MASTER OF COMPUTER APPLICATIONS

PRACTICAL RECORD WORK

ON

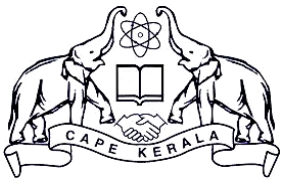
20MCA136 Networking & System Administration Lab

Submitted

By

APARNA MURUKAN

(Reg. No. : VDA20MCA-2011)



DEPARTMENT OF COMPUTER APPLICATIONS
COLLEGE OF ENGINEERING VADAKARA
(CAPE - GOVT. OF KERALA)

APRIL - 2021

DEPARTMENT OF COMPUTER APPLICATIONS
COLLEGE OF ENGINEERING VADAKARA
(CAPE - GOVT. OF KERALA)

CERTIFICATE

Certified that this is a bonafide record of the practical work on the course **20MCA136 NETWORKING & SYSTEM ADMINISTRATION LAB** done by Ms. APARNA MURUKAN (Reg.No.: **VDA20MCA-2011)** Second Semester MCA student of Department of Computer Applications at College of Engineering Vadakara in the partial fulfilment for the award of the degree of Master of Computer Applications (MCA) of APJ Abdul Kalam Technological University (KTU)

(Ms. Nidhina)

(Ms. Anagha)

FACULTY-IN-CHARGE **HEAD OF THE DEPARTMENT**

CEV
04-10-2021

EXAMINERS:

SL.NO	EXPERIMENTS	REMARKS
1	INTRODUCTION TO COMPUTER HARDWARE	
2	STUDY OF TERMINAL BASED TEXT EDITOR SUCH AS VIM	
3	FILE SYSTEM HIERARCHY IN A COMMON LINUX DISTRIBUTION	
4	SHELL SCRIPTING	
5	INSTALLATION AND CONFIGURATION OF LAMP STACK	
6	INSTALLATION AND CONFIGURATION OF COMMON SOFTWARE FRAMEWORKS SUCH AS LARAVEL	
7	BUILD AND INSTALL SOFTWARE FROM SOURCE CODE	
8	INTRODUCTION TO COMMAND LINE TOOLS FOR NETWORKING	
9	ANALYZING PACKET STREAM USING TCPDUMP AND WIRESHARK	
10	INTRODUCTION TO HYPERVISORS,VMS ,DOCKERS	
11	AUTOMATION USING ANSIBLE	

Experiment No.1

Aim

Introduction to computer hardware, physical identification of major components of a computer system such as motherboard, RAM modules, daughter cards, bus slots, SMPS, internal storage devices, interfacing ports.

Result

Computer Hardware

Computer hardware is the physical components that a computer system requires to function. It encompasses everything with a circuit board that operates within a PC or laptop; including the motherboard, graphics card, CPU (Central Processing Unit), ventilation fans, webcam, power supply, and so on. The hardware of a computer is infrequently changed. Computer hardware can be categorized as internal or external components. Internal components include items such as the motherboard, central processing unit (CPU), random access memory (RAM), hard drive, optical drive, heat sink, power supply, transistors, chips, graphics processing unit (GPU), network interface card (NIC) and Universal Serial Bus (USB) ports. These components collectively process or store the instructions delivered by the program or operating system (OS).External components, also called peripheral components, are those items that are often connected to the computer in order to control either its input or output. Common input components include a mouse, monitor, keyboard, microphone, camera, touchpad, joystick, scanner, USB flash driveor memory card.

a) MOTHERBOARD

The motherboard is at the center of what makes a PC work. It houses the CPU and is a hub that all other hardware runs through. The motherboard acts as a brain; allocating power where it's needed, communicating with and coordinating across all other components – making it one of the most important pieces of hardware in a computer. The mother board includes many components such as: central processing unit (CPU), random access memory (RAM), firmware, and internal and external buses.

b) RANDOM ACCESS MEMORY(RAM)

Random access memory (RAM) is fast-access memory that is cleared when the computer is power-down. RAM attaches directly to the motherboard, and is used to store programs that are currently running. RAM is a set of integrated circuits that allow the stored data to be accessed in any order (why it is called random). There are many different types of RAM. Distinctions between these different types include: writable vs. read-only, static vs. dynamic, volatile vs. non-volatile, etc.

c) DAUGHTER CARDS

The daughter board is a computer hardware. It is also known as the piggyback board, riser card, daughter board, daughtercard or daughter card. A daughter board is a printed circuit board which is connected to the motherboard or expansion card. As compared to the motherboard, it is smaller in size. A daughter board does not act as an expansion card. An expansion card adds extra new functions to the computer. But a daughter board that is connected to the motherboard adds or supports the main functions of the motherboard.

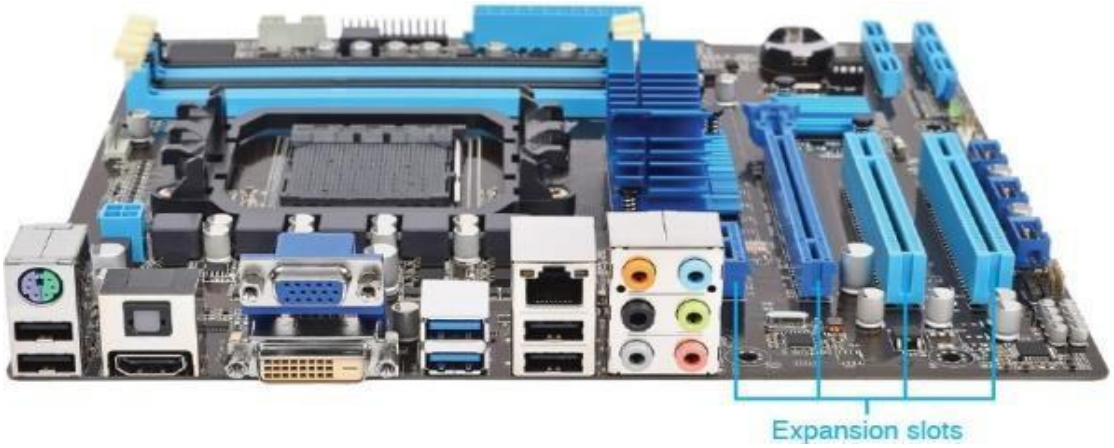
Daughter boards are directly connected to the motherboards. We know that expansion cards are connected to the motherboard by using the bus and other serial interfaces. But daughter board is directly connected to the board by soldering. As an update of the motherboard or expansion card, daughter boards are released to extend the features and services of the motherboard or expansion cards.

d) BUS SLOTS

Bus slots are known as a expansion port, an expansion slot is a connection or port inside a computer on the motherboard or riser card. It provides an installation point for a hardware expansion card to be connected. An expansion slot is a socket on the motherboard that is used to insert an expansion card (or circuit board), which provides additional features to a computer such as video, sound, advanced graphics, Ethernet or memory.

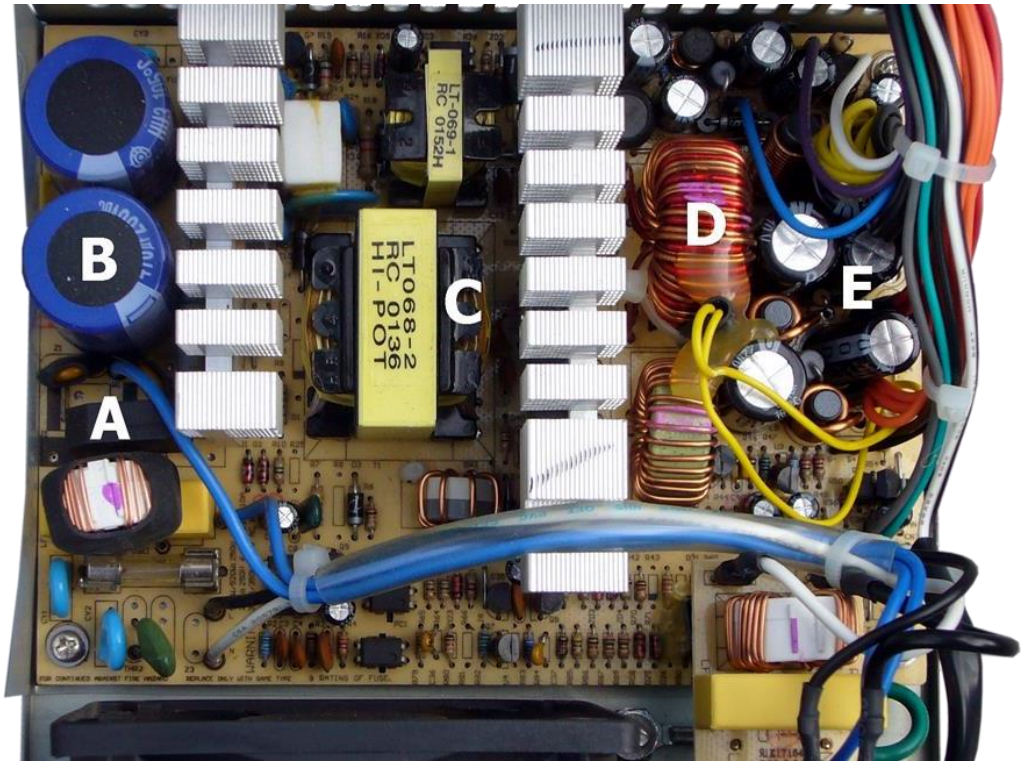
The expansion card has an edge connector that fits precisely into the expansion slot as well as a row of contacts that is designed to establish an electrical connection between the motherboard and the

electronics on the card, which are mostly integrated circuits. Depending on the form factor of the case and motherboard, a computer system generally can have anywhere from one to seven expansion slots. With a backplane system, up to 19 expansion cards can be installed.



e) SWITCHED MODE POWER SUPPLY

A switched-mode power supply (SMPS) is an electronic circuit that converts power using switching devices that are turned on and off at high frequencies, and storage components such as inductors or capacitors to supply power when the switching device is in its non-conduction state.Switching power supplies have high efficiency and are widely used in a variety of electronic equipment, including computers and other sensitive equipment requiring stable and efficient power supply.

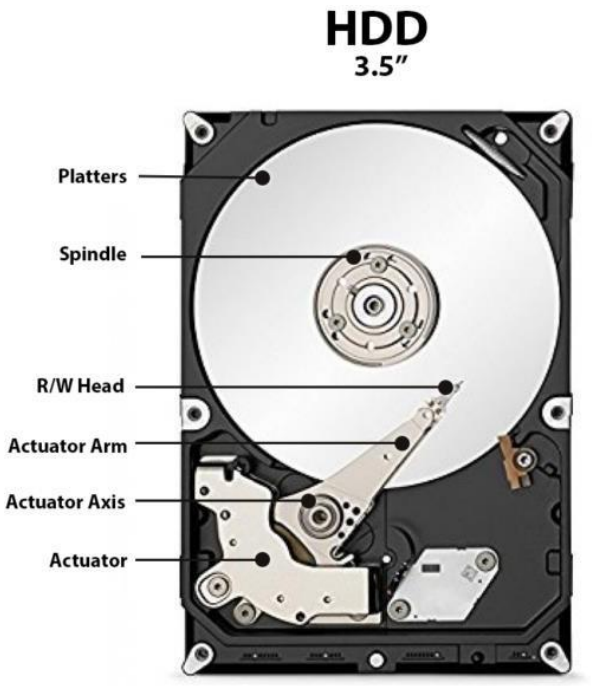


f) INTERNAL STORAGE

Internal storage is hardware that keeps data inside the computer for later use and remains persistent even when the computer has no power. This is the primary storage device used to store a user's files and applications. If a computer has multiple internal hard drives, they are all considered part of the computer's internal storage. There are a few different types of internal storage. Hard disks are the most popular type of internal storage

Hard Disk Drive

A hard disk drive (HDD), hard disk, hard drive, or fixed disk is an electro-mechanical data storage device that stores and retrieves digital data using magnetic storage and one or more rigid rapidly rotating platters coated with magnetic material. The platters are paired with magnetic heads, usually arranged on a moving actuator arm, which read and write data to the platter surfaces. Data is accessed in a random-access manner, meaning that individual blocks of data can be stored and retrieved in any order. HDDs are a type of non-volatile storage, retaining stored data even when powered off.

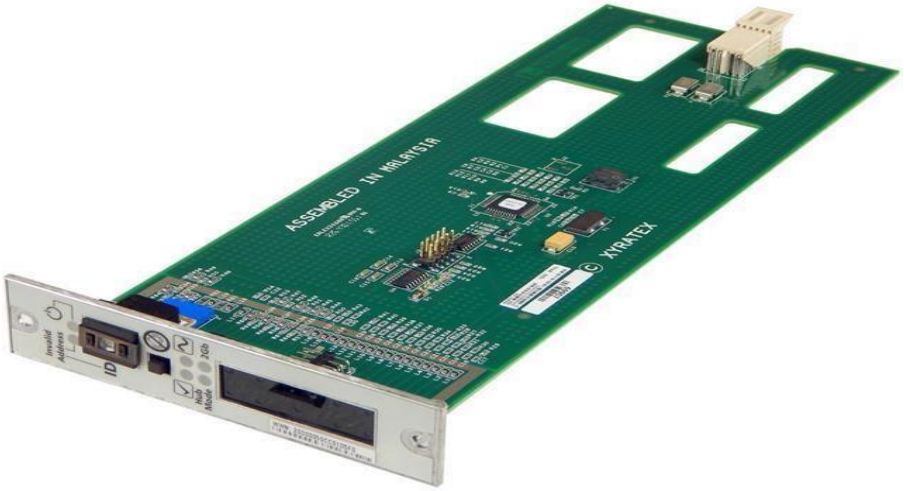


Solid-State Drive

A solid-state drive (SSD) is a solid-state storage device that uses integrated circuit assemblies to store data persistently, typically using flash memory, and functioning as secondary storage in the hierarchy of computer storage. It is also sometimes called a solid-state device or a solid-state disk, even though SSDs lack the physical spinning disks and movable read–write heads used in hard disk drives (HDDs) and floppy disks. Compared with electromechanical drives, SSDs are typically more resistant to physical shock, run silently, and have quicker access time and lower latency. SSDs store data in semiconductor cells.

