

2. In directory that you just created, create a file namedapp.yaml.

3. Edit the app.yaml file and add the following code to the file:

```
runtime: python27
api_version: 1
threadsafe: true

handlers:
- url: /
  static_files: www/index.html
  upload: www/index.html

- url: /(.*)
  static_files: www/\1
  upload:www/(.*)
```

More reference information about the app.yaml file can be found in the app.yaml reference documentation.

Creating the index.html file

Create an HTML file that will be served when someone navigates to the root page of your website. Store this file in your www directory.

```html
<html>
 <head>
  <title>Hello, world!</title>
  <link rel="stylesheet" type="text/css" href="/css/style.css">
 </head>
 <body>
  <h1>Hello, world!</h1>
  <p>
    This is a simple static HTML file that will be served from Google App
    Engine.
  </p>
 </body>
</html>
```

Deploying your application to App Engine

When you deploy your application files, your website will be uploaded to App Engine. To deploy your app, run the following command from within the root directory of your application where the app.yaml file is located:

```
gcloud app deploy
```

Optional flags:

- Include the --project flag to specify an alternate Cloud Console project ID to what you initialized as the default in the gcloud tool. Example: --project[YOUR_PROJECT_ID]

- Include the -v flag to specify a version ID, otherwise one is generated for you. Example: -v [YOUR_VERSION_ID]

To learn more about deploying your app from the command line, see Deploying a Python 2 App.

Viewing your application

To launch your browser and view the app at https://*PROJECT_ID*.REGION_ID.r.appspot.com, run the following command:

```
gcloud app browse
```

Viva Questions:

1. What is CSS?

2. What is HTML?

3. Explain about project creation.

4. Name the optional flags.

5. What command is used to deploy the app?

Result:

Thus the GAE launcher used to launch the web applications.

Ex No: 8 SIMULATE A CLOUD SCENARIO USING CLOUDSIM AND RUN A SCHEDULING ALGORITHMTHAT IS NOT PRESENT INCLOUDSIM.

Aim:

To simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.

Procedure:

Basics of Scheduling

In computers, Scheduling is a process of arranging the submitted jobs/task into a very specific sequence of execution. It is an essential characteristic of any software operating environment, which is handled by a very special program known as a scheduler.

Scheduler's main objective is to keep the underlined hardware resources(primarily processor) to be used effectively as well as efficient. In general, the scheduler may prefer to have any of the following scheduling approaches:

- **Space-shared:** In this, the requested resources are allocated dedicatedly to the requesting workload for execution and will be released only on completion. **Space-shared is also known as a batch process scheduling.**
- **Time-shared:** In this, the requested resources would be shared among more than one workload(task). The sharing is done based on time-sliced allocation where each workload is allocated with a required resource for a defined time(e.g., 200 milliseconds). Once the defined time slice is over, the current workload execution paused, and the resource is released. The released resource gets allocated to the next workload for the same defined time slice, and this cycle goes on till the time all the workloads execution is over. Time-shared is also known as round-robin **scheduling.**

Scheduling in Cloud

As cloud computing is the virtualized operating environment, and the virtual machines are the primary computing component which is responsible for the execution of the workloads(tasks). The virtual machine(s) are powered by a physical server host machine (i.e.) hardware. Depending on the requirement of the Virtual Machine(VM) there could be 'one to one' or 'many to one' mapping between the VM and host machine. That means in cloud computing the scheduling is done at both the mapping levels that are:

- Virtual Machine to Host Machines
- Tasks to Virtual Machines

Both of VM to Host as well as Workload(task) to VM mappings may utilize space-share or time-shared or any other specialized scheduling algorithm.

Scheduling in Cloudsim

The Cloudsim simulation toolkit framework has effectively addressed the Scheduling scenario and implemented it as a set of the programmable class hierarchies with parent class as:

1. Vm Scheduler
2. Cloudlet Scheduler

Also, Virtual Machine(VM) and Task(Cloudlet) scheduling are one of the most important and the popular use case to be simulated by researchers using the CloudSim simulation toolkit.

Note: In cloudsim, the task is called as cloudlet, therefore in the following text instead of 'task' we will be using the 'cloudlet'.

Cloudsim Virtual Machine Scheduling

The **VmScheduler** is an abstract class that defines and implements the policy used to share processing power among virtual machines running on a specified host. The hierarchy of the cloudsim virtual machine scheduler classes is as:
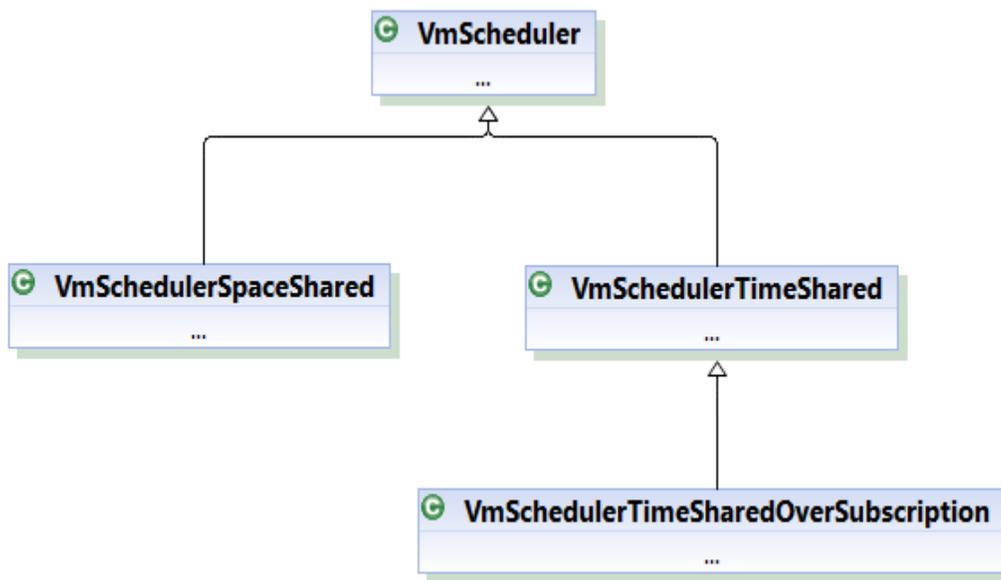
Cloudsim Virtual Machine Scheduler Class Hierarchy

These classes can be located in "*org.cloudbus.cloudsim*" package of cloudsim. The definition of this abstract class is extended to the following types of policies implemented as classes:

- **VmScheduler Time Shared**: This class implements the VM scheduling policy that allocatesoneormoreprocessingelementstoasingleVirtualmachineandallowsthe

sharing of processing elements by multiple virtual machines with a specified time slice. This class also considers the overhead of VM allocation switching(similar to context switching) in policy definition. Here, the VM allocation will fail if the number of processing elements requested is not available. for example, if the VM request for quad-core processor whereas the allocated host has an only dual-core the allocation will fail.

- **VmScheduler Space Shared**: This class implements the VM scheduling policy that allocates one or more processing elements to a single virtual machine, but this policy implementation does not support sharing of processing elements (i.e.) all the requested resources will be used by the allocated VM till the time the VM is not destroyed. Also, Under this allocation policy, if any virtual machine requests a processing element and is not available at that time, the allocation fails.

- **VmScheduler Time Shared Over Subscription**: This is an extended implementation of VMScheduler Time Shared VM scheduling policy, which allows over-subscription of processing elements by the virtual machine(s) (i.e.) the scheduler still allows the allocation of VMs that require more CPU capacity that is available. And this oversubscription results in performance degradation.

The application of the VmScheduler classes is while instantiating the host model. Following is the code snippet used in CloudsimExample1.java from line number 160 to 174:

```
int hostId = 0;
int ram = 2048; // host memory (MB)
long storage = 1000000; // host storage
int bw = 10000;

hostList.add(
            new Host(
                hostId,
                new RamProvisionerSimple(ram),
                new BwProvisionerSimple(bw),
                storage,
                peList,
                new VmSchedulerTimeShared(peList)
        )
); // This is our machine
```

This is where the processing element list is passed as a parameter to the VmSchedulerTimeShared() class call and during the simulation, the cloudsim will simulate the timeshare behavior for the virtual machines. Also, in case you want to test other VmScheduler you may replace it with VmSchedulerTimeShared() call with appropriate parameters, this includes your own designed custom virtual machine scheduler.

The "***CloudletScheduler***" is an abstract class that defines the basic skeleton to implement the policy to be used for cloudlet scheduling to be performed by a virtual machine. The hierarchy of the cloudsim Cloudlet scheduler classes is as:
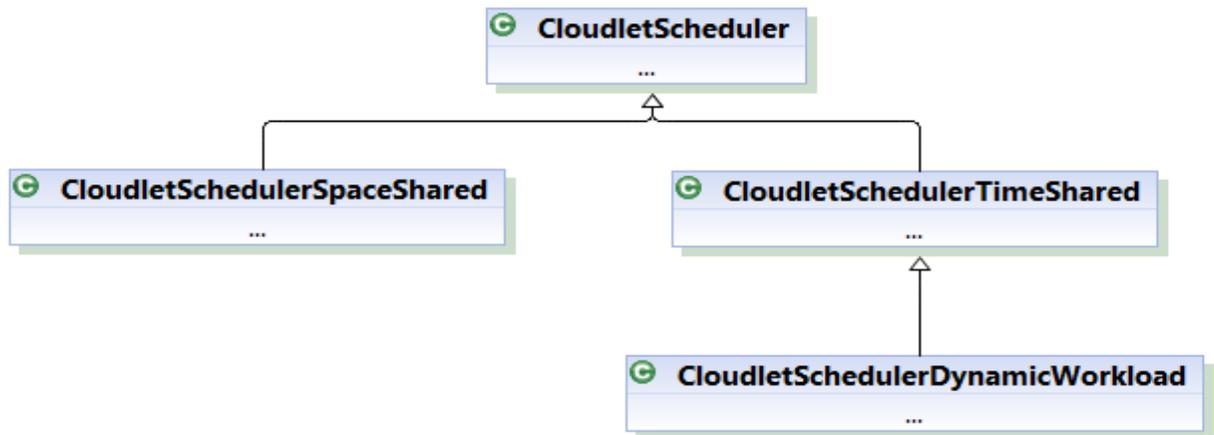
Cloudlet Scheduler Class Hierarchy

These classes again exist in "*org.cloudbus.cloudsim*" package of cloudsim. The definition of this abstract class is extended as the following types of policies implemented as three individual classes in cloudsim:

- **Cloudlet Scheduler Space Shared**: This class implements a policy of scheduling for Virtual machine to execute cloudlet(s) in space shared environment (i.e.) only one cloudlet will be executed on a virtual machine at a time. It means cloudlets share the same queue and requests are processed one at a time per computing core. *Space-sharing* is similar to batch processing.
- **Cloudlet Scheduler Time Shared**: This class implements a policy of cloudlet scheduling for Virtual machines to execute cloudlets in a time-shared environment (i.e.) more than one cloudlet will be submitted to the virtual machine and each will get its specified share of time. It means several requests (cloudlets) are processed at once but they must share the computing power of that virtual machine(by simulating context switching), so they will affect each other's processing time. It basically influences the completion time of a cloudlet in CloudSim. Time-sharing is probably referring to the concept of sharing executing power (such as CPU, logical processor, GPU) and is commonly known as the round-robin scheduling.
- **Cloudlet Scheduler Dynamic Workload:** This implements a special policy of scheduling for virtual machine assuming that there is just one cloudlet which is working as an online service with a different requirement of workload as per the need of peak/off peak user load at a specified period of time.

The application of the Cloudlet Scheduler classes is while instantiating the Vm model. Following is the code snippet used in CloudsimExample1.java from line number 82 to 91:

```
int vmid = 0;
int mips =1000;
long size = 10000; // image size (MB)
int ram = 512; // vm memory (MB)
long bw =1000;
int pesNumber = 1; // number of cpus
String vmm = "Xen"; // VMM name

Vm vm  = new  Vm(vmid, brokerId, mips,  pesNumber, ram, bw, size, vmm, new
CloudletSchedulerTimeShared()); // createVM
```

By instantiating the Cloudlet Scheduler Time Shared() class, the Virtual machine is decided to follow the timeshare(round-robin) approach while simulation for scheduling & executing the Cloudlets. Also, in case you want to test other Cloudlet Scheduler you may replace it with Cloudlet Scheduler TimeShared() call with appropriate parameters, this includes your own designed custom cloudlet scheduler.

Now in case you want to implement your own scheduling policies with respect to Virtual Machine or Cloudlet(s), you may simply extend the VmScheduler or Cloudlet Scheduler class to implement all the abstract methods as specified. This gives you the flexibility to design and implement your own set of algorithms and then later test & optimize during the repetitive simulation runs.

Learn More

In case you missed reading our popular article on Detailed introduction to Cloudsim you may follow CloudSim Simulation Toolkit: An Introduction.

You may subscribe to an online self-paced course named Learn Basics of Cloudsim, The content for this course which will be updated weekly till August 2021.

 Viva questions:

 1. What is cloudSim?
 2. Explain the applications.
 3. What is scheduling?
 4. Name the cloud scheduler.
 5. What is VMscheduler?

Result:

 Thus the cloud scenario using CloudSim and scheduling algorithm in CloudSim executed.

Ex No: 9 MOVING FILES BETWEEN VIRTUAL MACHINES

Aim:

To move the files between virtual machine.

You can move files between virtual machines in several ways:

- You can copy files using network utilities as you would between physical computers on your network. To do this between two virtual machine:
    - Both virtual machines must be configured to allow access to your network. Any of the networking methods (host-only, bridged and NAT) are appropriate.
    - With host-only networking, you copy files from the virtual machines to the host and vice-versa, since host-only networking only allows the virtual machines see your host computer.
    - With bridged networking or NAT enabled, you can copy files across your network between the virtual machines.
- You can create a shared drive, either a virtual disk or a raw partition, and mount the drive in each of the virtual machines.

How to Enable File sharing in VirtualBox.

Step 1. Install Guest Additions on the Guest machine.
Step 2. Configure File Sharing on VirtualBox.

Step 1. Install Guest Additions on the Guest machine.
1. Start the Virtuabox Guest Machine (OS).
2. From Oracle's VM VirtualBox main menu, select **Devices** >**Install Guest Additions***

a. . Open Windows Explorer.

b. Double click at the "CD Drive (X:) VirtualBox Guest additions "to explore its contents.
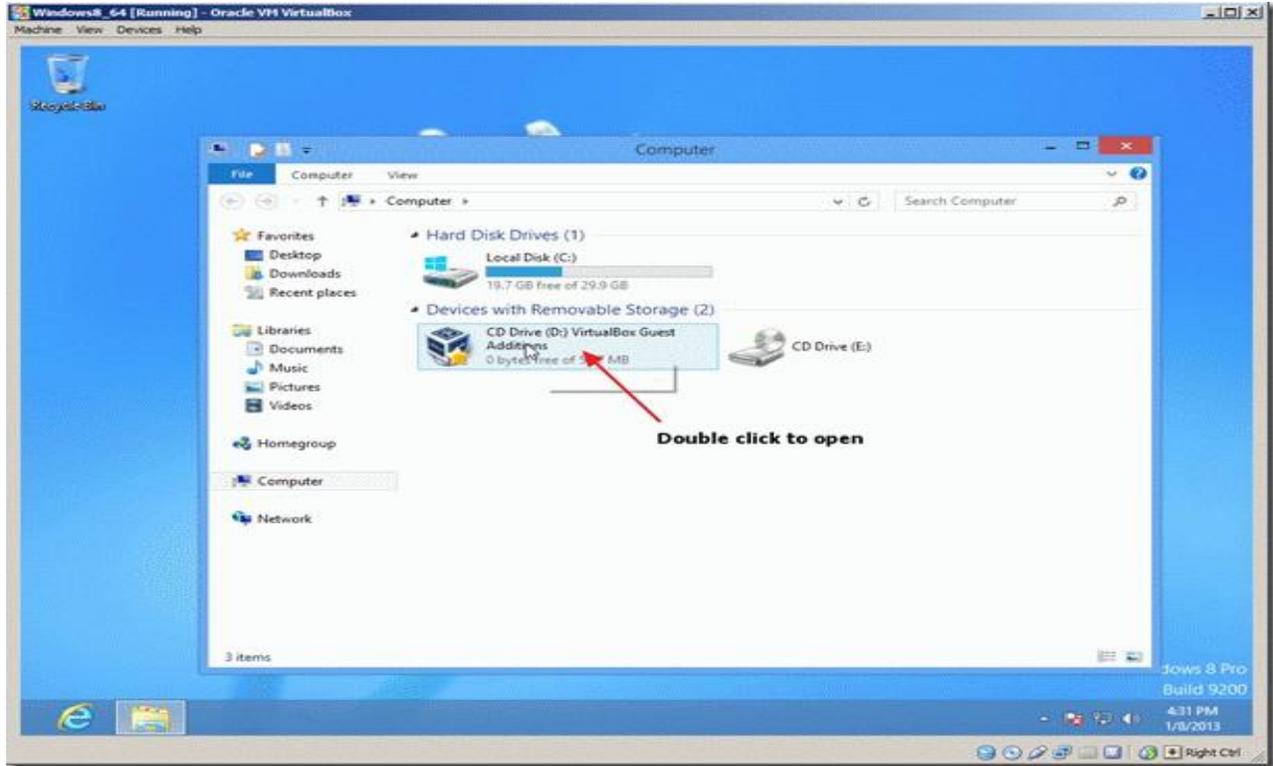
c. Right click at "VBoxWindowsAdditions" application and from the pop-up menu, choose "**Run as administrator**".
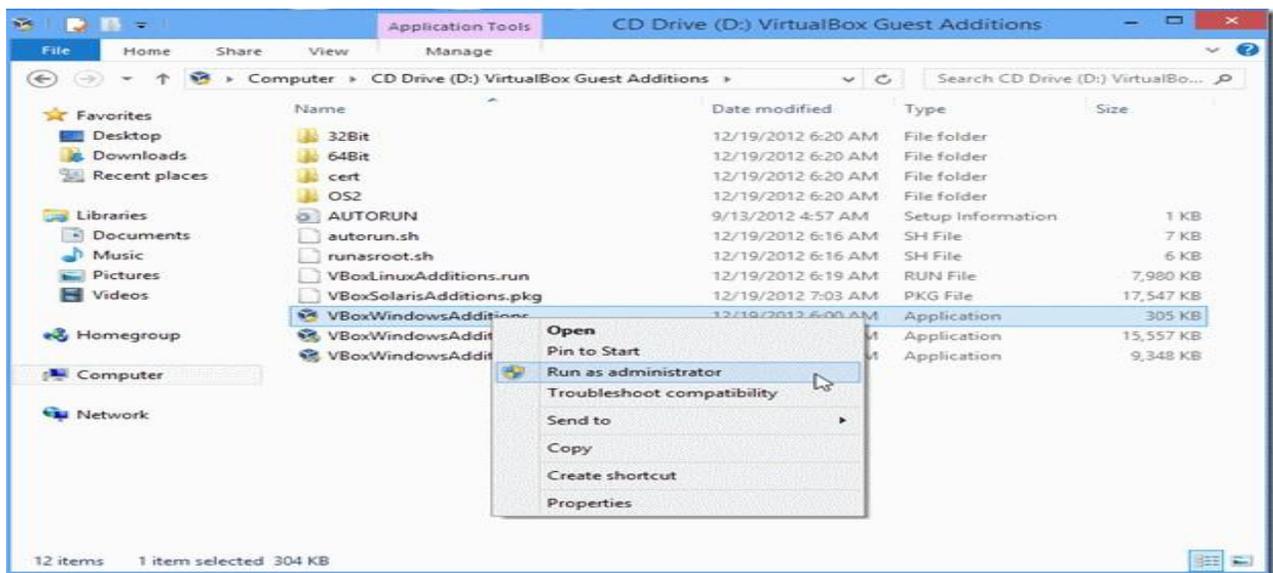
3. Press **Next** and then follow the on screen instructions to complete the Guest Additions installation.

4. When the setup is completed, choose **Finish** and **restart** the Virtuabox guest machine.

Step 2. Setup File Sharing on VirtualBox Guest Machine.