Disk Array Controller

A disk array controller is a device that manages the physical disk drives and presents them to the computer as logical units. It almost always implements hardware RAID, thus it is sometimes referred to as RAID controller. RAID (Redundant Array of Independent Drives) is a technology that employs the simultaneous use of two or more hard disk drives to achieve greater levels of performance, reliability, and/or larger data volume sizes It also often provides additional disk cache.Disk array controller is often improperly shortened to disk controller.



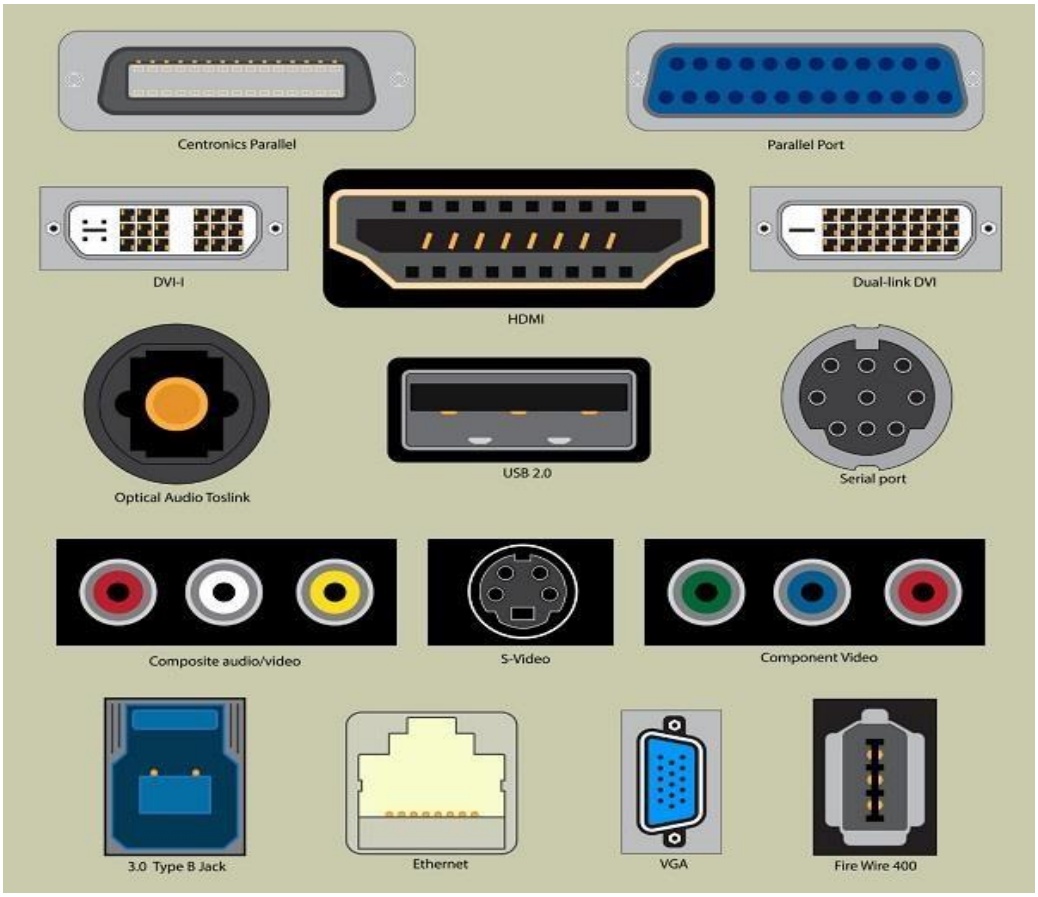
Interfacing Ports

A port is a physical docking point using which an external device can be connected to the computer. It can also be programmatic docking point through which information flows from a program to the computer or over the Internet.

A port has the following characteristics −

- External devices are connected to a computer using cables and ports.
- Ports are slots on the motherboard into which a cable of external device is plugged in.

- Examples of external devices attached via ports are the mouse, keyboard, monitor, microphone, speakers, etc.



Few important types of ports are: −

Serial Port

- Used for external modems and older computer mouse
- Two versions: 9 pin, 25 pin model
- Data travels at 115 kilobits per second

Parallel Port

- Used for scanners and printers
- Also called printer port
- 25 pin model
- IEEE 1284-compliant Centronics port

PS/2 Port

- Used for old computer keyboard and mouse
- Also called mouse port
- Most of the old computers provide two PS/2 port, each for the mouse and keyboard
- IEEE 1284-compliant Centronics port

Universal Serial Bus (or USB) Port

- It can connect all kinds of external USB devices such as external hard disk, printer, scanner, mouse, keyboard, etc.
- It was introduced in 1997.
- Most of the computers provide two USB ports as minimum.
- Data travels at 12 megabits per seconds.
- USB compliant devices can get power from a USB port.

VGA Port

- Connects monitor to a computer's video card.
- It has 15 holes.
- Similar to the serial port connector. However, serial port connector has pins, VGA port has holes.

Power Connector

- Three-pronged plug.
- Connects to the computer's power cable that plugs into a power bar or wall socket.

Firewire Port

- Transfers large amount of data at very fast speed.
- Connects camcorders and video equipment to the computer.
- Data travels at 400 to 800 megabits per seconds.
- Invented by Apple.
- It has three variants: 4-Pin FireWire 400 connector, 6-Pin FireWire 400 connector, and 9-Pin FireWire 800 connector.

Modem Port

- Connects a PC's modem to the telephone network.

Ethernet Port

- Connects to a network and high speed Internet.
- Connects the network cable to a computer.
- This port resides on an Ethernet Card.
- Data travels at 10 megabits to 1000 megabits per seconds depending upon the network bandwidth.

Game Port

- Connect a joystick to a PC
- Now replaced by USB

Digital Video Interface, DVI port

- Connects Flat panel LCD monitor to the computer's high-end video graphic cards.
- Very popular among video card manufacturers.

Sockets

- Sockets connect the microphone and speakers to the sound card of the computer.

Experiment No.2

Aim

Study of a terminal based text editor such as Vim or Emacs. Basic Linux commands, familiarity with following commands/operations expected

1. man
2. ls, echo, read
3. more, less, cat,
4. cd, mkdir, pwd, find
5. mv, cp, rm ,tar
6. wc, cut, paste
7. head, tail, grep, expr 8 chmod, chown
9. Redirections & Piping
10. useradd, usermod, userdel, passwd
11. df,top, ps
12. ssh, scp, ssh-keygen, ssh-copy-id

Result

Text Editor

A text editor is program that allows you to open, view, and edit plain text files. Unlike word processors, text editors do not add formatting to text, instead focusing on editing functions for plain text. Text editors are used by a wide variety of people, for a wide variety of purposes

Vim is a highly configurable text editor built to enable efficient text editing. Vim is acronym for Vi IMproved. It is an improved version of the vi editor distributed with most UNIX systems .Vim is often called a "programmer's editor" .It is free and open source text editor written by Bram Moolenaar. It was first released in 1991 for UNIX variants and its main goal was to provide enhancement to the Vi editor, which was released way back in 1976.

Some common Unix Text editors are : vim, nano, GUI editors , emacs, mousepad , xedit , Pico, Emacs , Xemacs – a GUI-based version of Emacs., vi – a mostly terminal-based text editor

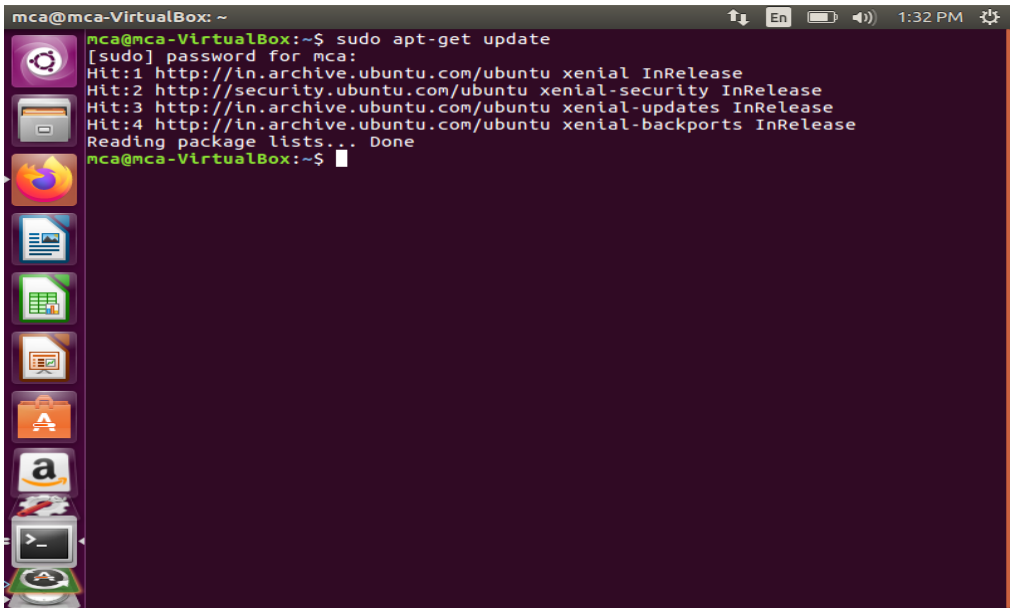
Some of the important features of Vim −

- Its memory footprint is very low

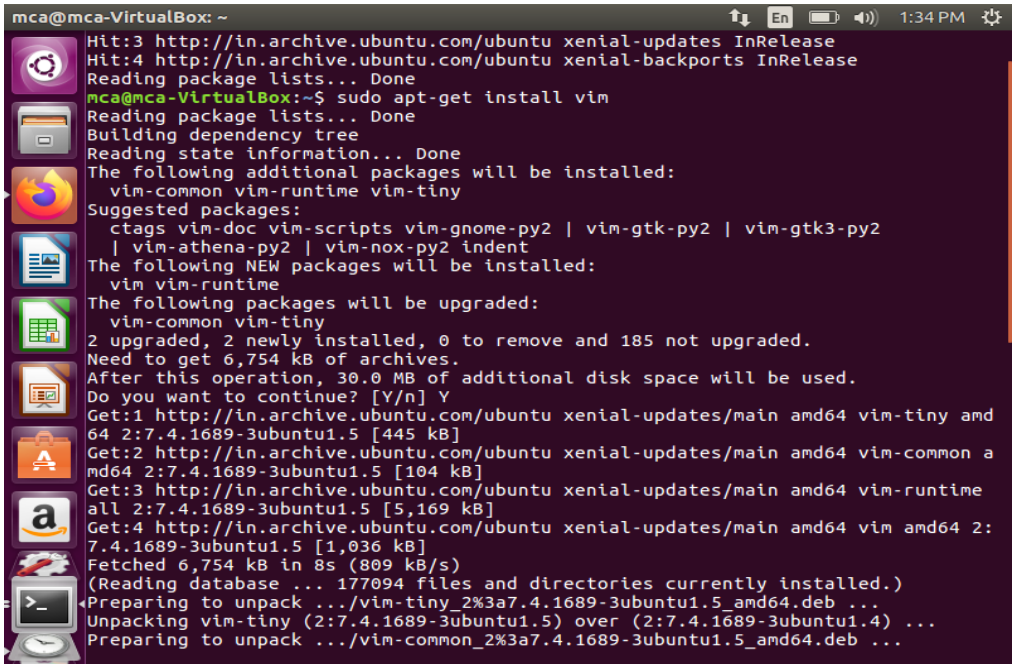
- It is command centric. You can perform complex text related task with few commands
- It is highly configurable and uses simple text file to store its configuration
- There are many plug-in available for Vim. Its functionality can be extended in great manner using these plug-in
- It supports multiple windows. Using this feature screen can be split into multiple windows
- Same as multiple windows, it also supports multiple buffers
- It supports multiple tabs which allows to work on multiple files

Vim Installation

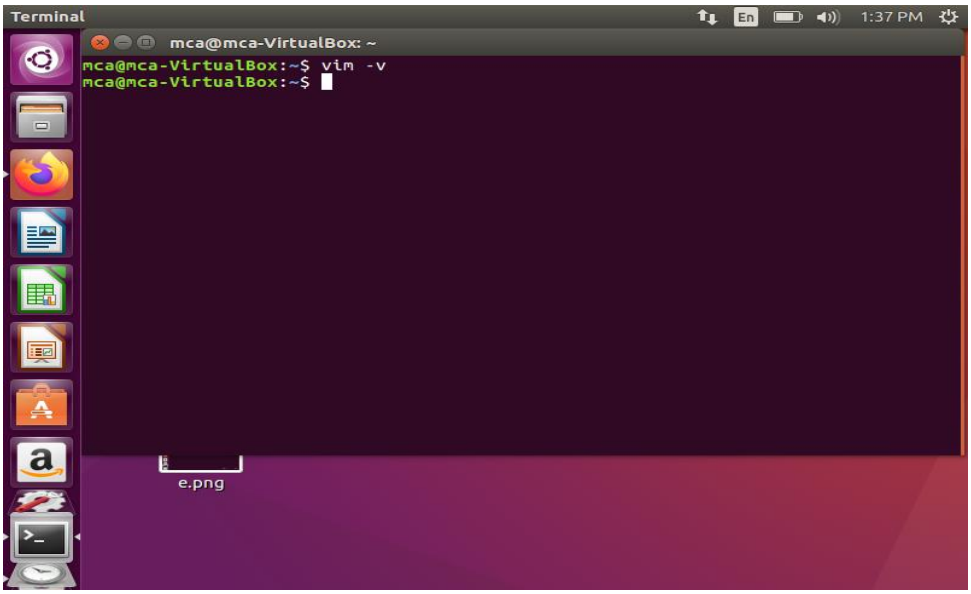
➢ Open terminal and type the command *sudo apt-get update*