1. From VirtualBox menu click **Devices** and choose **Shared Folders** ->**Shared Folder Settings.**
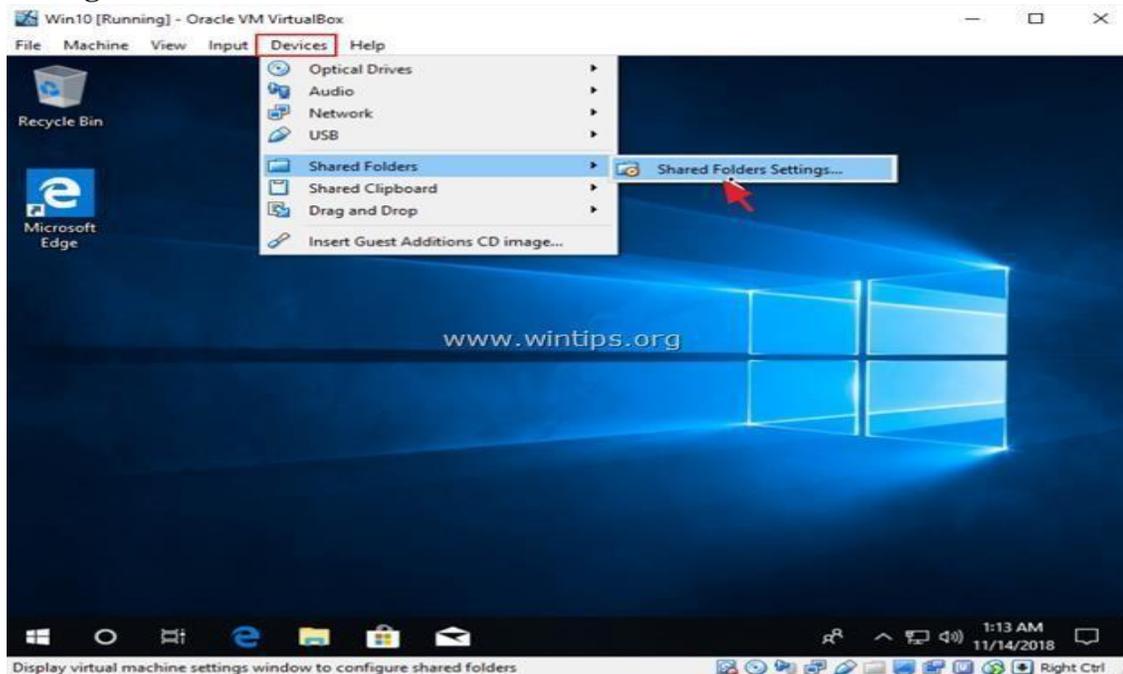
2. Click the **Add new shared folder** icon.

3. Click the drop-down arrow and select **Other**.
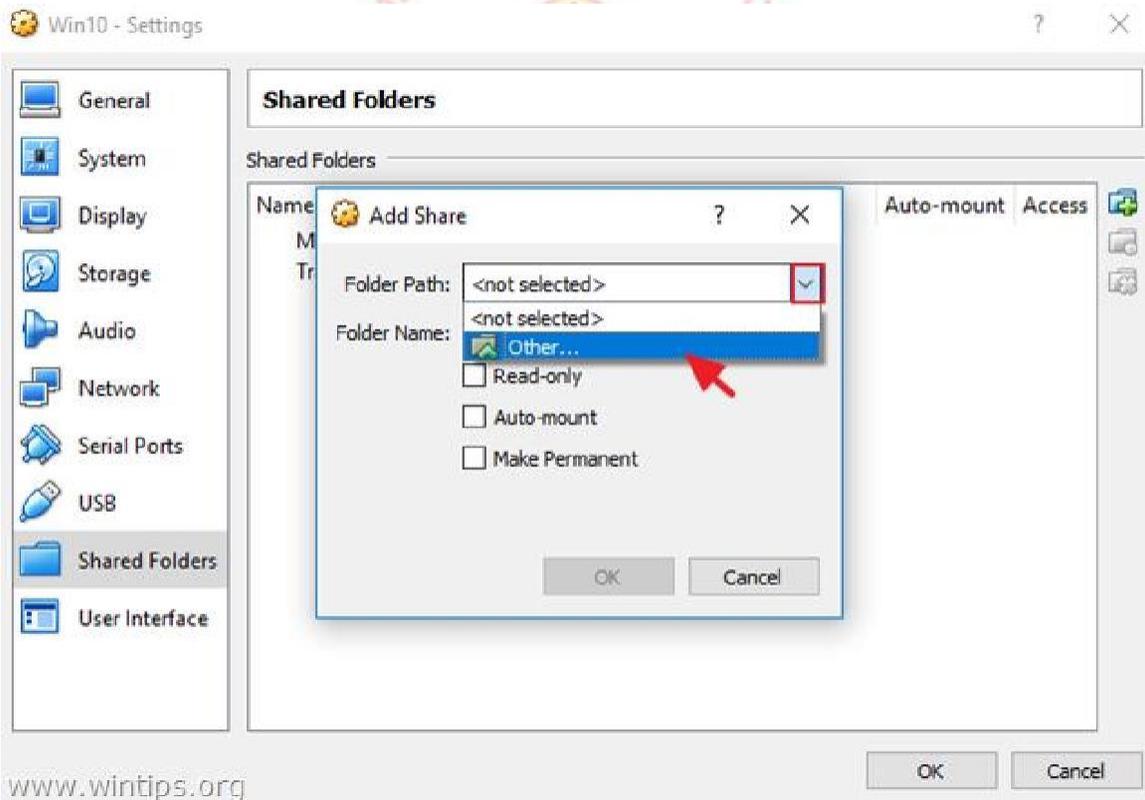
4. Locate and highlight (from the Host OS) the folder that you want to share between the VirtualBox Guest machine and the Host and click Select Folder.*

* Note: To make your life easier, create a new folder for the file sharing, on the Host OS and give it with a recognizable name. (e.g. "Public")
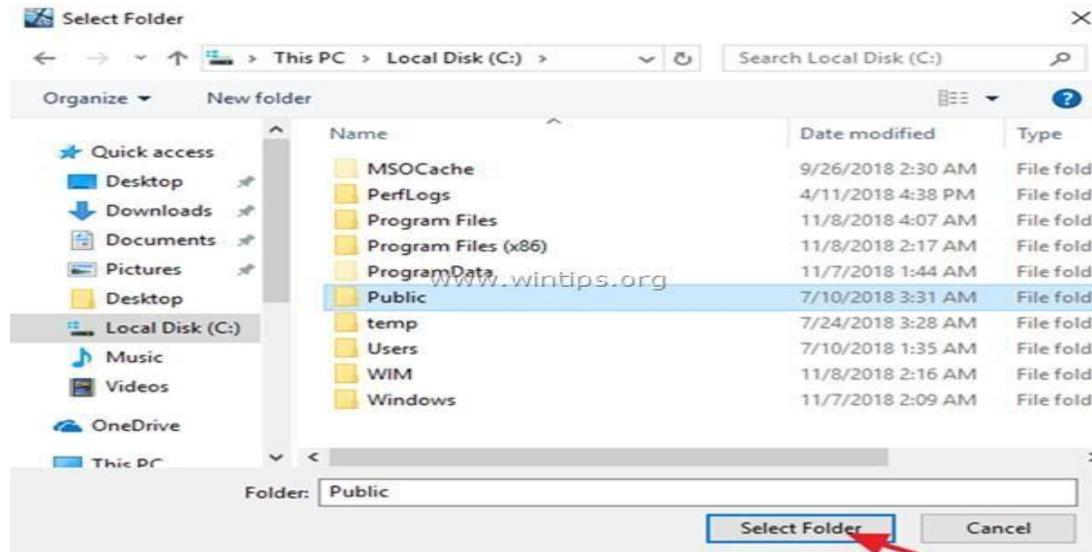
4. Now, in the 'Add Share' options, type a name (if you want) at the 'Folder Name' box, click the **Auto Mount** and the **Make Permanent** checkboxes and click **OK** twice to close the Shared Folder Settings.
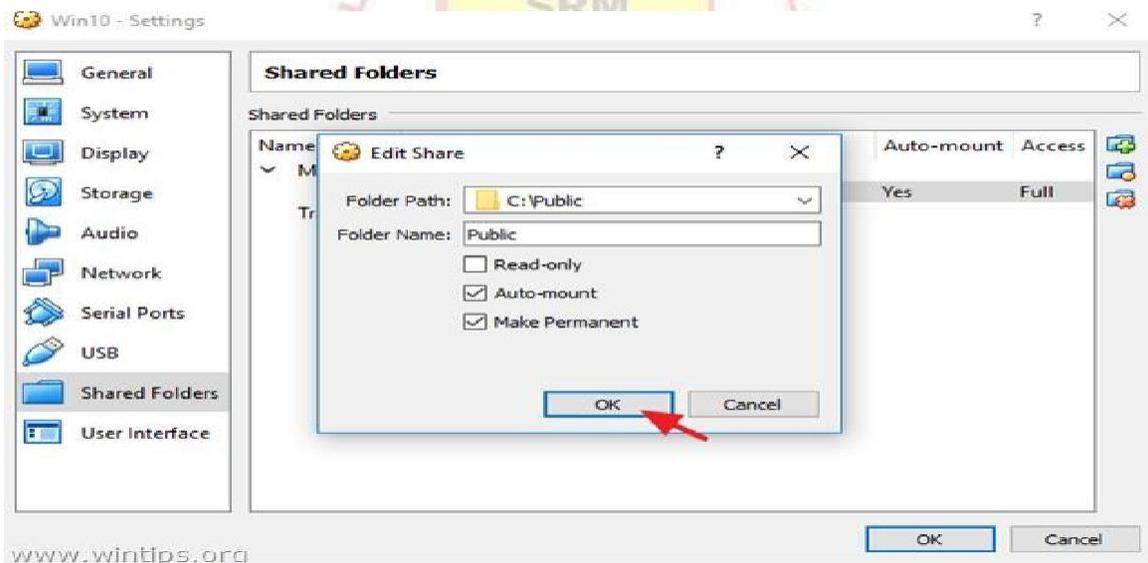
5. You 're done! To access the shared folder from the Guest OS, open Windows Explorer and under the 'Network locations' you should see a new network drive that corresponds to the shared folder on the Host OS.

Viva Questions:

1. How to Enable File sharing in Virtual Box?

2. What is mount?

3. How can we share the folder in VM?

4. What is VM?

5. What is guest OS?

Result:

    Thus the virtual machine files are moved to another VM.

Ex No: 10 KVM AND OPENSTACK INSTALLATION

Aim:

To install the KVM and Open satck in Ubuntu 14.04 version an d creation of virtual machine.

Mandatory prerequisite:

1. Linux 64 bit Operating System (The commands mentioned are for Ubuntu Linux Operating System latest version).

Installing KVM (Hypervisor for Virtualization)

1. Check if the Virtualization flag is enabled in BIOS

Run the command in terminal

egrep -c '(vmx|svm)' /proc/cpuinfo

If the result is any value higher than 0, then virtualization is enabled.

If the value is 0, then in BIOS enable Virtualization – Consult system administrator for this step.

2. To check if your OS is 64bit,

Run the command in terminal

uname -m

If the result is x86_64, it means that your Operating system is 64 bit Operating system.

3. Few KVM packages are available with Linux

installation. To check this, run the

command,

ls /lib/modules/{press tab}/kernel/arch/x86/kvm

nagarajan@JBL01**:~$ ls /lib/modules/4.4.0-21-generic/kernel/arch/x86/kvm**

The three files which are installed in your system will be displayed

kvm-amd.ko   kvm-intel.ko   kvm.ko

4. Install the KVM packages

1. Switch to root (Administrator)user

sudo -i

export http_proxy=http://172.16.0.3:8080

2. To install the packages, run the following commands,

apt-get update

apt-get install qemu-kvm
apt-get install libvirt-bin
apt-get install bridge-utils
apt-get install virt-manager
apt-get install qemu-system

5. To verify your installation, run the command
virsh -c qemu:///system list
it shows output

Id   Name   State

If Vms are running, then it shows name of VM. If VM is not running, the system shows blank output, which means your KVM installation is perfect.

6. Run the command
virsh --connect qemu:///system list --all

7. Working with KVM

run the command
virsh
version(this command displays version of software tools installed)
nodeinfo(this command displays your system information)
quit(come out of the system)

8. To test KVM installation - we can create Virtual machines but these machines are to be done in manual mode. Skipping this, directly install Openstack.

Installation of Openstack

1. add new user named stack – This stack user is the administrator of the open stack
services. To add new user – run the command as root user.
Add user stack
2. run the command
apt-get install sudo -y || install -y sudo

3. Be careful in running the command – please be careful with the syntax. If any error in this following command, the system will crash because of permission errors.

echo "stack ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers

4. Logout the system and login as stack user

5. Run the command (this installs git repopackage)
Run in Root

export http_proxy=http://172.16.0.3:8080
        sudo apt-get install git
6. Run the command (This clones update sd version of dev-stack (which is binary auto-installer package ofOpenstack)
stack@JBL01:/$ **export http_proxy=http://172.16.0.3:8080**
stack@JBL01:/$ **export https_proxy=http://172.16.0.3:8080**
stack@JBL01:/$ **git config --global http.proxy $http_proxy**
stack@JBL01**:/$ git config --global https.proxy $http_proxy**
        git clone http://git.openstack.org/openstack-dev/devstack
        ls (this shows a folder named devstack)
        cd devstack (enter into the folder)
7. create a file called **local.conf**. To do this run the command,
        nano local.conf
stack@JBL01:/devstack$ **sudo nano local.conf**
8. In the file, make the following entry (Contact Your Network Administrator for doubts in these values)

        [[local|localrc]]
FLOATING_RANGE=192.168.1.224/27
FIXED_RANGE=10.11.11.0/24
FIXED_NETWORK_SIZE=256
FLAT_INTERFACE=eth0
ADMIN_PASSWORD=root
DATABASE_PASSWORD=root
RABBIT_PASSWORD=root
SERVICE_PASSWORD=root
SERVICE_TOCKEN=root

9. Save thisfile

    stack@JBL01:/devstack$ **sudo geditstackrc**
 Save this file
Change File Permission:
stack@JBL01:~$ **chown stack * -R**

10. Run the command (This installs Openstack)

    ./stack.sh

11. If any error occurs, then run the command for uninstallation

    ./unstack.sh

        1. update the packages

    apt-get update

         2. Then reinstall the package

    ./stack.sh

12. Open the browser, http://IP address of your machine, you will get the open stack portal.

13. If you restart the machine, then to again start open stack

    open terminal,

        su stack

        cd devstack

        run ./rejoin.sh

14. Again you can access openstack services in the browser, http://IP address of your machine,

VIRTUAL MACHINE CREATION

Launch an instance

1. Log in to the dashboard

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

3. On the Project tab, open the Compute tab and click Instances category.

The dashboard shows the instances with its name, its private and floating IP addresses, size,

status, task, power state, and so on.

4. Click Launch Instance.

5. In the Launch Instance dialog box, specify the following values:

Details tab

    Availability Zone

    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.

    Instance Name

    Assign a name to the virtual machine.

Note

The name you assign here becomes the initial host name of the server. If the name is longer than 63 characters, the Compute service truncates it automatically to ensure dnsmasq works correctly. After the server is built, if you change the server name in the API or change the host name directly, the names are not updated in the dashboard.

Server names are not guaranteed to be unique when created so you could have two instances with the same hostname.

Flavor

Specify the size of the instance to launch.

- Note

The flavor is selected based on the size of the image selected for launching an instance. For example, while creating an image, if you have entered the value in the Minimum RAM (MB) field as 2048, then on selecting the image, the default flavor is m1.small.

Instance Count

To launch multiple instances, enter a value greater than 1. The default is1.

Instance Boot Source

Your options are:

Boot from image

If you choose this option, a new field for Image Name displays. You can select the image from the list.

Boot from snapshot

If you choose this option, a new field for Instance Snapshot displays. You can select the snapshot from the list.

Boot from volume

If you choose this option, a new field for Volume displays. You can select the volume from the list.

Boot from image (creates a new volume)

With this option, you can boot from an image and create a volume by entering the Device Size and Device Name for your volume. Click the Delete Volume on Instance Delete option to delete the volume on deleting the instance.

Boot from volume snapshot (creates a new volume)

Using this option, you can boot from a volume snapshot and create a new volume by choosing Volume Snapshot from a list and adding a Device Name for your volume. Click the Delete Volume on Instance Delete option to delete the volume on deleting the instance.

Image Name

This field changes based on your previous selection. If you have chosen to launch an instance using an image, the Image Name field displays. Select the image name from the dropdown list.

Instance Snapshot

This field changes based on your previous selection. If you have chosen to launch an instance using a snapshot, the Instance Snapshot field displays. Select the snapshot name from the dropdown list.

Volume

This field changes based on your previous selection. If you have chosen to launch an instance using a volume, the Volume field displays. Select the volume name from the dropdown list. If you want to delete the volume on instance delete, check the Delete Volume on Instance Delete option.

Access & Security tab

Key Pair

Specify a key pair.

If the image uses a static root password or a static key set (neither is recommended), you do not need to provide a key pair to launch the instance.

Security Groups

Activate the security groups that you want to assign to the instance.

Security groups are a kind of cloud firewall that define which incoming network traffic is forwarded to instances.

If you have not created any security groups, you can assign only the default security group to the instance.

Networking tab

Selected Networks

To add a network to the instance, click the + in the Available Networks field.

Network Ports tab

Ports

Activate the ports that you want to assign to the instance.

Post-Creation tab

Customization Script Source

Specify a customization script that runs after your instance launches.

Advanced Options tab

Disk Partition

Select the type of disk partition from the dropdown list:

Automatic

Entire disk is single partition and automatically resizes.

Manual

Faster build times but requires manual partitioning.

6. Click Launch.

The instance starts on a compute node in the cloud.

Note

If you did not provide a key pair, security groups, or rules, users can access the instance only from inside the cloud through VNC. Even pinging the instance is not possible without an ICMP rule configured.

You can also launch an instance from the Images or Volumes category when you launch an instance from an image or a volume respectively.

When you launch an instance from an image, OpenStack creates a local copy of the image on the compute node where the instance starts.

For details on creating images, see Creating images manually in the OpenStack Virtual Machine Image Guide.

When you launch an instance from a volume, note the following steps:

•To select the volume from which to launch, launch an instance from an arbitrary image on the volume. The arbitrary image that you select does not boot. Instead, it is replaced by the image on the volume that you choose in the next steps.

To boot a Xen image from a volume, the image you launch in must be the same type, fully virtualized or para virtualized, as the one on the volume.

•Select the volume or volume snapshot from which to boot. Enter a device name. Enter vda for KVM images or xvda for Xen images.
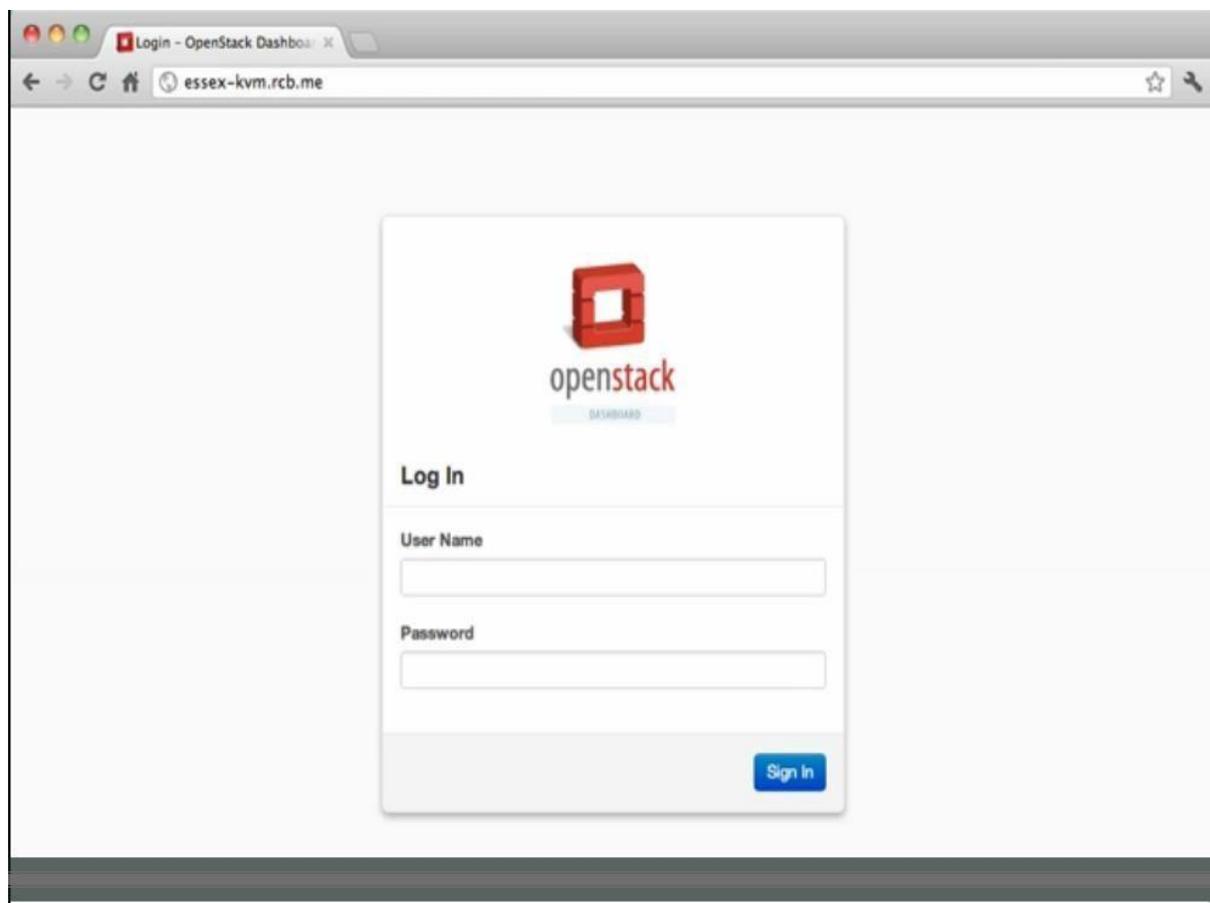
Overview

Select a month to query its usage:

| April | ⇕ | 2012 | ⇕ | Submit |

Active Instances: 2 Active Memory: 1GB This Month's VCPU-Hours: 28.50 This Month's GB-Hours: 0.00