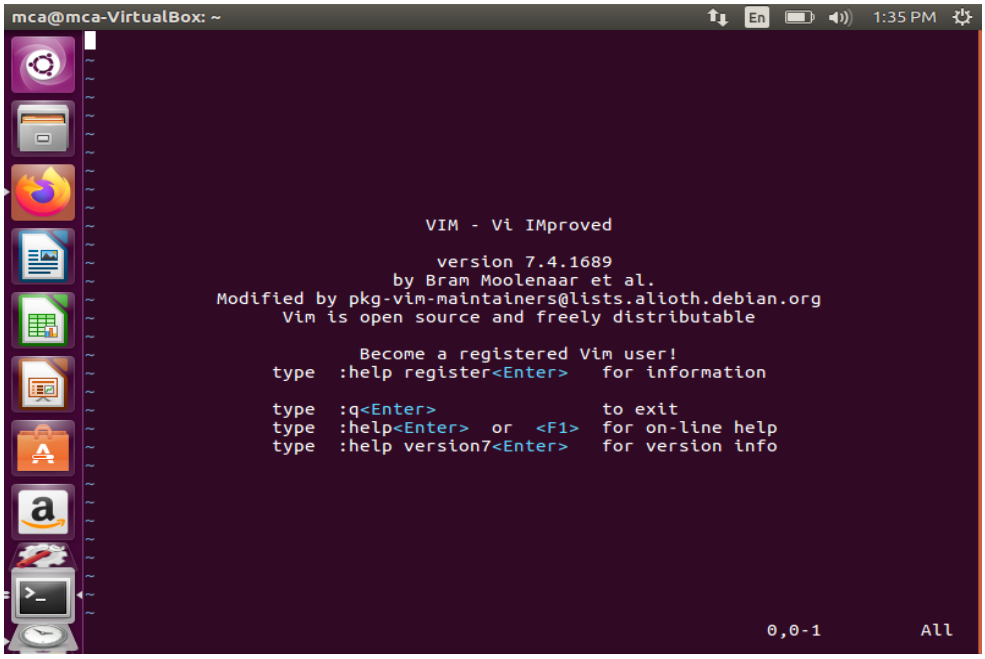


➢ Install vim using the command *sudo apt-get install vim*



➢ Then type *vim –v*





Modes in VIM editor

There are three modes in vim editor.

- Command Mode
- Insert Mode
- Last Line Mode

Command mode

When we first start editing a file using the Vim editor, the editor will be opened in a command mode.We can issue many commands that allow us to insert, append, delete text or search and navigate within our file.When using command mode we cannot insert text immediately.We first need to issue an insert(i), append(a), or open(o) command to insert the text in the file.

Insert mode

When we issue an insert,append,or open command, we will be in insert mode.Once in an insert mode we can type text into our file or navigate within the file.We can toggle between the command mode and the insert mode by pressing the ESC key.

Last Line Mode

The last line mode normally is used to perform operations like quitting the Vim session or savig a file.To go to the last line mode we first need to be in the command mode.From command mode we can go to last line mode by pressing the colon(:) key.After pressing this key,we will see a colon character at the beginning of the last line of our editor window with a

cursor blinking near it. This indicates that the editor is ready for accepting a ‘last line command’.

It is possible to toggle back to the command mode from the last line mode by pressing the ESC key twice or by pressing the backspace key until the initial ‘ : ‘ character is gone along with all the characters that we had typed or by simply pressing the ENTER key.

Vim Commands

1.<u>Basic Vim commands</u> : The most simple commands allow you to open and close documents as well as saving them.

- :help [keyword] - Performs a search of help documentation for whatever keyword you enter
- :e [file] - Opens a file, where [file] is the name of the file you want opened
- :w - Saves the file you are working on
- :w [filename] - Allows you to save your file with the name you've defined
- :wq - Save your file and close Vim
- :q! - Quit without first saving the file you were working on

2. <u>Vim commands for movement</u> : In movement commands, putting a number in front of the command can increase the number of times a task is completed.

- h - Moves the cursor to the left
- l - Moves the cursor to the right
- j - Moves the cursor down one line
- k - Moves the cursor up one line
- H - Puts the cursor at the top of the screen
- M - Puts the cursor in the middle of the screen
- L - Puts the cursor at the bottom of the screen
- w - Puts the cursor at the start of the next word
- b - Puts the cursor at the start of the previous word
- e - Puts the cursor at the end of a word
- 0 - Places the cursor at the beginning of a line
- $ - Places the cursor at the end of a line
-) - Takes you to the start of the next sentence
- (- Takes you to the start of the previous sentence
- } - Takes you to the start of the next paragraph or block of text
- { - Takes you to the start of the previous paragraph or block of text
- Ctrl + f - Takes you one page forward
- Ctrl + b - Takes you one page back
- gg - Places the cursor at the start of the file
- G - Places the cursor at the end of the file

- # - Where # is the number of a line, this command takes you to the line specified

3.<u>Vim commands for editing</u> : The command for copying a word is yw, which stands for yank word, and the command for pasting whatever has been copied is p, meaning put.

- yy - Copies a line
- yw - Copies a word
- y$ - Copies from where your cursor is to the end of a line
- v - Highlight one character at a time using arrow buttons or the h, k, j, l buttons
- V - Highlights one line, and movement keys can allow you to highlight additional lines
- p - Paste whatever has been copied to the unnamed register
- d - Deletes highlighted text
- dd - Deletes a line of text
- dw - Deletes a word
- D - Deletes everything from where your cursor is to the end of the line
- d0 - Deletes everything from where your cursor is to the beginning of the line
- dgg - Deletes everything from where your cursor is to the beginning of the file
- dG - Deletes everything from where your cursor is to the end of the file
- x - Deletes a single character
- u - Undo the last operation; u# allows you to undo multiple actions
- Ctrl + r - Redo the last undo
- . - Repeats the last action

4. <u>Vim commands for searching text</u> : Vim allows you to search your text and find and replace text within your document.

- /[keyword] - Searches for text in the document where keyword is whatever keyword, phrase or string of characters you're looking for
- ?[keyword] - Searches previous text for your keyword, phrase or character string
- n - Searches your text again in whatever direction your last search was
- N - Searches your text again in the opposite direction
- :%s/[pattern]/[replacement]/g - This replaces all occurrences of a pattern without confirming each one
- :%s/[pattern]/[replacement]/gc - Replaces all occurrences of a pattern and confirms each one

5. <u>Vim commands for working with multiple files</u> : We can also edit more than one text file at a time. Vim gives you the ability to either split your screen to show more than one file at a time or you can switch back and forth between documents.

- :bn - Switch to next buffer
- :bp - Switch to previous buffer
- :bd - Close a buffer
- :sp [filename] - Opens a new file and splits your screen horizontally to show more than one buffer
- :vsp [filename] - Opens a new file and splits your screen vertically to show more than one buffer
- :ls - Lists all open buffers
- Ctrl + ws - Split windows horizontally
- Ctrl + wv - Split windows vertically
- Ctrl + ww - Switch between windows
- Ctrl + wq - Quit a window
- Ctrl + wh - Moves your cursor to the window to the left
- Ctrl + wl - Moves your cursor to the window to the right
- Ctrl + wj - Moves your cursor to the window below the one you're in
- Ctrl + wk - Moves your cursor to the window above the one you're in

6. Marking text (visual mode) : Visual mode allows you to select a block of text in Vim. Once a block of text is selected you can use visual commands to perform actions on the selected text such as deleting it, copying it, etc.

- v - starts visual mode, you can then select a range of text, and run a command
- V - starts linewise visual mode (selects entire lines)
- Ctrl + v - starts visual block mode (selects columns)
- ab - a block with ()
- aB - a block with {}
- ib - inner block with ()
- iB - inner block with {}
- aw - mark a word
- Esc - exit visual mode

Once you've selected a particular range of text, you can then run a command on that text such as the following:

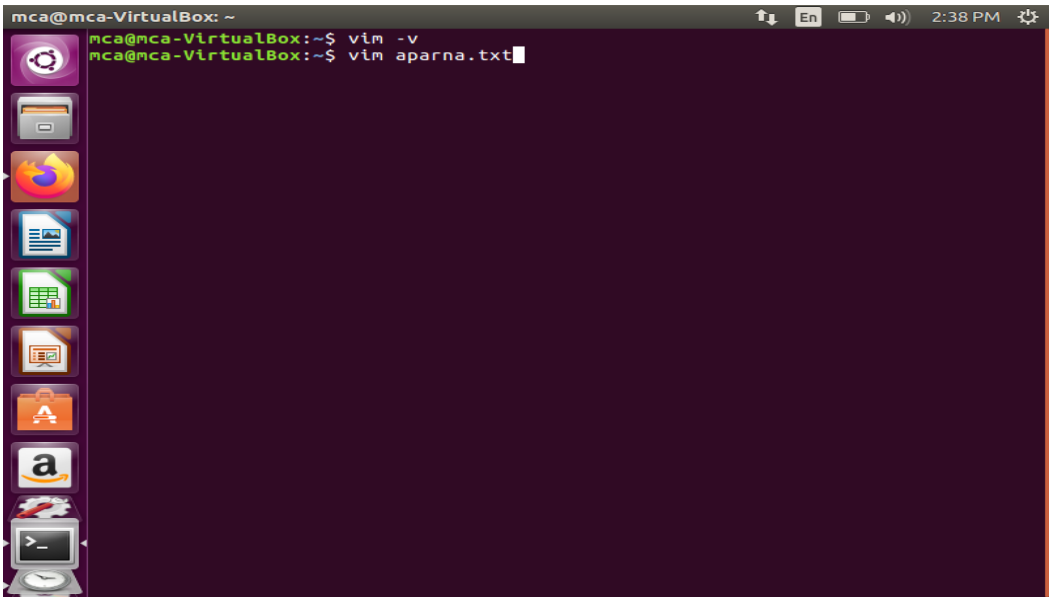
- d - delete marked text
- y - yank (copy) marked text
- shift text right
- < - shift text left
- ~ - swap case (upper or lower)

7. Tab pages : Just like any browser, you can also use tabs within Vim. This makes it incredibly easy to switch between multiple files while you're making some code changes instead of working in one single file, closing it, and opening a new one. Below are some useful Vim commands for using tab pages:

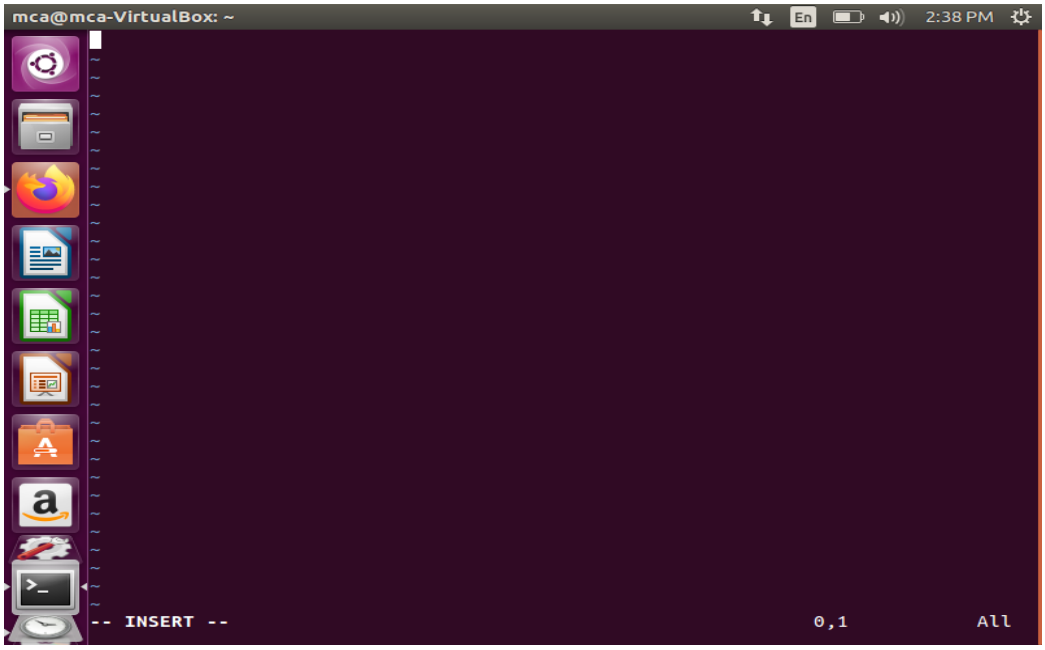
- :tabedit file - opens a new tab and will take you to edit "file"
- gt - move to the next tab
- gT - move to the previous tab
- #gt - move to a specific tab number (e.g. 2gt takes you to the second tab)
- :tabs - list all open tabs
- :tabclose - close a single tab

Simple Vim workflow example

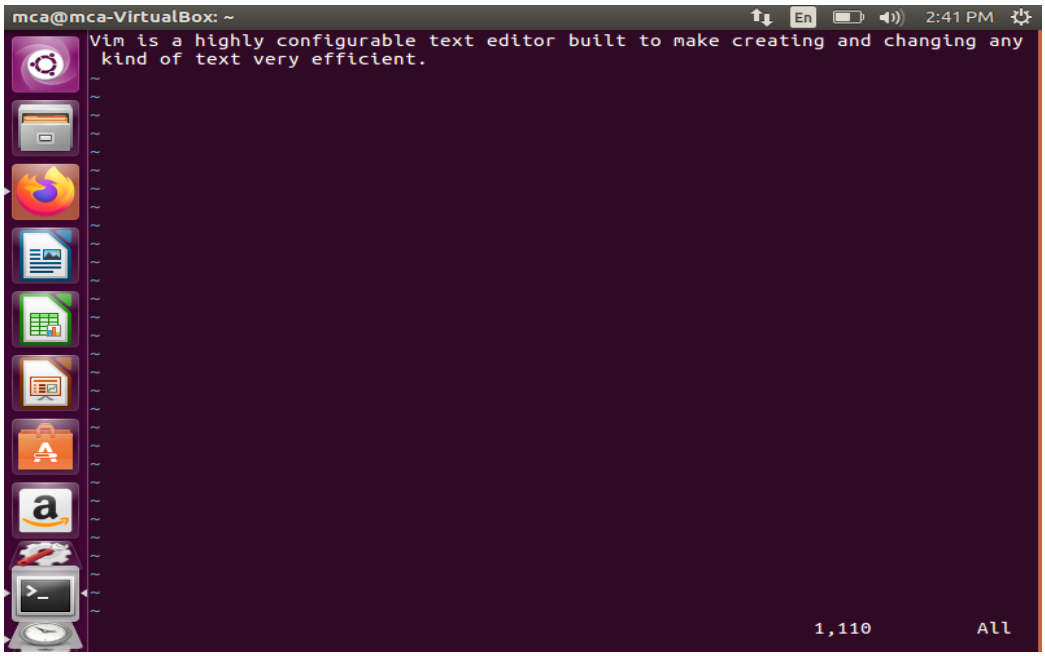
➢ Open a new or existing file with *vim filename*.



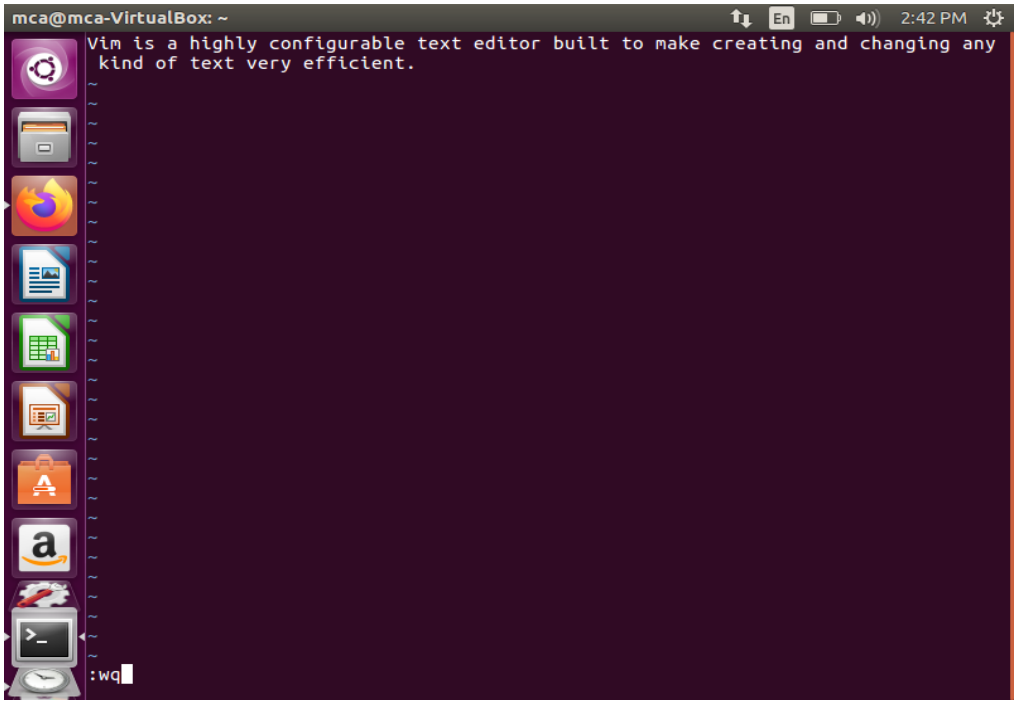
➢ Type i to switch into insert mode so that you can start editing the file. Enter or modify the text with your file.

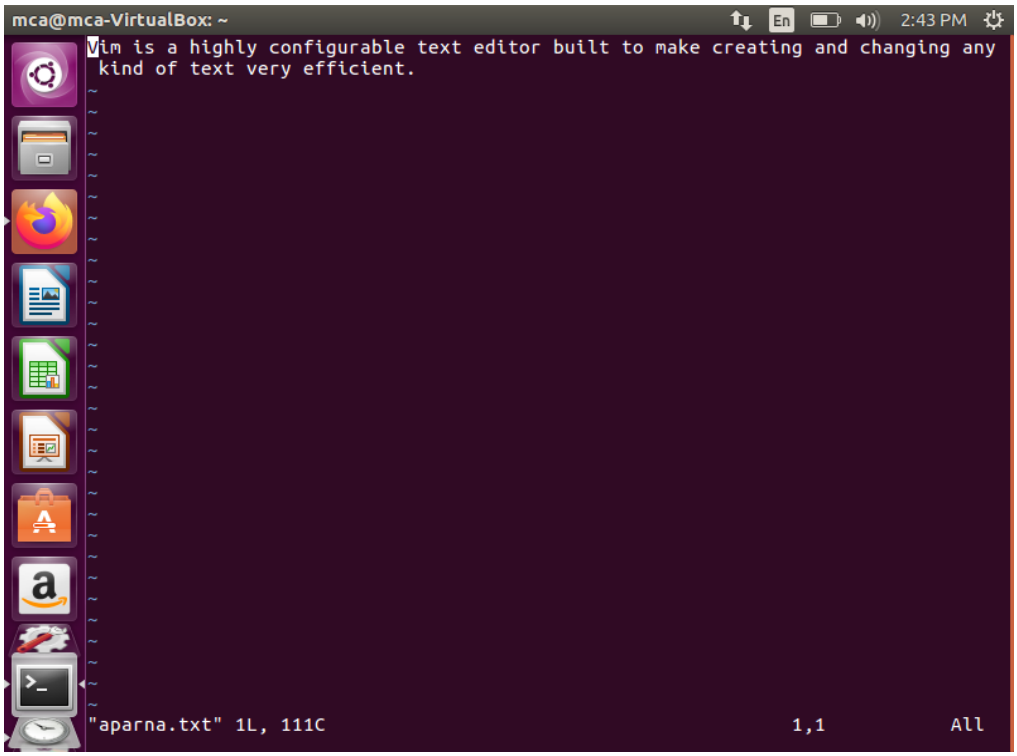


➢ Once you're done, press the escape key Esc to get out of insert mode and back to command mode.



➢ Type :wq to save and exit your file.





BASIC LINUX COMMANDS

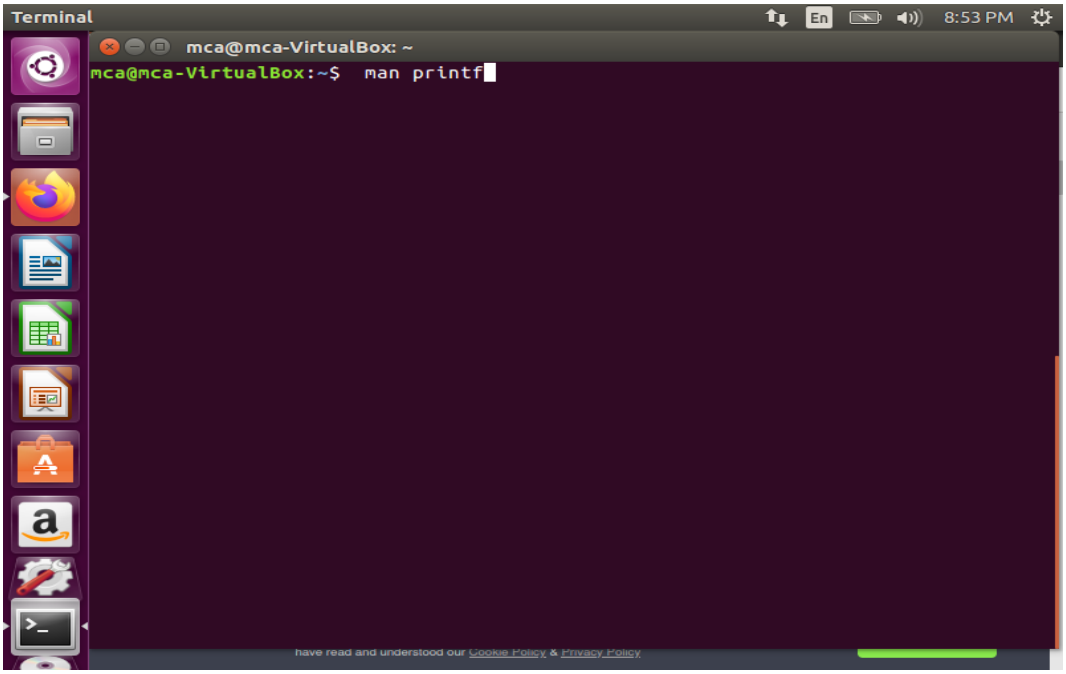
1. man

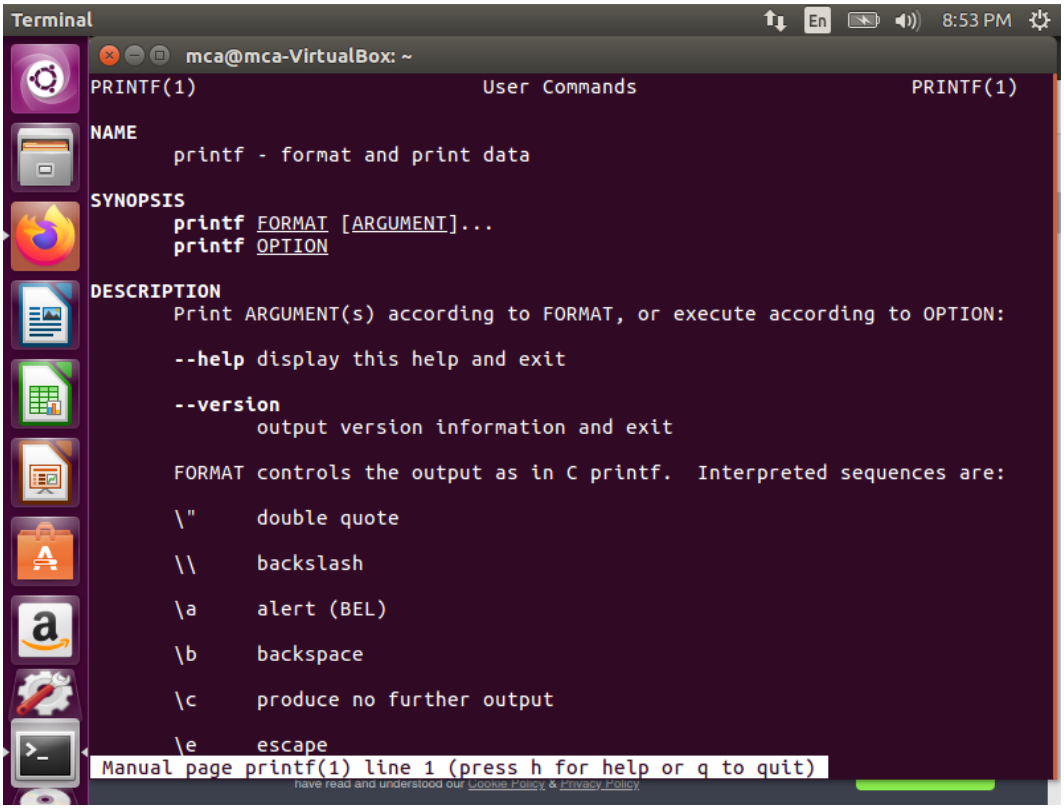
man command in Linux is used to display the user manual of any command that we can run on the terminal. It provides a detailed view of the command which includes *NAME, SYNOPSIS, DESCRIPTION, OPTIONS, EXIT STATUS, RETURN VALUES, ERRORS, FILES, VERSIONS, EXAMPLES, AUTHORS* and *SEE ALSO.*

Syntax : *$ man [OPTION]... [COMMAND NAME]...*

- No Option: It displays the whole manual of the command.