Usage Summary

Download CSV Summary

Project ID	VCPUs	Disk	RAM	VCPU Hours	Disk GB Hours
75c3ff32dc7c4f4385c14738acebace9	1	-	512MB	1.16	0.00
a4588506b03e4323a03cc5d398fc0edc	1	-	512MB	27.34	0.00

Displaying 2 items

All Instances

Instances

	Tenant	Host	Instance Name	IP Address	Size	Status	Task	Power State	Action
☐	jesse	356591-essex-k1	jesse's instance	10.4.128.11	512MB RAM	1 VCPU	0 Disk	Active	None	Running	Edit
☐	admin	356597-essex-k2	dafa	10.4.128.16	512MB RAM	1 VCPU	0 Disk	Active	None	Running	Edit

Displaying 2 items

All Instances

Logged in as: admin   Settings   Sign Out

Instances

Terminate Instances

	Tenant	Host	Instance Name	IP Address	Size	Status	Task	Power State	Actions
☐	jesse	356591-essex-k1	jesse's instance	10.4.128.11	512MB RAM	1 VCPU	0 Disk	Active	None	Running	Edit Instance ▼
☐	admin	356597-essex-k2	dafa	10.4.128.16	512MB RAM	1 VCPU	0 Disk	Active	None	Running	

Displaying 2 items

VNC Console
View Log
Snapshot
Pause Instance
Suspend Instance
Reboot Instance
Terminate Instance

---

openstack
DASHBOARD

Project   Admin

System Panel

Overview

Instances

Services

Flavors

Images

Projects

Users

Quotas

Services

Logged in as: admin   Settings   Sign Out

Services

Filter [   ] 🔍 Filter

Name	Service	Host	Enabled
Compute Service	compute	50.56.12.206	Enabled
S3 Service	s3	50.56.12.206	Enabled
Image Service	image	50.56.12.206	Enabled
Volume Service	volume	50.56.12.206	Enabled
EC2 Service	ec2	50.56.12.206	Enabled
Swift Service	object-store	50.56.12.206	Enabled
Identity Service	identity (native backend)	50.56.12.206	Enabled

Displaying 7 items

essex-kvm.rcb.me/syspanel/flavors/

Flavors

Logged in as: admin   Settings   Sign Out

openstack
DASHBOARD

Project   Admin

System Panel

Overview

Instances

Services

Flavors

Images

Projects

Users

Quotas

Flavors

Create Flavor   Delete Flavors

	ID	Flavor Name	VCPUs	Memory	Root Disk	Ephemeral Disk	Actions
☐	5	m1.xlarge	8	16384	10	160	Delete Flavor
☐	4	m1.large	4	8192	10	80	Delete Flavor
☐	3	m1.medium	2	4096	10	40	Delete Flavor
☐	2	m1.small	1	2048	10	20	Delete Flavor
☐	1	m1.tiny	1	512	-	-	Delete Flavor

Displaying 5 items

---

essex-kvm.rcb.me/syspanel/images/images/

Images

Logged in as: admin   Settings   Sign Out

openstack
DASHBOARD

Project   Admin

System Panel

Overview

Instances

Services

Flavors

Images

Projects

Users

Quotas

Images

Delete Images

	Image Name	Type	Status	Public	Container Format	Actions
☐	cirros-0.3.0-x86_64-rootfs	Image	Active	Yes	BARE	Edit ▾
☐	cirros-0.3.0-x86_64-uec	Image	Active	Yes	AMI	Edit ▾
☐	cirros-0.3.0-x86_64-uec-ramdisk	Image	Active	Yes	ARI	Edit ▾
☐	cirros-0.3.0-x86_64-uec-kernel	Image	Active	Yes	AKI	Edit ▾
☐	ubuntu-12.04-beta1-server-cloudimg-amd64	Image	Active	Yes	AMI	Edit ▾
☐	ubuntu-12.04-beta1-server-cloudimg-amd64-kernel	Image	Active	Yes	AKI	Edit ▾
☐	oneiric-server-cloudimg-amd64-kernel	Image	Active	Yes	AKI	Edit ▾
☐	-server-cloudimg-amd64	Image	Active	Yes	AMI	Edit ▾

Creating USER

Viva Questions:

1. What is KVM?
2. What is flavor?
3. What is open stack?
4. How the VM created in open stack?
5. What is volume?

Result:

Thus the virtual machine is launched in Ubuntu 14.04.

Ex No: 11 FIND PROCEDURE TO SET UP THE ONE NODE HADOOP CLUSTER.

Aim:

To Set up the one node Hadoop Cluster and execute a word count program

Procedure:

1) Installing Java:

Hadoop is a framework written in Java for running applications on large clusters of commodity hardware. Hadoop needs Java 6 or above to work.

Step 1: Download Jdk tar.gz file for linux-62 bit, extract it into "/usr/local"
boss@solaiv[]# cd /opt
boss@solaiv[]# sudo tar xvpzf /home/itadmin/Downloads/jdk-8u5-linux-x64.tar.gz
boss@solaiv[]# cd/opt/jdk1.8.0_05

Step 2:
Open the "/etc/profile" file and Add the following line as per the version
seta environment for Java
Use the root user to save the /etc/proflie or use gedit instead of vi .
The 'profile' file contains commands that ought to be run for login shells

boss@solaiv[]# sudovi /etc/profile
#--insert JAVA_HOME

JAVA_HOME=/opt/jdk1.8.0_05
#--in PATH variable just append at the end of the line PATH=$PATH:$JAVA
_HOME/bin
#--Append JAVA_HOME at end of the export statement export PATH JAVA_HOME
save the file using by pressing "Esc" key followed by :wq!

Step 3: Source the /etc/profile boss@solaiv[]# source /etc/profile

Step 4: Update the java alternatives

By default OS will have a open jdk. Check by "java -version". You will be prompt "openJDK"

If you also have openjdk installed then you'll need to update the java alternatives:

If your system has more than one version of Java, configure which one your system causes by entering the following command in a terminal window

By default OS will have a open jdk. Check by "java -version". You will be prompt "Java HotSpot(TM) 64-BitServer"

boss@solaiv[]#   update-alternatives   --install   "/usr/bin/java"   java
"/opt/jdk1.8.0_05/bin/java"1

boss@solaiv[]# update- alternatives --config java --type selectionnumber:

boss@solaiv[]# java-version

2)   configuressh

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

The need to create a Password-less SSH Key generation based authentication is so

that the master node can then login to slave nodes (and the secondary node) to start/stop them easily without any delays for authentication

If you skip this step, then have to provide password

Generate an SSH key for the user. Then Enable password-less SSH access to yo

sudo apt-get install openssh-server

--You will be asked to enter password, root@solaiv[]# sshlocalhost

root@solaiv[]# ssh-keygenroot@solaiv[]# ssh-copy-id -i localhost

--After above 2 steps, You will be connected without password, root@solaiv[]# sshlocalhost

root@solaiv[]# exit

3) Hadoop installation

Now Download Hadoop from the official Apache, preferably a stable release version of Hadoop 2.7.x and extract the contents of the Hadoop package to a location of your choice.

We chose location as "/opt/"

Step 1: Download the tar.gz file of latest version Hadoop(hadoop-2.7.x) from the official site .
Step 2: Extract(untar) the downloaded file from this commands to /opt/big data

root@solaiv[]# cd /opt
root@solaiv[/opt]#   sudo   tar   xvpzf   /home/itadmin/Downloads/hadoop-2.7.0.tar.gz
root@solaiv[/opt]# cdhadoop-2.7.0/

Like java, update Hadoop environment variable in /etc/profile

boss@solaiv[]# sudovi /etc/profile

#--insert HADOOP_PREFIX HADOOP_PREFIX=/opt/hadoop-2.7.0

#--in   PATH   variable   just   append   at   the   end   of   the   line
PATH=$PATH:$HADOOP_PREFIX/bin

#--Append   HADOOP_PREFIX   at   end   of   the   export   statement   export   PATH
JAVA_HOMEHADOOP_PREFIX

save the file using by pressing "Esc" key followed by :wq!
Step 3: Source the /etc/profile boss@solaiv[]# source /etc/profile
Verify Hadoop installation
boss@solaiv[]# cd $HADOOP_PREFIX
boss@solaiv[]# bin/hadoop version
Modify the Hadoop Configuration Files
Add the following properties in the various hadoop configuration files which is available
under $HADOOP_PREFIX/etc/hadoop/
core-site.xml, hdfs-site.xml, mapred-site.xml & yarn-site.xml

Update Java, hadoop path to the Hadoop environment file
boss@solaiv[]# cd $HADOOP_PREFIX/etc/hadoop

boss@solaiv[]# vi hadoop-env.sh
Paste following line at beginning of the file
export JAVA_HOME=/usr/local/jdk1.8.0_05
export HADOOP_PREFIX=/opt/hadoop-2.7.0
Modify the core-site.xml
boss@solaiv[]# cd $HADOOP_PREFIX/etc/hadoopboss@solaiv[]# vi core-site.xml
Paste following between <configuration> tags
<configuration>
<property>
<name>fs.defaultFS</name>
<value>hdfs://localhost:9000</value>
</property>
</configuration>
Modify the hdfs-site.xml
boss@solaiv[]# vi hdfs-site.xml
Paste following between <configuration> tags
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
</configuration>

YARN configuration - Single Node modify the mapred-site.xml
boss@solaiv[]# cpmapred-site.xml.template mapred-site.xml
boss@solaiv[]# vi mapred-site.xml
Paste following between <configuration> tags
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
Modiy yarn-site.xml
boss@solaiv[]# vi yarn-site.xml
Paste following between <configuration> tags
<configuration>

<property><name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value></property>
</configuration>
Formatting the HDFS file-system via the NameNode

The first step to starting up your Hadoop installation is formatting the Hadoop files system which is implemented on top of the local file system of our "cluster" which includes only our local machine. We need to do this the first time you set up a Hadoop cluster.