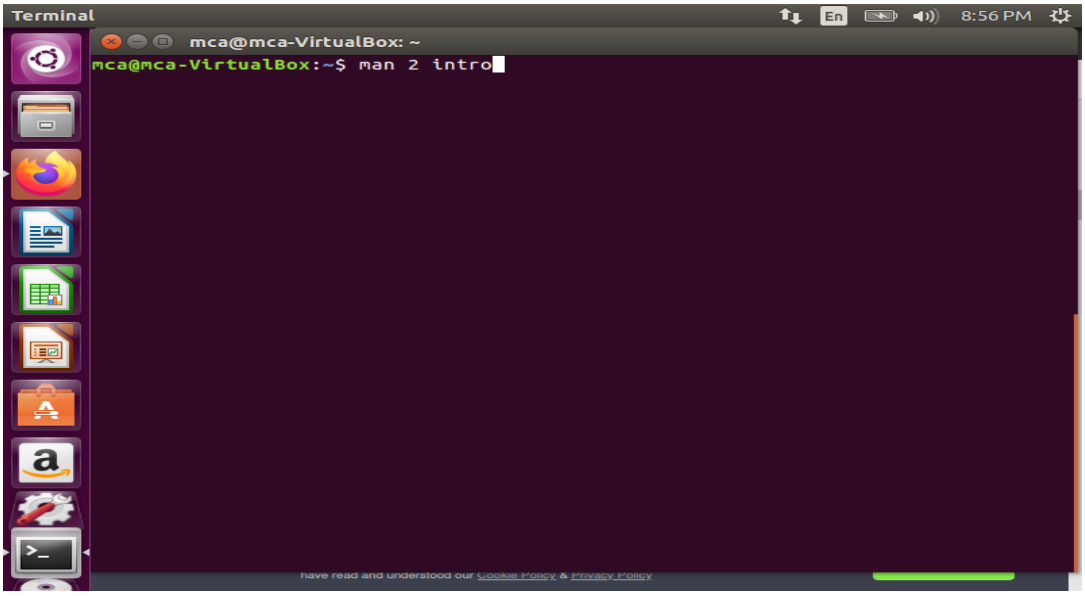
Syntax : *$ man [COMMAND NAME]*

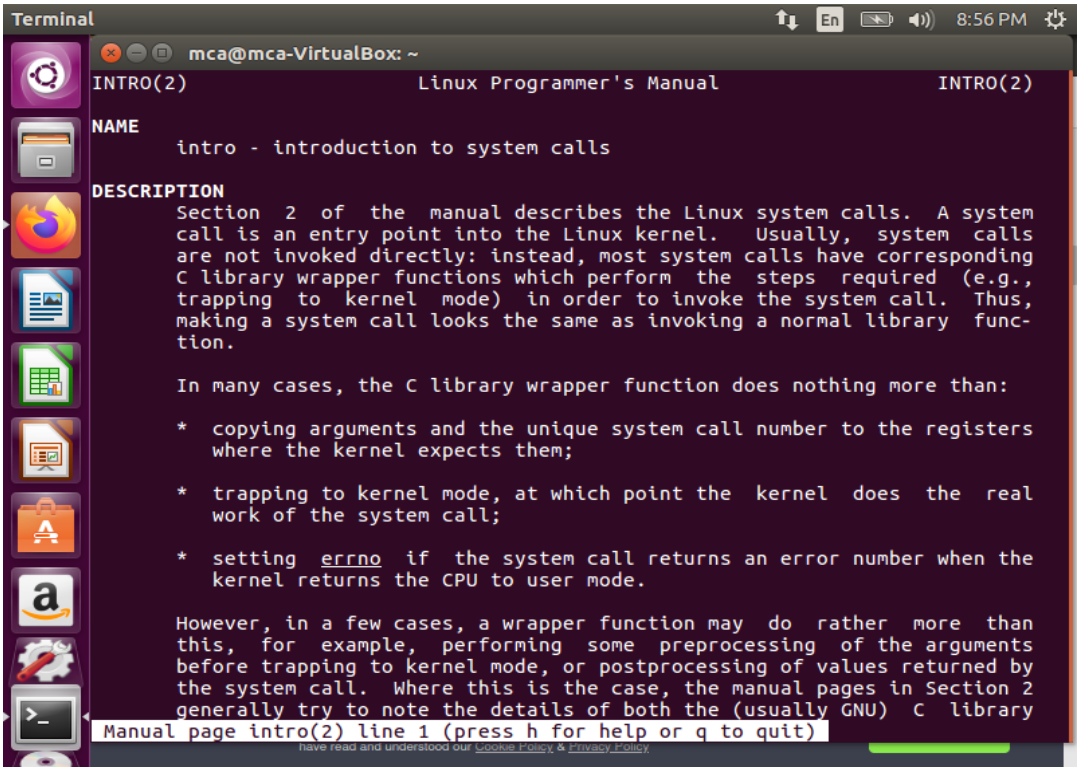




- Section-num: Since a manual is divided into multiple sections so this option is used to display only a specific section of a manual.

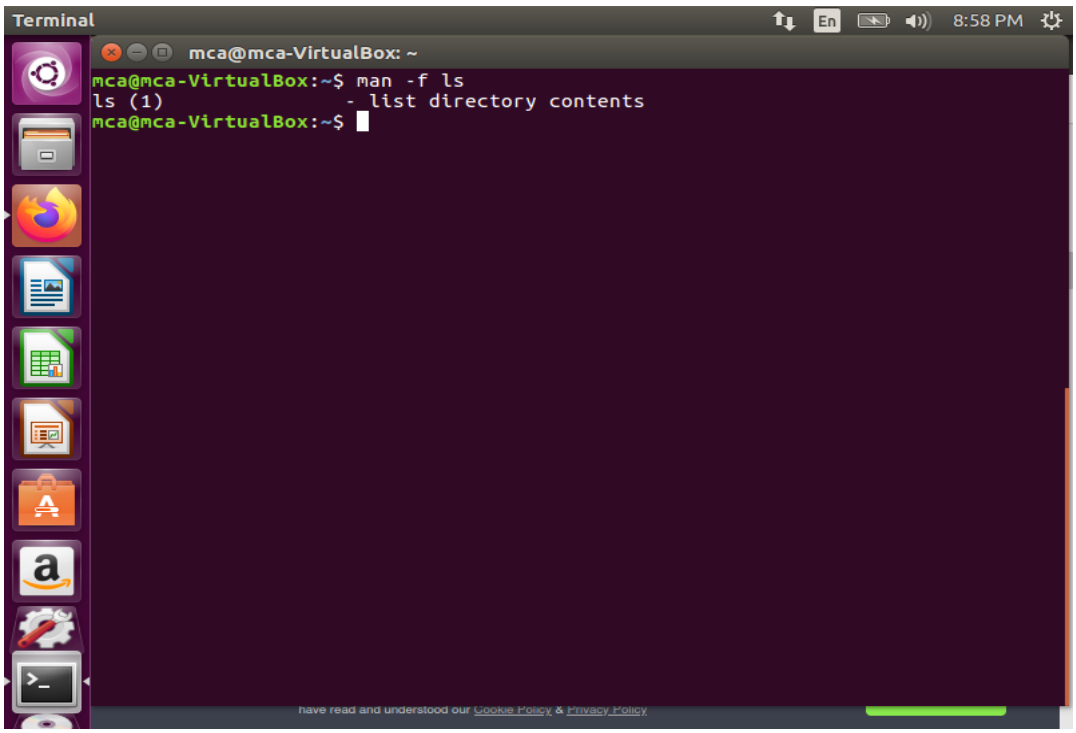
Syntax : *$ man [SECTION-NUM] [COMMAND NAME]*





- -f option: One may not be able to remember the sections in which a command is present. So this option gives the section in which the given command is present.

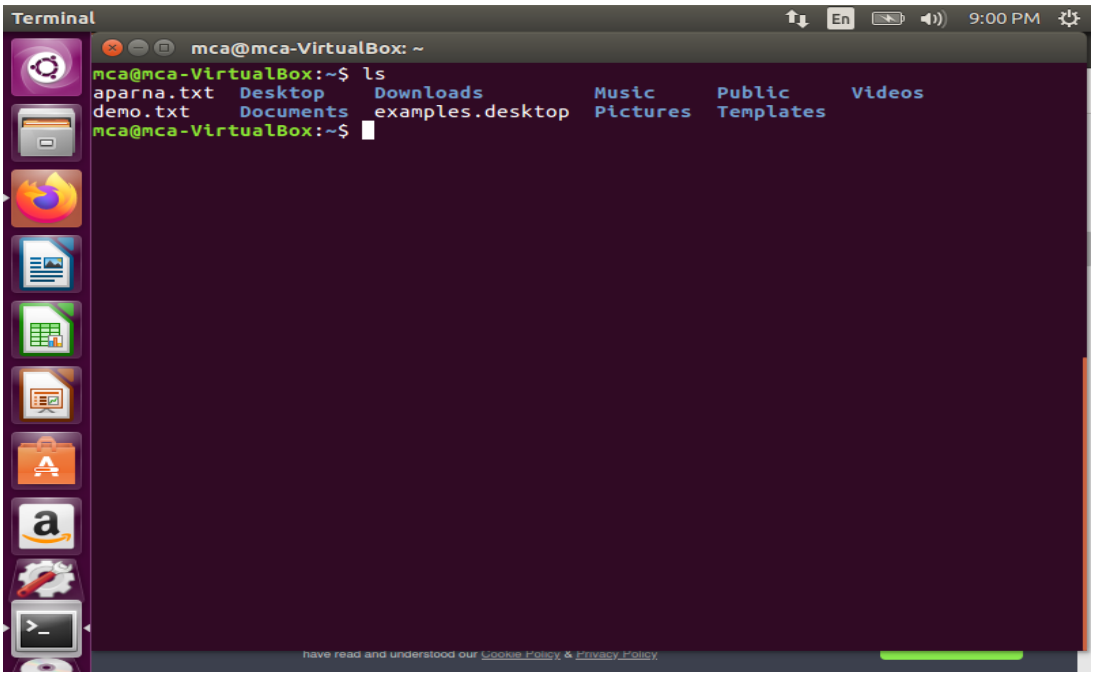
Syntax: *$ man -f [COMMAND NAME]*



2. ls

ls is a Linux shell command that lists directory contents of files and directories.

Syntax: ***$ ls***



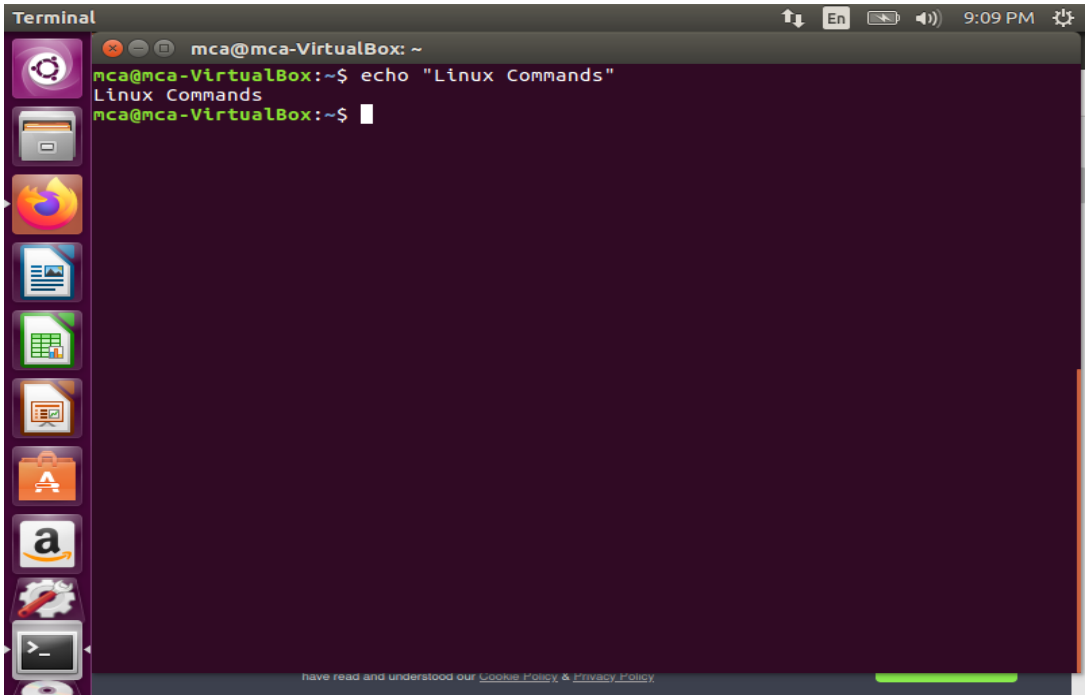
3. echo

echo command in linux is used to display line of text/string that are passed as an argument . This is a built in command that is mostly used in shell scripts and batch files to output status text to the screen or a file.

Syntax : *echo [option] [string]*

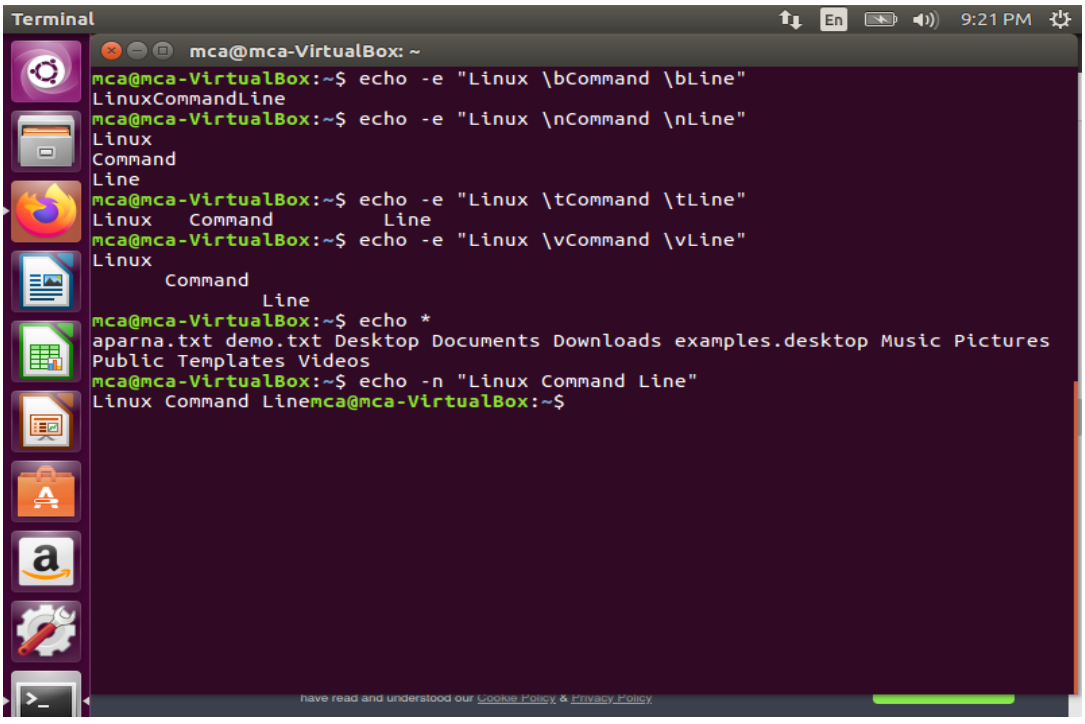
- For displaying a text/string :

Syntax : *echo [string]*



Options of echo command

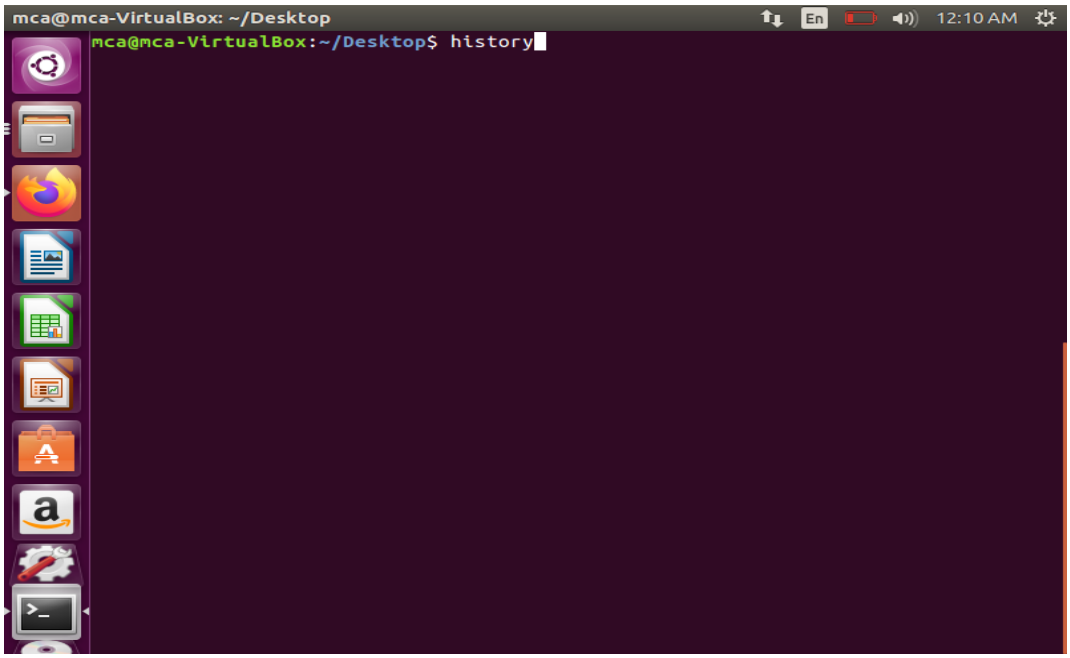
- \b : it removes all the spaces in between the text
- \n : this option creates new line from where it is used.
- \t : this option is used to create horizontal tab spaces.
- \v : this option is used to create vertical tab spaces.
- echo * : this command will print all files/folders, similar to ls command .
- -n : this option is used to omit echoing trailing newline .

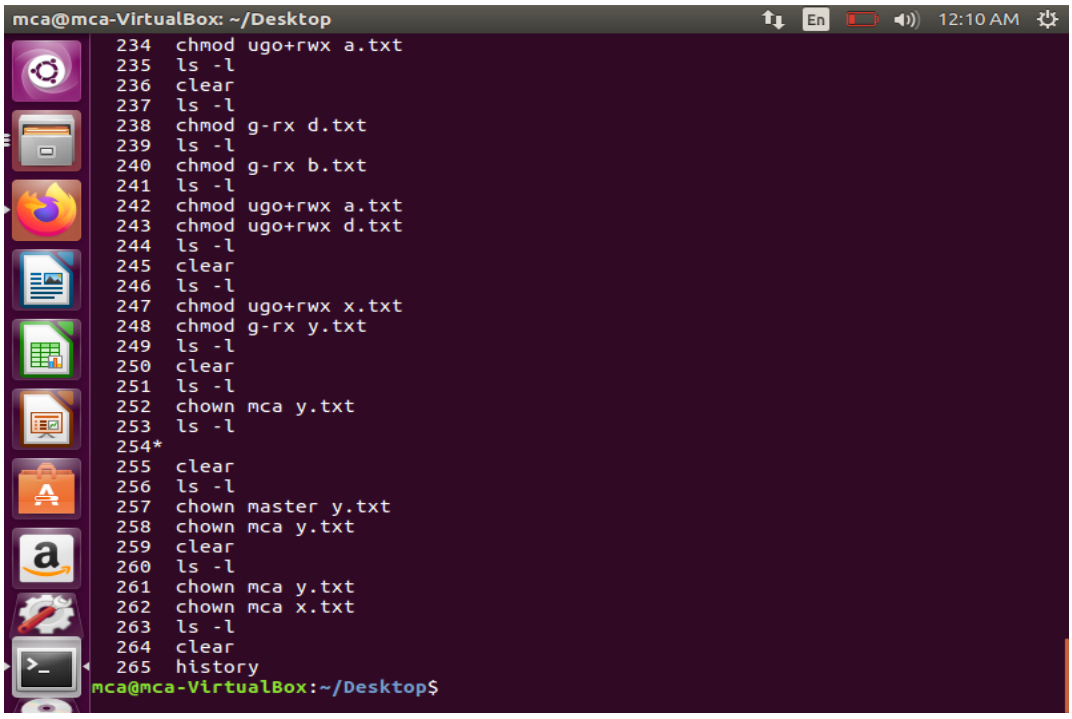


4. history

history command is used to view the previously executed command. This feature was not available in the Bourne shell. Bash and Korn support this feature in which every command executed is treated as the event and is associated with an event number using which they can be recalled and changed if required. These commands are saved in a history file. In Bash shell history command shows the whole list of the command**.**

Syntax: *$ history*





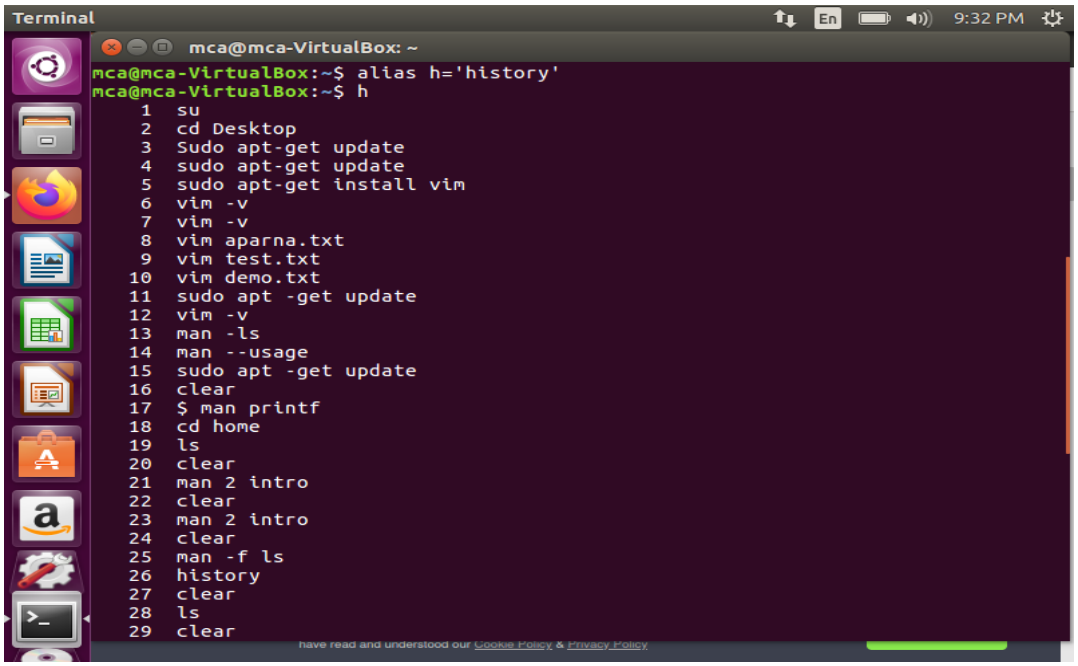
5. alias

alias command instructs the shell to replace one string with another string while executing the commands**.**

Syntax: *alias [-p] [name[=value] ...]*

For creating an alias :

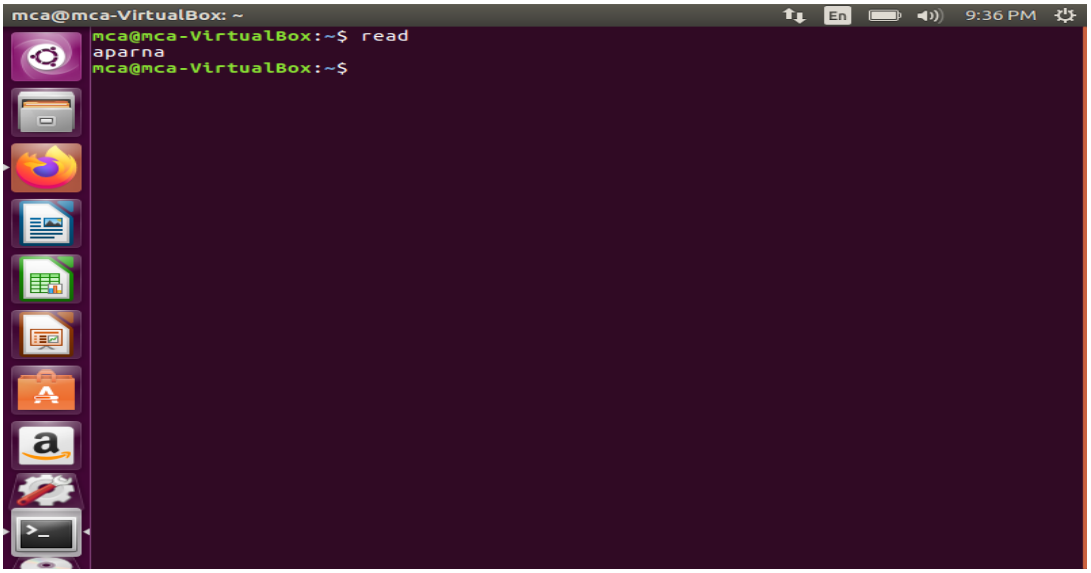
Syntax: *alias name="value"*

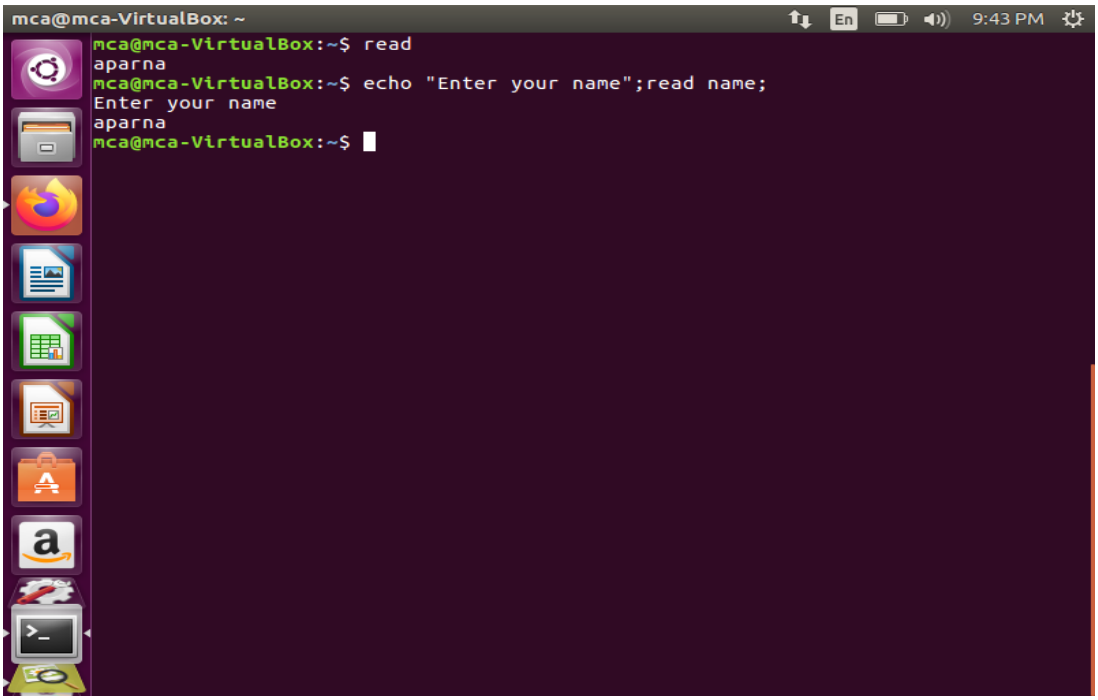


6. read

read command in Linux system is used to read from a file descriptor. Basically, this command read up the total number of bytes from the specified file descriptor into the buffer. If the number or count is zero then this command may detect the errors. But on success, it returns the number of bytes read. Zero indicates the end of the file. If some errors found then it returns -1.

Syntax: *read*





7. more

more command is used to view the text files in the command prompt, displaying one screen at a time in case the file is large. The more command also allows the user do scroll up and down through the page. When the output is large, we can use more command to see output one by one.

Syntax: *more [-options] [-num] [+/pattern] [+linenum] [file_name]*

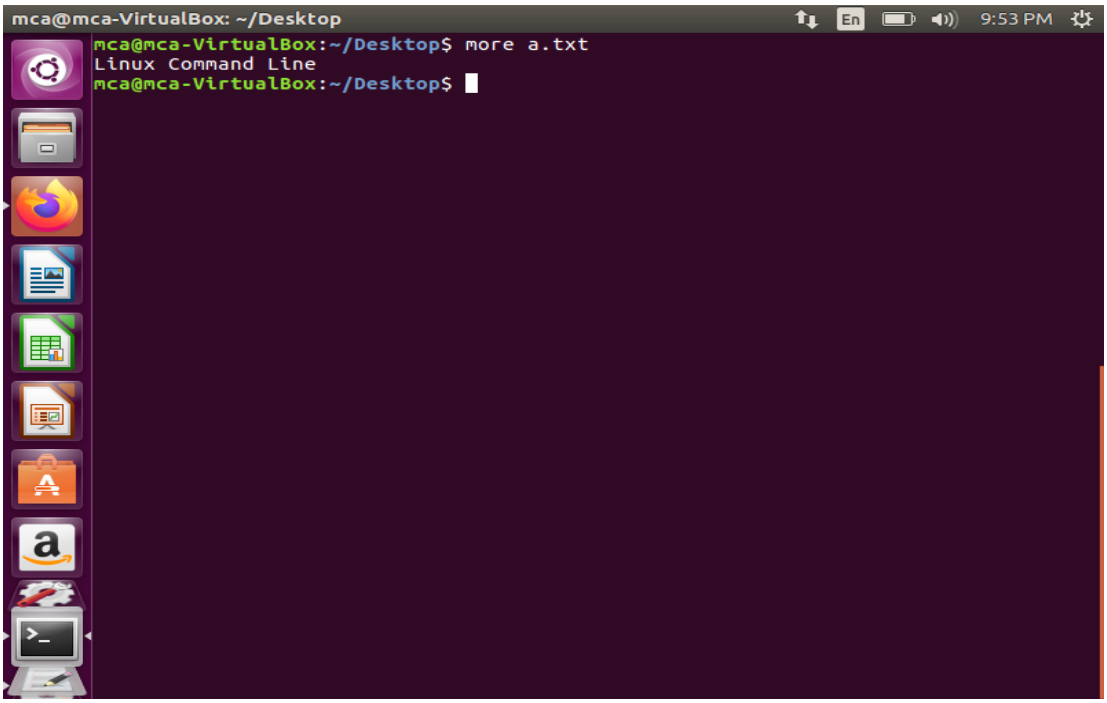
- *[-options]: any option that you want to use in order to change the way the file is displayed. Choose any one from the followings: (-d,-f, -p, -u)*
- *[-num]: type the number of lines that you want to display per screen.*
- *[+/pattern]: replace the pattern with any string that you want to find in the text file.*
- *[+linenum]: use the line number from where you want to start displaying the text content.*
- *[file_name]: name of the file containing the text that you want to display on the screen.*

Options:

-d : Use this command in order to help the user to navigate.