Do not format a running Hadoop file system as you will lose all the data currently in the cluster (in HDFS)

root@solaiv[]# cd $HADOOP_PREFIX root@solaiv[]# bin/hadoopnamenode -format
Start Name Node daemon and Data Node daemon: (port 50070)
root@solaiv[]# sbin/start-dfs.sh

To know the running daemons jut type jps or /usr/local/jdk1.8.0_05/bin/jps Start ResourceManager daemon and NodeManager daemon: (port 8088)
root@solaiv[]# sbin/start-yarn.sh
To stop the running process
root@solaiv[]# sbin/stop-dfs.sh
To know the running daemons jut type jps or /usr/local/jdk1.8.0_05/bin/jps Start ResourceManager daemon and NodeManager daemon: (port 8088)
root@solaiv[]# sbin/stop-yarn.sh

PROCEDURE:
Update APT
     test@cs-88:~$ sudo apt-get update
E: Unable to lock directory /var/lib/apt/lists/
Check Backgroud Process :

test@cs-88:~$ ps -ef |grep apt

root	2292	2285	0 11:11 ?	00:00:00 /bin/sh /etc/cron.daily/apt
root	2612	2292	0 11:30 ?	00:00:00 apt-get -qq -y update
root	2615	2612	0 11:30 ?	00:00:00 /usr/lib/apt/methods/http
root	2616	2612	0 11:30 ?	00:00:00 /usr/lib/apt/methods/http
root	2617	2612	0 11:30 ?	00:00:00 /usr/lib/apt/methods/http
root	2619	2612	0 11:30 ?	00:00:00 /usr/lib/apt/methods/gpgv
root	2627	2612	0 11:30 ?	00:00:01 /usr/lib/apt/methods/bzip2

test   2829 2813 011:36 pts/0   00:00:00 grep --color=autoapt

Kill Backgroud Process :
test@cs-88:~$ sudo kill -9 2292 2612 2615 2616 2617 2619 2627 2829

Updaet apt :
test@cs-88:~$ sudo apt-get update

git installation :
root@cs-88:~# sudo apt-get install git

Clone:
root@cs-88:~# git clone https://git.openstack.org/openstack-dev/devstack

root@cs-88:~# ls
devstack
root@cs-88:~# cd devstack
root@cs-88:~/devstack# nano local.conf

[[local|localre]]

        HOST_IP=192.168.4.88
        // FLOATING_RANGE=192.168.1.224/27
        FIXED_RANGE=10.11.12.0/24
        FIXED_NETWORK_SIZE=256
        FLAT_RANGE=eth0
        ADMIN_PASSWORD=linux
        DATABASE_PASSWORD=linux
        RABBIT_PASSWORD=linux
        SERVICE-TOKEN=linux

        Save Nono file:
        control+x

    Hadoop Installation;

        stack@cs-88:~/Downloads$ sudo scp -r * /opt/
        stack@cs-88:~/Downloads$ ls /opt/

        test@cs-88:/opt$ ls

hadoop-2.7.0.tar.gz Hadoop Pseudo-Node.pdf HDFSCommands.pdf jdk-8u60-linux-x64

test@cs-88:/opt$ ls
total 383412
drwxr-xr-x 2 root root   4096 May 6 15:41 ./
drwxr-xr-x 23 root root   4096 May 6 16:34 ../
-rw-r--r-- 1 root root 210343364 May 6 15:41 hadoop-2.7.0.tar.gz
-rw-r--r-- 1rootroot   159315 May 6 15:41 HadoopPseudo-Node.pdf
-rw-r--r-- 1rootroot   43496 May 6 15:41HDFSCommands.pdf
-rw-r--r-- 1 root root 181238643 May 6 15:41 jdk-8u60-linux-x64.gz
-rw-r--r-- 1rootroot   402723 May 6 15:41mrsampledata(1).tar.gz
-rw-r--r-- 1rootroot   402723 May 6 15:41 mrsampledata.tar.gz

Change root user to test user:

test@cs-88:/opt$ sudo chown -Rh test:test /opt/
test@cs-88:/opt$ ll /* display list file with permission
total 383412
drwxr-xr-x 2testtest   4096 May 6 15:41 ./
drwxr-xr-x 23 root root   4096 May 6 16:34 ../
-rw-r--r-- 1 test test 210343364 May 6 15:41 hadoop-2.7.0.tar.gz
-rw-r--r-- 1testtest   159315 May 6 15:41 HadoopPseudo-Node.pdf
-rw-r--r-- 1testtest   43496 May 6 15:41HDFSCommands.pdf
-rw-r--r-- 1 test test 181238643 May 6 15:41 jdk-8u60-linux-x64.gz
-rw-r--r-- 1testtest   402723 May 6 15:41mrsampledata(1).tar.gz
-rw-r--r-- 1testtest   402723 May 6 15:41 mrsampledata.tar.gz
test@cs-88:/opt$

Unzip JAVA
test@cs-88:/opt$ tar -zxvf jdk-8u60-linux-x64.gz

test@cs-88:/opt$ cd jdk1.8.0_60
test@cs-88:/opt/jdk1.8.0_60$ pwd
/opt/jdk1.8.0_60
test@cs-88:/

Set profiloe for JAVA

test@cs-88:/opt/jdk1.8.0_60$ sudo nano /etc/profile

```
JAVA_HOME=/opt/jdk1.8.0_60
HADOOP_PREFIX=/opt/hadoop-2.7.0

PATH=$PATH:$JAVA_HOME/bin
PATH=$PATH:$HADOOP_PREFIX/bin

export PATH JAVA_HOME HADOOP_PREFIX
```

Save: Control +x
Press   y
Press   Enterkey
test@cs-88:/opt/jdk1.8.0_60$ cd ..
test@cs-88:/opt$ pwd
/opt
test@cs-88:/opt$

Unzip hadoop file

test@cs-88:/opt$tar -zxvf hadoop-2.7.0.tar.gz

test@cs-88:/opt$ source /etc/profile

Java Version
test@cs-88:/opt$ java -version

test@cs-88:/$ source /etc/profile
test@cs-88:/$ java -version
java version "1.8.0_60"
Java(TM) SE Runtime Environment (build 1.8.0_60-b27)
Java HotSpot(TM) 64-Bit Server VM (build 25.60-b23, mixed mode)

SSH keygeneration
test@cs-88:/opt$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/test/.ssh/id_rsa):
Created directory '/home/test/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/test/.ssh/id_rsa.

Your public key has been saved in /home/test/.ssh/id_rsa.pub.
The key fingerprint is:
c6:f4:33:42:4d:87:fb:3a:72:29:e9:5b:ce:ee:e9:e4 test@cs-88
The key's randomart image is:
+--[RSA2048]----+
| ... |
| o.. |
| o .. |
| +.. |
| S+. |
| . . o. |
| .oo |
| +*=. |
| .oOE. |
+   +
test@cs-88:/opt$

configure ssh
test@cs-88:/opt$ ssh-copy-id -i localhost
\/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any
that are already installed

test@cs-88:/opt$ sudo apt-get install openssh-server

test@cs-88:/opt$ ssh-copy-id -i localhost

The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is 67:12:a1:69:99:ea:b7:b7:96:b1:f5:4a:29:b5:d0:29.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that
are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it
is to install the new keys
test@localhost'spassword:

Number of key(s) added:1

Now try logging into the machine, with: "ssh 'localhost'"
and check to make sure that only the key(s) you wanted were added.

test@cs-88:/opt$

Configure Hadoop

Verify Hadoop installation

test@cs-88:/opt$ cd $HADOOP_PREFIX
test@cs-88:/opt/hadoop-2.7.0$ pwd
/opt/hadoop-2.7.0
test@cs-88:/opt/hadoop-2.7.0$

test@cs-88:/opt/hadoop-2.7.0$ ls
bininclude libexec   NOTICE.txt sbin
etclib   LICENSE.txt README.txtshare

test@cs-88:/opt/hadoop-2.7.0$ bin/hadoop version
Hadoop 2.7.0
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r
d4c8d4d4d203c934e8074b31289a28724c0842cf
Compiled by jenkins on 2015-04-10T18:40Z
Compiled with protoc 2.5.0
From source with checksum a9e90912c37a35c3195d23951fd18f
This command was run using /opt/hadoop-2.7.0/share/hadoop/common/hadoop-common-
2.7.0.jar
test@cs-88:/opt/hadoop-2.7.0$

Update Java, hadoop path to the Hadoop environment file

test@cs-88:/opt/hadoop-2.7.0$ cd $HADOOP_PREFIX/etc/hadoop
test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$ pwd
/opt/hadoop-2.7.0/etc/hadoop
test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$

test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$ nano hadoop-env.sh
type at last
export JAVA_HOME=/opt/jdk1.8.0_60
exportHADOOP_PREFIX=/opt/hadoop-2.7.0

test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$ nano core-site.xml

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

test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$ nano hdfs-site.xml
```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
</configuration>
```

test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$ cp mapred-site.xml.template mapred-site.xml
test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$ nano mapred-site.xml
```
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

yet another resourcce negociated : yarn

test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$ nano yarn-site.xml
```
<configuration>
<property>
<name>yarn.nodemanager.aux-services</name><value>mapreduce_shuffle</value>
</property>
</configuration>
```

test@cs-88:/opt/hadoop-2.7.0/etc/hadoop$ cd $HADOOP_PREFIX
test@cs-88:/opt/hadoop-2.7.0$ bin/hadoop namenode -format
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

16/05/07 09:24:13 INFO namenode.NameNode: STARTUP_MSG:

```
/***********************************************************
images with txid >= 0
16/05/07 09:24:14 INFO util.ExitUtil: Exiting with status 0
16/05/07 09:24:14 INFO namenode.NameNode: SHUTDOWN_MSG:
/***********************************************************
SHUTDOWN_MSG: Shutting down NameNode at cs-88/127.0.1.1
***********************************************************/
test@cs-88:/opt/hadoop-2.7.0$
test@cs-88:/opt/hadoop-2.7.0$ sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /opt/hadoop-2.7.0/logs/hadoop-test-namenode-
cs-88.out
localhost: starting datanode, logging to /opt/hadoop-2.7.0/logs/hadoop-test-datanode-cs-
88.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 67:12:a1:69:99:ea:b7:b7:96:b1:f5:4a:29:b5:d0:29.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or '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 secondary namenode, logging to /opt/hadoop-2.7.0/logs/hadoop-
test- secondarynamenode-cs-88.out
test@cs-88:/opt/hadoop-2.7.0$

jps- java machines proces status
test@cs-88:/opt/hadoop-2.7.0$ jps
5667 DataNode
5508 NameNode
5863 SecondaryNameNode
5997 Jps
test@cs-88:/opt/hadoop-2.7.0$

test@cs-88:/opt/hadoop-2.7.0$ sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /opt/hadoop-2.7.0/logs/yarn-test-resourcemanager-
cs-88.out
localhost: starting nodemanager, logging to /opt/hadoop-2.7.0/logs/yarn-test-
nodemanager-cs-88.out
test@cs-88:/opt/hadoop-2.7.0$
/* now getting 6 Services */
```

test@cs-88:/opt/hadoop-2.7.0$ jps
5667 DataNode
6084 ResourceManager
5508 NameNode
5863 SecondaryNameNode
6218 NodeManager
6524 Jps
test@cs-88:/opt/hadoop-2.7.0$

test@cs-88:/opt/hadoop-2.7.0$ bin/hdfs dfs -put /opt/file1.txt /user
test@cs-88:/opt/hadoop-2.7.0$

CREATING DIRECGTORY

test@cs-88:/opt/hadoop-2.7.0$ bin/hdfs dfs -mkdir /user
test@cs-88:/opt/hadoop-2.7.0$ bin/hdfs dfs -mkdir /exs
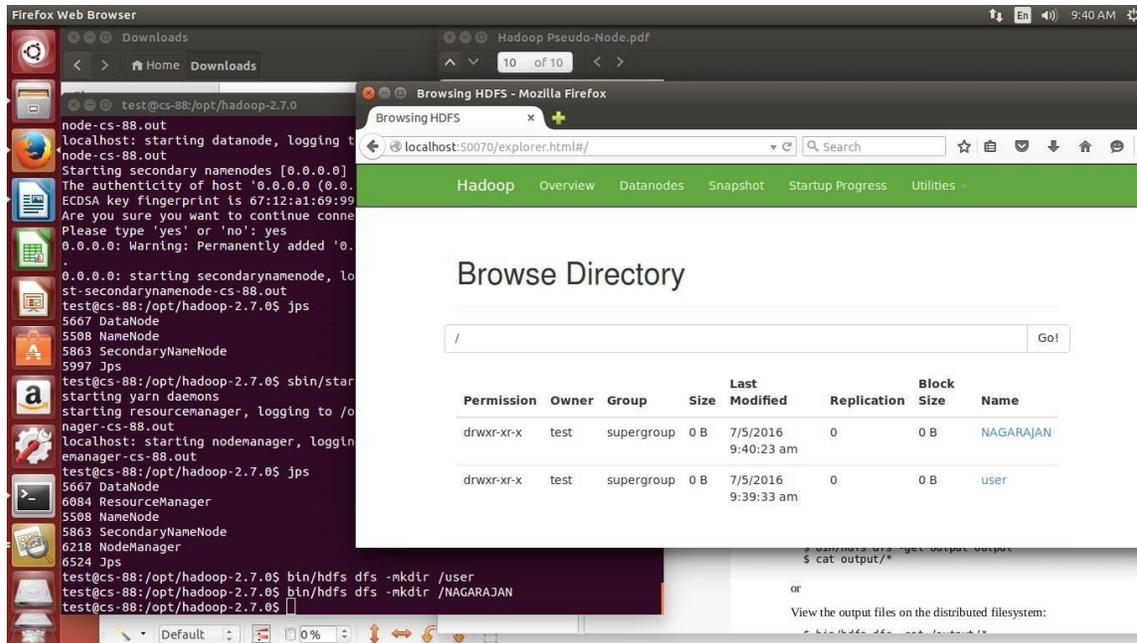
test@cs-88:/opt/hadoop-2.7.0$ cd ..

test@cs-88:/opt$ ls

hadoop-2.7.0   HDFSCommands.pdf   mrsampledata(1).tar.gz

hadoop-2.7.0.tar.gz   jdk1.8.0_60   mrsampledata.tar.gz

Hadoop Pseudo-Node.pdfjdk-8u60-linux-x64.gz

test@cs-88:/opt$ tar zxvfmrsampledata.tar.gz

file2.txt

file5.txt

file1.txt

file4.txt

file3.txt

test@cs-88:/opt$

test@cs-88:/opt/hadoop-2.7.0$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0.jar grep /user/ /op '(CSE)'

Word count program to demonstrate the use of Map and Reduce tasks

Procedure:

1. Format the path.
2. Start the dfs and check the no. of nodes running.
3. Start the yarn and check the no. of nodes running.
4. Open the browser and check whether the hadoop is installed correctly.
5. Add a file and check whether we can view the file.
6. Implement the grep command for the file added and see the result.
7. Implement the word count command for the file added and see the result.
8. After completing the process stop dfs and yarn properly.

Commands:

Install the hadoop cluster by using the commands

1. $sudochown –Rhgee.gee/opt/

2. $nano yarn –site.xml
<configuration>
<property>
<name>yarn.nodemanager.aux-services </name>
<value>mapreduce_shuffle</value>
</property>
</configuration>
3. $cd $Hadoop_prefix
4. $bin /Hadoopnamenode_format
5. $s.bin/start_dfs.sh
6. jps

PROGRAM:

```
packagehadoop;
import java.util.* ;
importjava.io.IOException;
importjava.io.IOException;
importorg.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.* ;
import org.apache.hadoop.io.* ;
importorg.apache.hadoop.m apred.* ;
import org.apache.hadoop.util.* ;
public class ProcessUnits
{
public static class E_EMapper extends MapReduceBaseimplements
Mapper<LongWritable Text,
Text,
IntWritable>
{
public void m ap(LongWritable key, Text value,
OutputCollector<Text, IntWritable> output,
Reporter reporter) throws IOException
{
String line = value.toString();
String lasttoken = null;
StringTokenizer s = new StringTokenizer(line,"\t");
String year = s.nextToken();
while(s.hasMoreTokens())
{
lasttoken=s.nextToken();
}
intavgprice = Integer.parseInt(lasttoken);
output.collect(new Text(year), new IntWritable(avgprice));
}
}
public static class E_EReduce extends MapReduceBaseimplements
Reducer< Text, IntWritable, Text, IntWritable>
{
public void reduce( Text key, Iterator <IntWritable> values,
OutputCollector<Text, IntWritable> output, Reporter reporter) throws
IOException
{
```
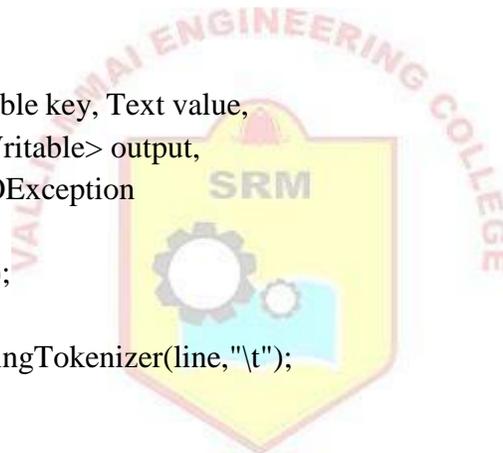
```
int m axavg=30;
intval=Integer.MIN_VALUE;
while (values.hasNext())
{
if((val=values.next().get())>m  axavg)
{
output.collect(key, new IntWritable(val));
}
}
}
}
public static void m ain(String args[])throws Exception
{
JobConfconf = new JobConf(Eleunits.class);
conf.setJobNam e("m ax_eletricityunits");
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(E_EMapper.class);
conf.setCombinerClass(E_EReduce.class);
conf.setReducerClass(E_EReduce.class);
conf.setInputForm at(TextInputFormat.class);
conf.setOutputForm at(TextOutputFormat.class);
FileInputFormat.setInputPaths(conf, new Path(args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));
JobClient.runJob(conf);
}
}
```

OUTPUT:
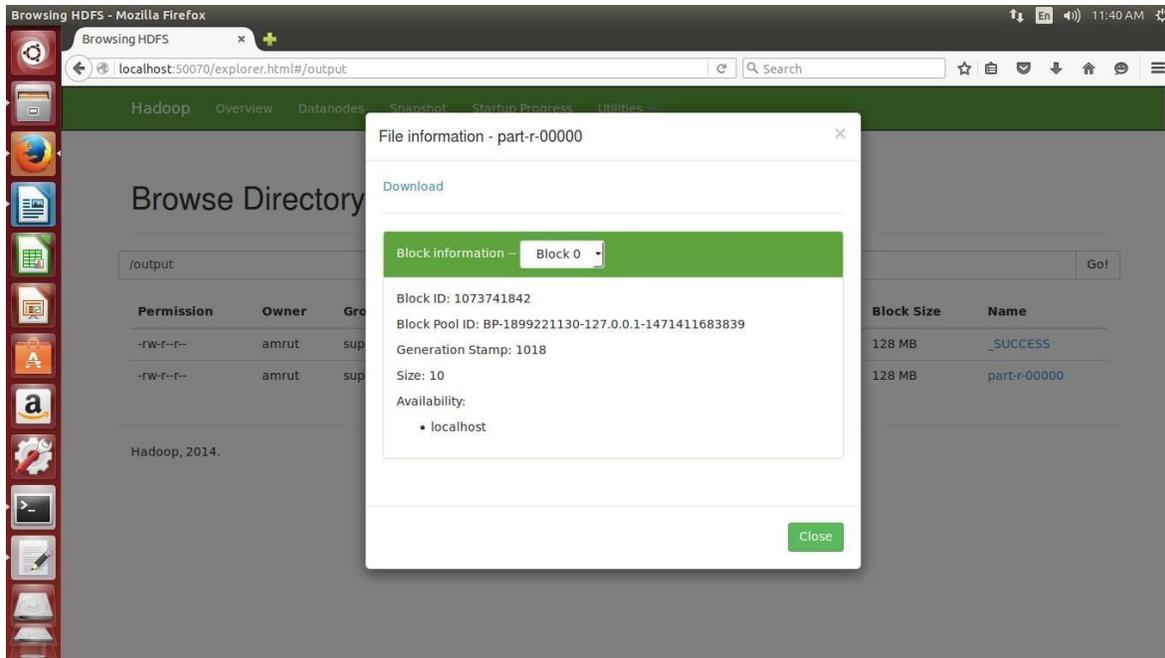
POST LAB QUESTIONS:
1. What is Map?
2. What is Reduce?
3. What is Hadoop core?
4. What is HBase, Pig, Zooeeper and Hive?

Result:

   Thus the map & reduce is successfully done using word count program using one node Hadoop cluster.

TOPIC BEYOND THE SYLLABUS

Ex No : 12 OPEN NEBULA - PRIVATE CLOUD SETUP

Aim:

To install a open nebula and to create a virtual machine

Procedure:

OpenNebula offers a simple but feature-rich and flexible solution to build and manage enterprise clouds and virtualized data centers. OpenNebula is designed to be simple. Simple to install, update and operate by the admins, and simple to use by end users.

Interfaces Provided by OpenNebula

OpenNebula provides many different interfaces that can be used to interact with the functionality offered to manage physical and virtual resources. There are four main different perspectives to interact with OpenNebula:

- Cloud interfaces for Cloud Consumers, like EC2 Query and EBS interfaces, and a simple Sunstone cloud user view that can be used as a self-service portal.
- Administration interfaces for Cloud Advanced Users and Operators, like a Unix-like command line interface and the powerful Sunstone GUI.
- Extensible low-level APIs for Cloud Integrators in Ruby, JAVA and XMLRPC API
- A Marketplace for Appliance Builders with a catalog of virtual appliances ready to run in OpenNebula environments.

OpenNebula Support to Cloud Consumers

OpenNebula provides a powerful, scalable and secure multi-tenant cloud platform for fast delivery and elasticity of virtual resources. Multi-tier applications can be deployed and consumed as pre-configured virtual appliances from catalogs.

- **Image Catalogs:** OpenNebula allows to store disk images in catalogs (termed data stores), that can be then used to define VMs or shared with other users. The images can be OS installations, persistent data sets or empty data blocks that are created within the data store.
- **Network Catalogs:** Virtual networks can be also be organized in network catalogs, and providemeanstointerconnectvirtualmachines.Thiskindofresourcescanbedefinedas

IPv4, IPv6, or mixed networks, and can be used to achieve full isolation between virtual networks.

- **VM Template Catalog:** The template catalog system allows to register virtual machine definitions in the system, to be instantiated later as virtual machine instances.

- **Virtual Resource Control and Monitoring:** Once a template is instantiated to a virtual machine, there are a number of operations that can be performed to control lifecycle of the virtual machine instances, such as migration (live and cold), stop, resume, cancel, poweroff,etc.

- **Multi-tier Cloud Application Control and Monitoring:** OpenNebula allows to define, execute and manage multi-tiered elastic applications, or services composed of interconnected Virtual Machines with deployment dependencies between them and auto-scalingrules

OpenNebula Support to Cloud Consumers

- **Users and Groups**: OpenNebula features advanced multi-tenancy with powerful users and groups management, fine-grained ACLs for resource allocation, and resource quota management to track and limit computing, storage and networking utilization.

- **Virtualization**: Various hypervisors are supported in the virtualization manager, with the ability to control the complete lifecycle of Virtual Machines and multiple hypervisors in the same cloud infrastructure.

- **Hosts**: The host manager provides complete functionality for the management of the physical hosts in the cloud.

- **Monitoring**: Virtual resources as well as hosts are periodically monitored for key performance indicators. The information can then used by a powerful and flexible scheduler for the definition of workload and resource-aware allocation policies.

- **Accounting**: A Configurable accounting system to visualize and report resource usage data, to allow their integration with chargeback and billing platforms, or to guarantee fair share of resources among users.

- **Networking**: An easily adaptable and customizable network subsystem is present in OpenNebula in order to better integrate with the specific network requirements of existing data centers and to allow full isolation between virtual machines that composes a virtualized service.

- **Storage**: The support for multiple data stores in the storage subsystem provides extreme flexibility in planning the storage backend and important performance benefits.

- **Security**: This feature is spread across several subsystems: authentication and authorization mechanisms allowing for various possible mechanisms to identify a authorize users, a powerful Access Control List mechanism allowing different role management with fine grain permission granting over any resource managed by OpenNebula, support for isolation at different levels...

- **High Availability**: Support for HA architectures and configurable behavior in the event of host or VM failure to provide easy to use and cost-effective fail over solutions.

- **Clusters**: Clusters are pools of hosts that share datastores and virtual networks. Clusters are used for load balancing, high availability, and high performance computing.

- **Multiple Zones**: The Data Center Federation functionality allows for the centralized management of multiple instances of OpenNebula for scalability, isolation and multiple-site support.

- **VDCs**. An OpenNebula instance (or Zone) can be further compartmentalized in Virtual Data Centers (VDCs), which offer a fully-isolated virtual infrastructure environment where a group of users, under the control of the group administrator, can create and manage compute, storage and networking capacity.

- **Cloud Bursting**: OpenNebula gives support to build a hybrid cloud, an extension of a private cloud to combine local resources with resources from remote cloud providers. A whole public cloud provider can be encapsulated as a local resource to be able to use extra computational capacity to satisfy peak demands.

- **App Market**: OpenNebula allows the deployment of a private centralized catalog of cloud applications to share and distribute virtual appliances across OpenNebula instances

OpenNebula Support to Cloud builders

OpenNebula offers broad support for commodity and enterprise-grade hypervisor, monitoring, storage, networking and user management services:

User Management: OpenNebula can validate users using its own internal user database based on passwords, or external mechanisms, like ssh, x509, ldap or Active Directory

Virtualization: Several hypervisor technologies are fully supported, like Xen, KVM and VMware.

Monitoring: OpenNebula provides its own customizable and highly scalable monitoring system and also can be integrated with external data center monitoring tools.

Networking: Virtual networks can be backed up by 802.1Q VLANs, ebtables, OpenvSwitch or VMware networking.

Storage: Multiple back ends are supported like the regular (shared or not) file system datastore supporting popular distributed file systems like NFS, Lustre, GlusterFS, ZFS, GPFS, MooseFS...; the VMware datastore (both regular filesystem or VMFS based) specialized for the VMware hypervisor that handle the vmdk format; the LVM datastore to store disk images in a block device form; and Ceph for distributed block device.

Databases: Aside from the original sqlite backend, mysql is also supported.

OpenNebula 4.6 Installation on Ubuntu 14.04 and Kernel Virtual Machine (KVM) Hypervisor

OpenNebula installation using Ubuntu 14.04 as Operating system and KVM as the hypervisor. The two components are present in it are Frontend and Nodes. The Frontend server execute the OpenNebula services and the Nodes functionality is to execute virtual machines. The following steps to execute both components in a single server. The machine has to support virtualization (i.e.) VT enabled system. The command to test the virtualization support is as follows