-f : This option does not wrap the long lines and displays them as such.

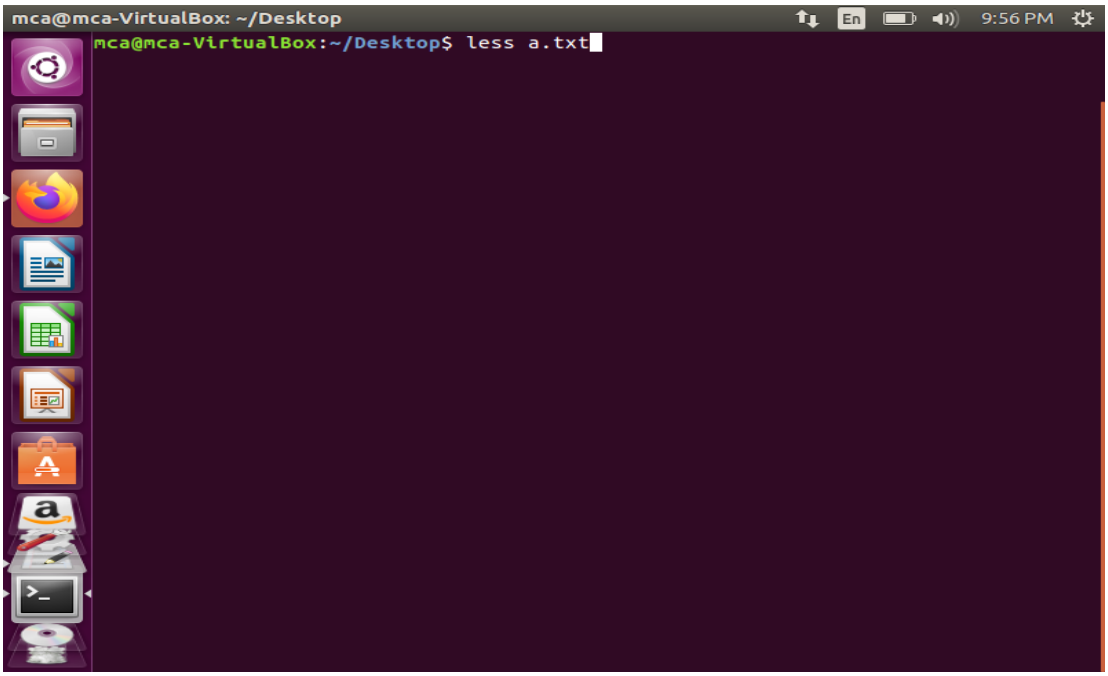
-p : This option clears the screen and then displays the text

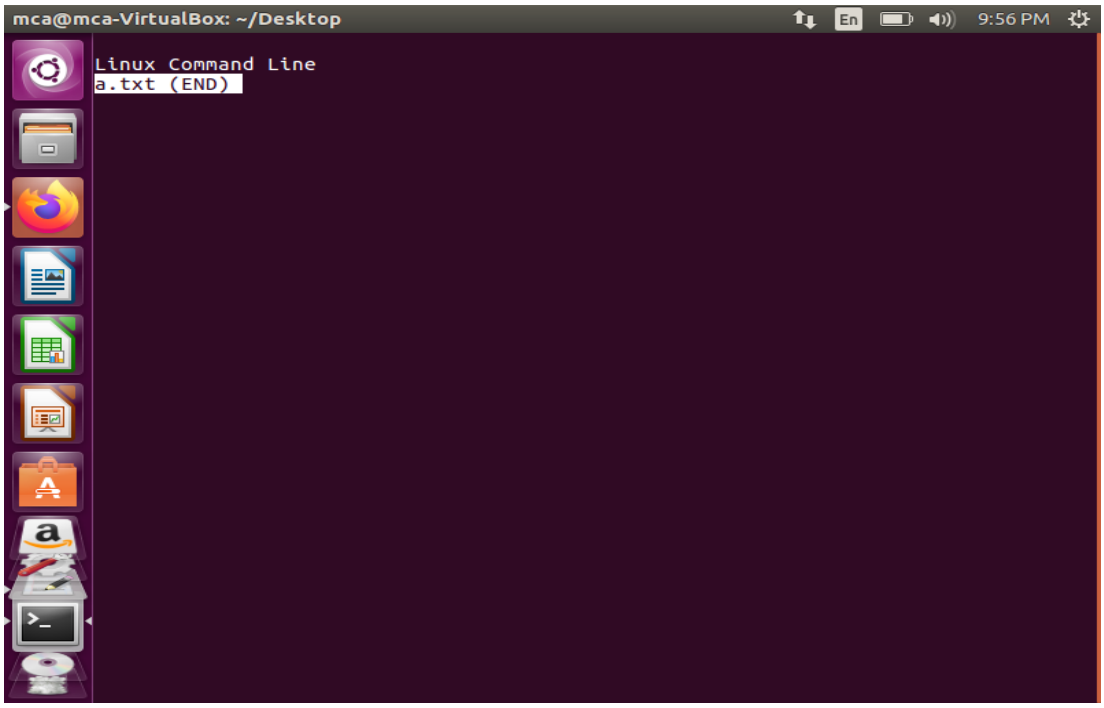


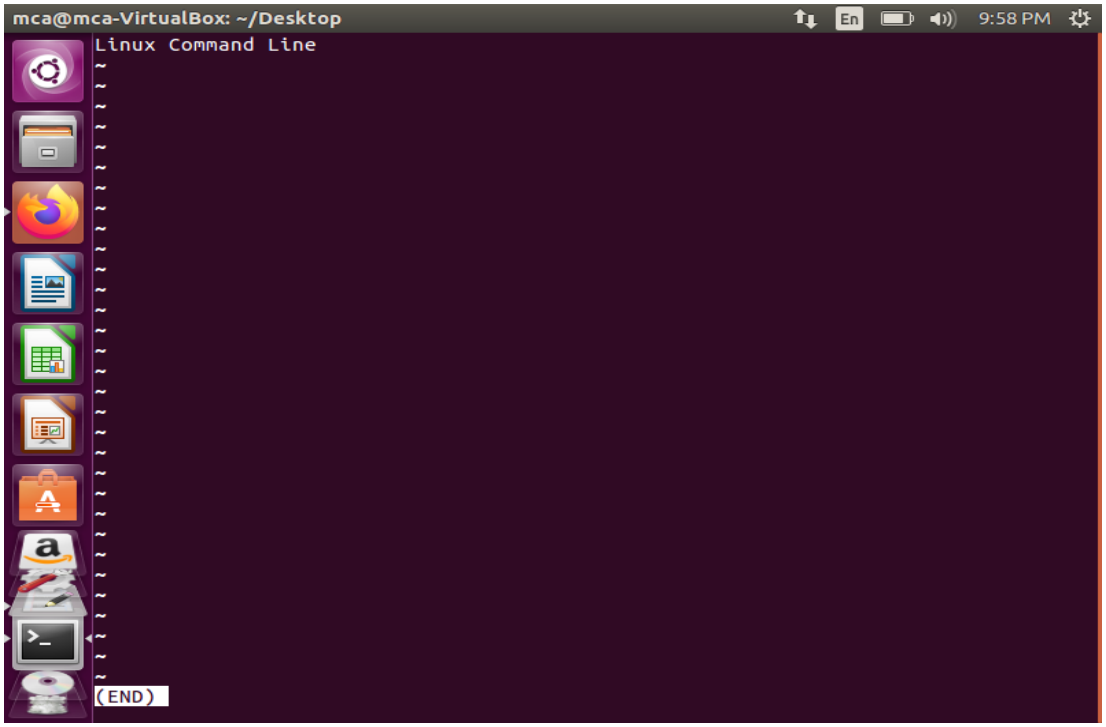
8. less

Less command is a Linux utility that can be used to read the contents of a text file one page(one screen) at a time. It has faster access because if file is large it doesn't access the complete file, but accesses it page by page.

Syntax : *less filename*







9. cat

Cat(concatenate) command is very frequently used in Linux. It reads data from the file and gives their content as output. It helps us to create, view, concatenate files. So let us see some frequently used cat commands.

- To view a single file

Command: *$ cat filename*

- To view multiple files

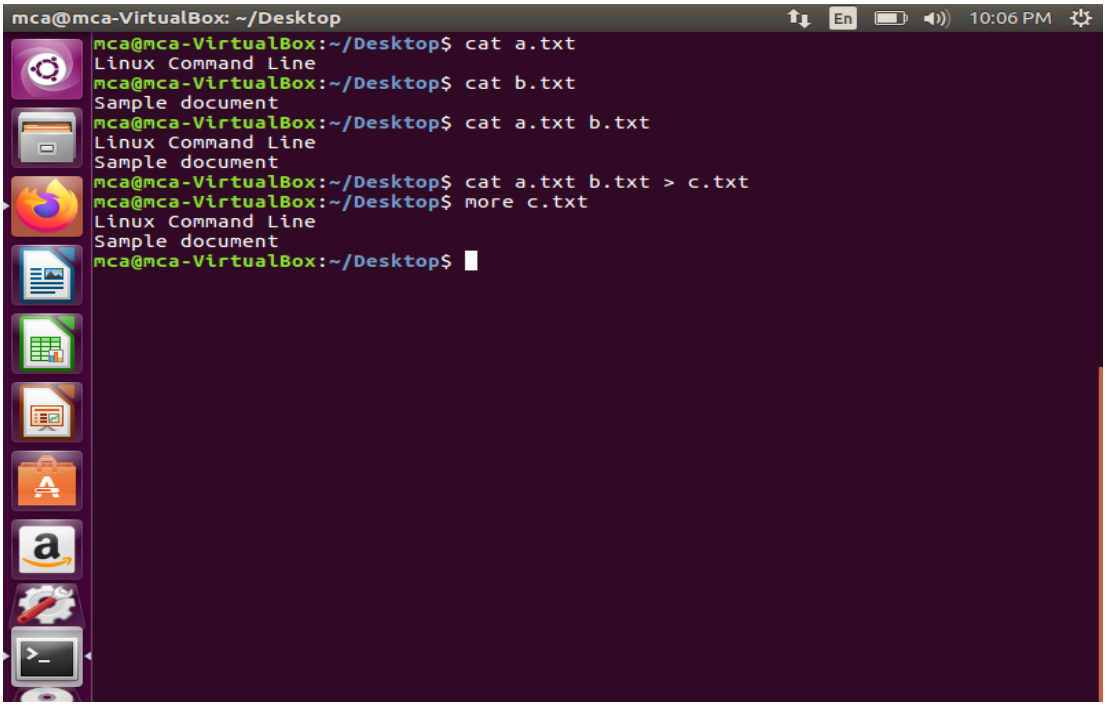
Command: *$cat file1 file2*

- Create a file

Command: *$ cat >newfile*

- Copy the contents of one file to another file.

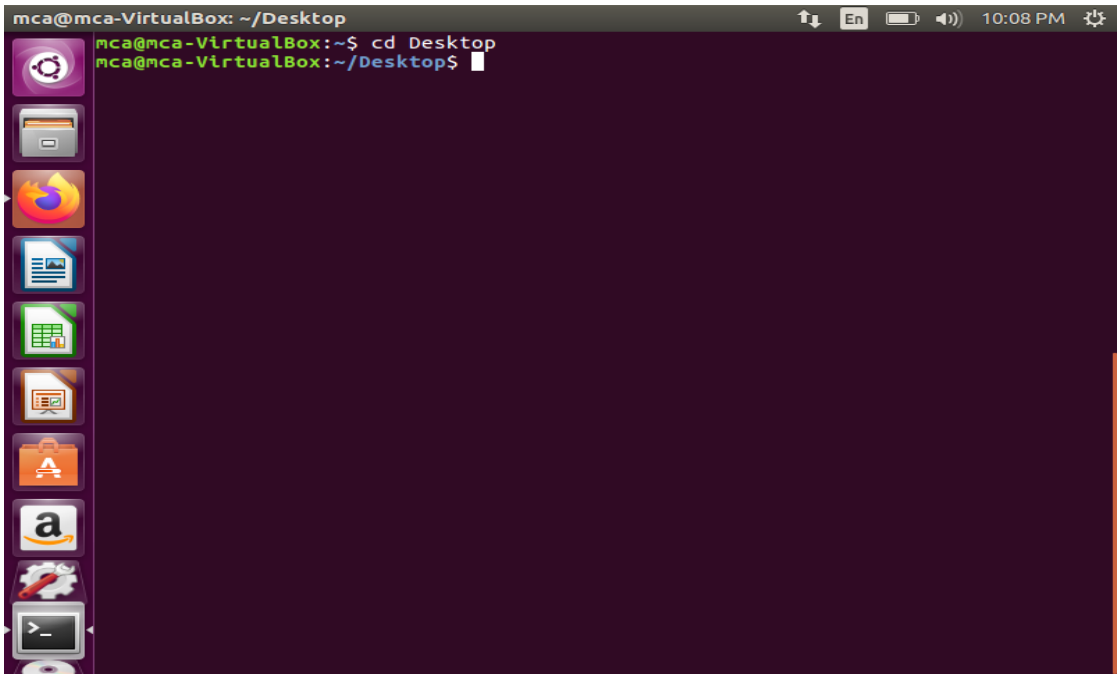
Command: *$cat [filename-whose-contents-is-to-be-copied] > [destination-filename]*



10. cd

cd command in linux known as change directory command. It is used to change current working directory.