```
grep -E 'svm|vmx'/proc/cpuinfo
```

Package Layout

opennebula-common: Provides the user and common files

libopennebula-ruby: All ruby libraries

opennebula-node: Prepares a node as an opennebula-node

opennebula-sunstone: OpenNebula Sunstone Web Interface

opennebula-tools: Command Line interface

opennebula-gate: Gate server that enables communication between VMs and OpenNebula

opennebula-flow: Manages services and elasticity

opennebula: OpenNebula Daemon

Step 1. Installation in the Frontend

1. Add therepository

```
sudo wget -q -O- http://downloads.opennebula.org/repo/Ubuntu/repo.key | apt-key add -
sudo echo "deb http://downloads.opennebula.org/repo/Ubuntu/14.04 stable opennebula"
```

```
> /etc/apt/sources.list.d/opennebula.list
```

2. Install the requiredpackages

```
sudo apt-get update
sudo apt-get install opennebula opennebula-sunstone nfs-kernel-server
```

3. Configure and start theservices

There are two main processes that must be started, the main OpenNebula daemon: **oned**, and the

graphical user interface:**sunstone**.

Sunstone listens only in the loopback interface by default for security reasons. To change it

edit /etc/one/sunstone-server.conf and

change :host: 127.0.0.1 to :host: 0.0.0.0.

The command to restart the Sunstone:

```
sudo /etc/init.d/opennebula-sunstone restart
```

4. Configure Network File Service (NFS) (This is not needed if both Frontend and Nodes are in the same

machine)

Export /var/lib/one/ from the frontend to the worker nodes. To do so add the following to

the /etc/exports file in thefrontend:

```
/var/lib/one/ *(rw,sync,no_subtree_check,root_squash)
```

Refresh the NFS export by the following command

```
sudo service nfs-kernel-server restart
```

5. Configure SSH publickey

```
sudo su – oneadmin
cp ~/.ssh/id_rsa.pub ~/.ssh/authorized_keys
```

Add the following snippet to ~/.ssh/config so it doesn't prompt to add the keys to the known_hosts file:.

```
$ cat<< EOT > ~/.ssh/config
Host *
   StrictHostKeyChecking no
   UserKnownHostsFile /dev/null
```

```
EOT

$ chmod 600 ~/.ssh/config
```

Step 2. Installation in the Nodes

1. Install the required packages

```
sudo apt-get install opennebula-node nfs-common bridge-utils
```

2. Configure the network

In DHCP,

edit /etc/network/interfaces

```
auto lo
iface lo inet loopback
auto br0
iface br0 inet dhcp
      bridge_ports eth0
      bridge_fd 9
      bridge_hello2
      bridge_maxage 12
      bridge_stpoff
```

In case of static IP address

```
auto lo
iface lo inet loopback


auto br0
iface br0 inet static
      address192.168.0.10
      network192.168.0.0
      netmask 255.255.255.0
      broadcast 192.168.0.255
      gateway 192.168.0.1
      bridge_ports eth0
      bridge_fd 9
      bridge_hello 2
```

> **bridge_maxage 12**
>
> **bridge_stp off**

Restart the network

sudo /etc/init.d/networking restart

3. Configure NFS (This is not needed if both Frontend and Nodes are in the same machine)

edit the file /etc/fstab as

<Frontend IP>:/var/lib/one/ /var/lib/one/ nfs soft,intr,rsize=8192,wsize=8192,noauto

Mount the NFS

sudo mount /var/lib/one

/* if there is any problem , then it will be firewall issue

4. ConfigureQemu

oneadmin user must be able to manage libvirt as root

 # cat<< EOT >/etc/libvirt/qemu.conf

user = "oneadmin"

group = "oneadmin"

dynamic_ownership = 0

EOT

Restart libvirt

sudo service libvirt-bin restart

Step 3: Start the sunstone from the web browser

http://frontend:9869

One admin password is available in the file ~/.one/one_auth

1. From the command line, to login as one admin in the Frontend ,execute

sudo su - oneadmin

2. AddingHost

To start running VMs, you should first register a worker node for OpenNebula.

```
onehost create localhost -i kvm -v kvm -n dummy
```

Run **onehost list** - Try these command until it's on. If there is any failure, look at /var/log/one/oned.log

3. Adding Virtual resources

To create networks, create a network template file "mynetwork.one" and edit the file as follows:

```
NAME = "private"
TYPE = FIXED
BRIDGE = br0
LEASES = [ IP=192.168.0.100 ]
LEASES = [ IP=192.168.0.101 ]
LEASES = [ IP=192.168.0.102 ]
```

Create resources in opennebula

```
$ onevnet create mynetwork.one
$ oneimage create --name "CentOS-6.5_x86_64" \
   --path "http://appliances.c12g.com/CentOS-6.5/centos6.5.qcow2.gz" \
   --driver qcow2 \
   --datastore default
$ onetemplate create --name "CentOS-6.5" --cpu 1 --vcpu 1 --memory 512 \
   --arch x86_64 --disk "CentOS-6.5_x86_64" --nic "private" --vnc \
   --ssh
```

Monitor the resources are created by running the command **oneimage list**

In order to dynamically add ssh keys to Virtual Machines we must add our ssh key to the user template, by editing the user template:

```
$ EDITOR = vi oneuser update oneadmin
```

Add a new line to the template (cat ~/.ssh/id_dsa.pub)

```
SSH_PUBLIC_KEY="ssh-dss AAAAB3NzaC1kc3MAAACBANBWTQmm4Gt..."
```

4. To run a virtual machine

```
$ onetemplate instantiate "Centos-6.5" --name "My Scratch VM"
```

To test, execute onevm list , VM going from Pending to Prolog to Running. If it fails, check the log as
/var/log/one/<VM_ID>/vm.log

Viva Questions:

1. What is opennebula?
2. How will you create the resource in nebula?
3. What is VDC?
4. What is cluster?
5. Explain cloud bursting.

Result:

Thus the virtual machine is created successfully on open nebula.