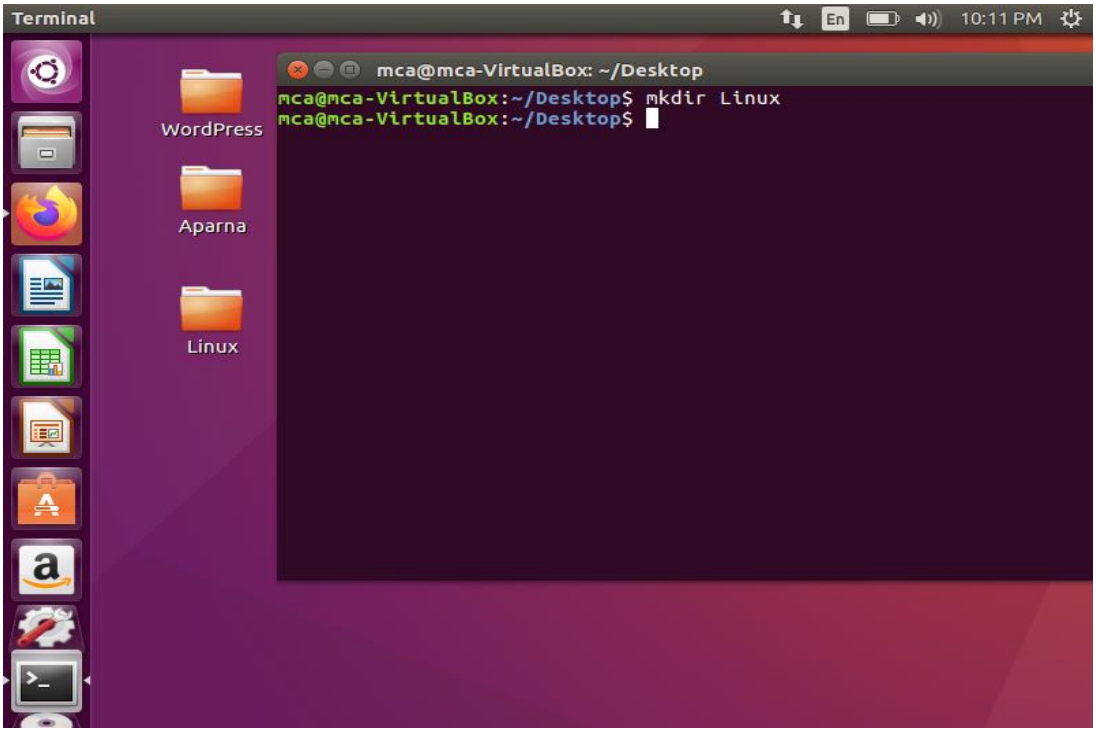
Syntax: *$ cd [directory]*

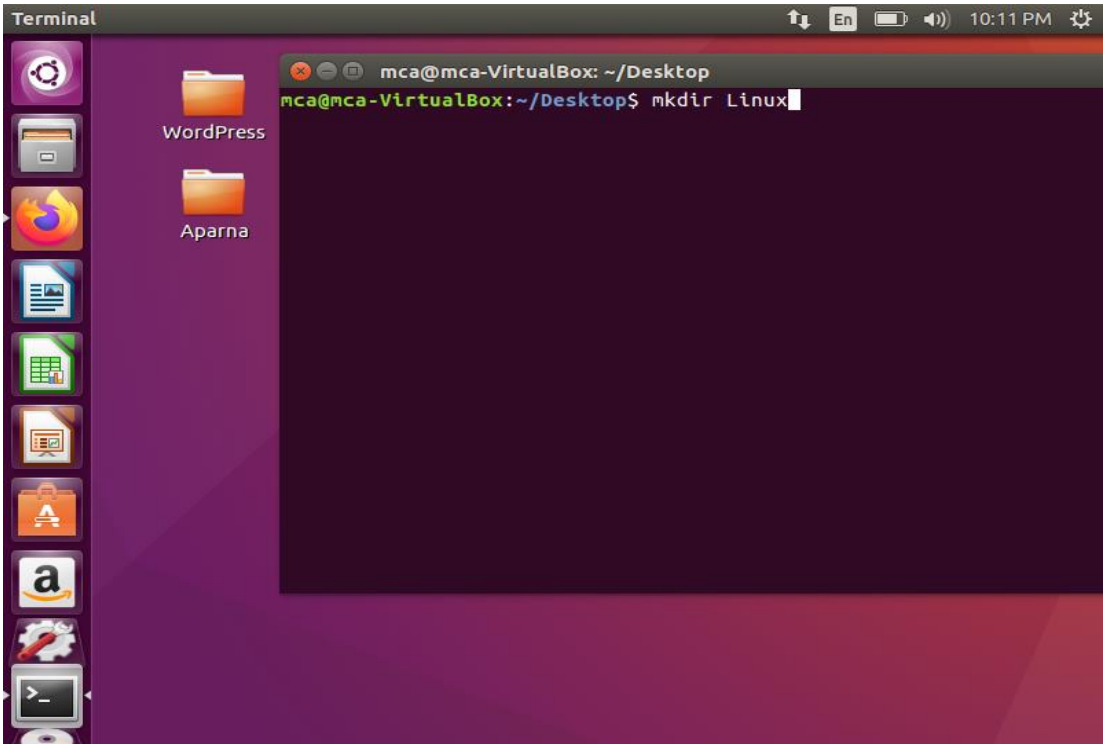


11. mkdir

mkdir command in Linux allows the user to create directories (also referred to as folders in some operating systems).

Syntax: *mkdir [options...] [directories ...]*

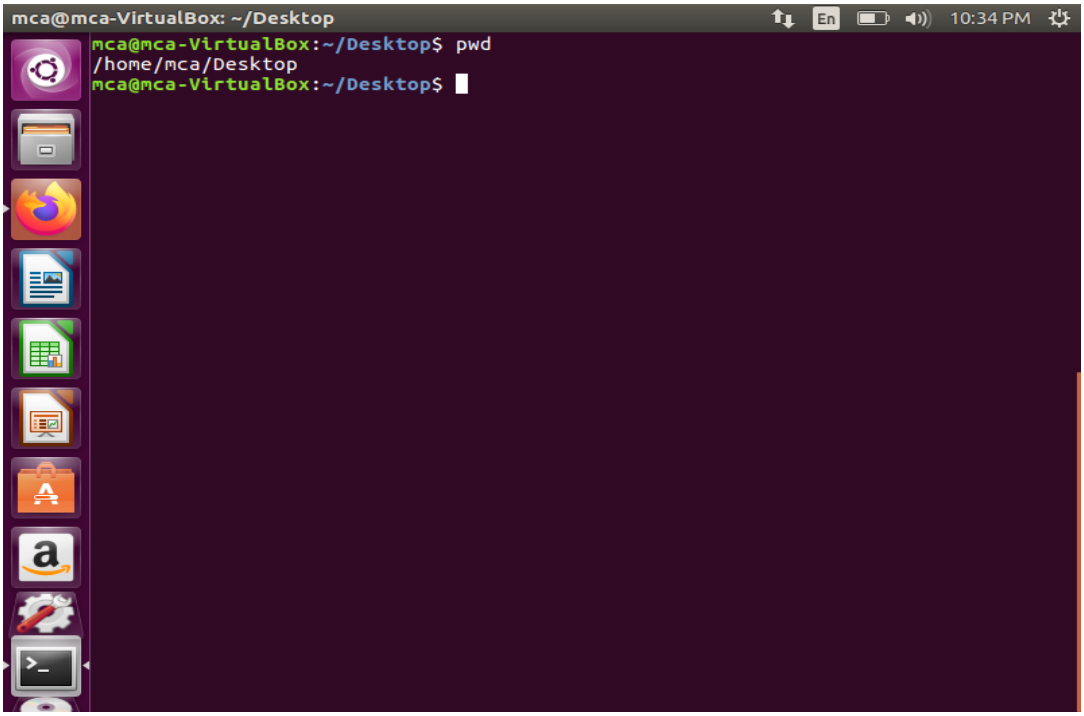




12. pwd

pwd stands for Print Working Directory. It prints the path of the working directory, starting from the root. pwd is shell built-in command(pwd) or an actual binary(/bin/pwd). $PWD is an environment variable which stores the path of the current directory.

Syntax: *pwd*

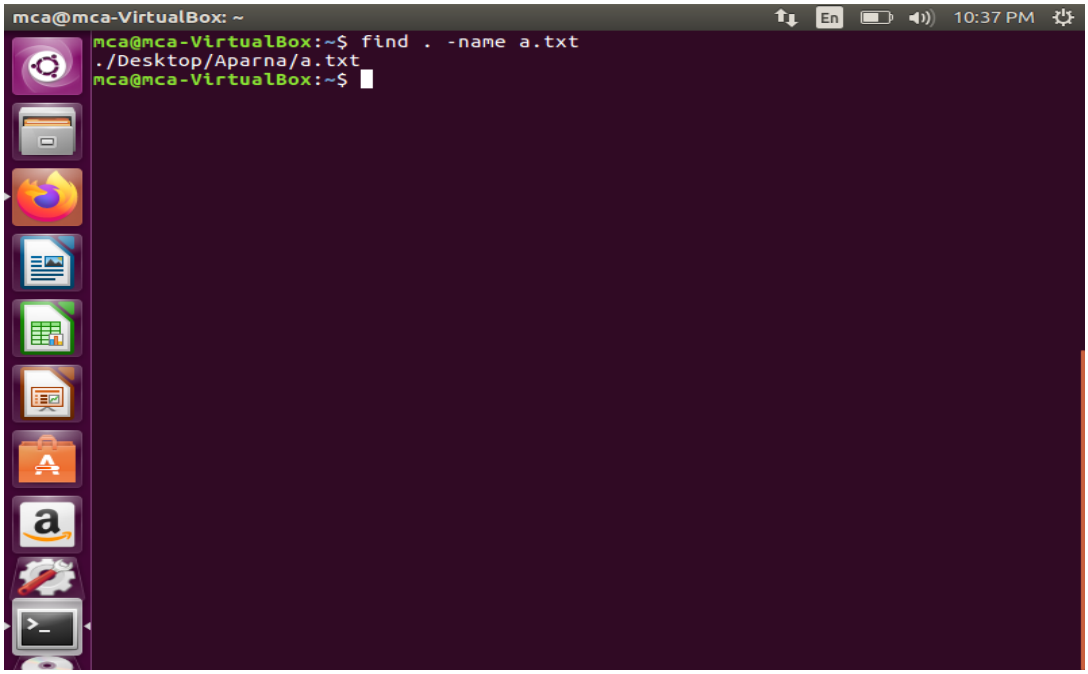


13. find

The find command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them. It supports searching by file, folder, name, creation date, modification date, owner and permissions.

Syntax : *$ find [where to start searching from]*

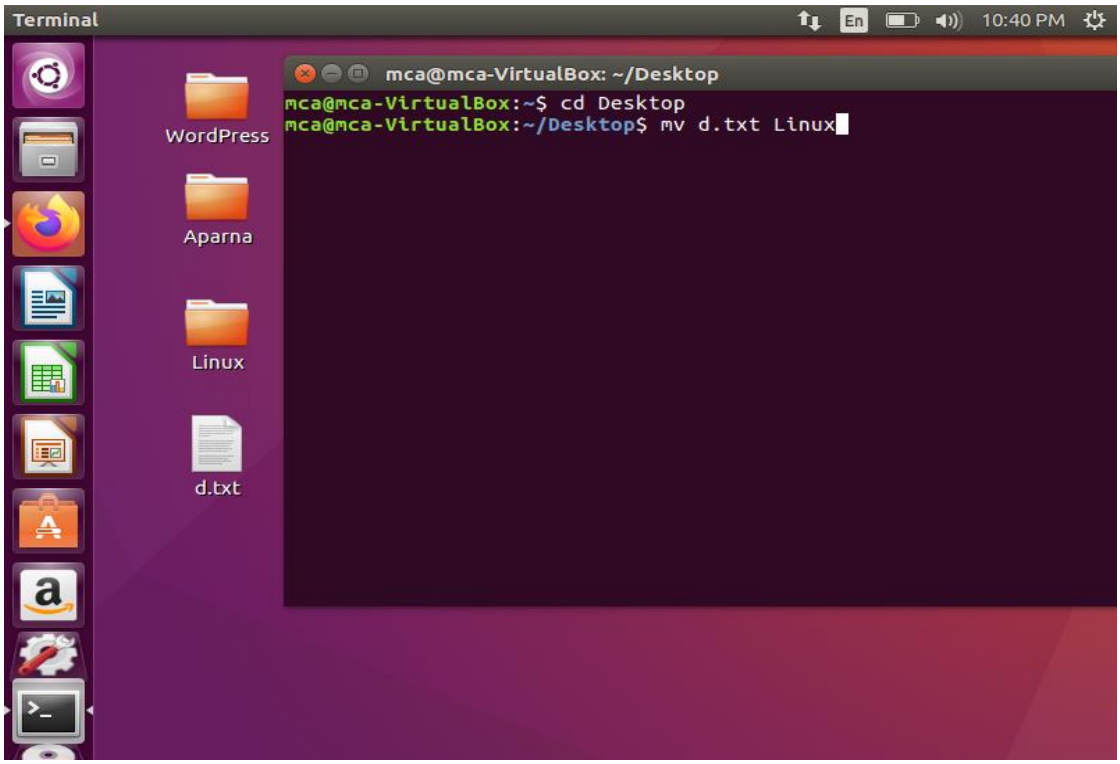
[expression determines what to find] [-options] [what to find]

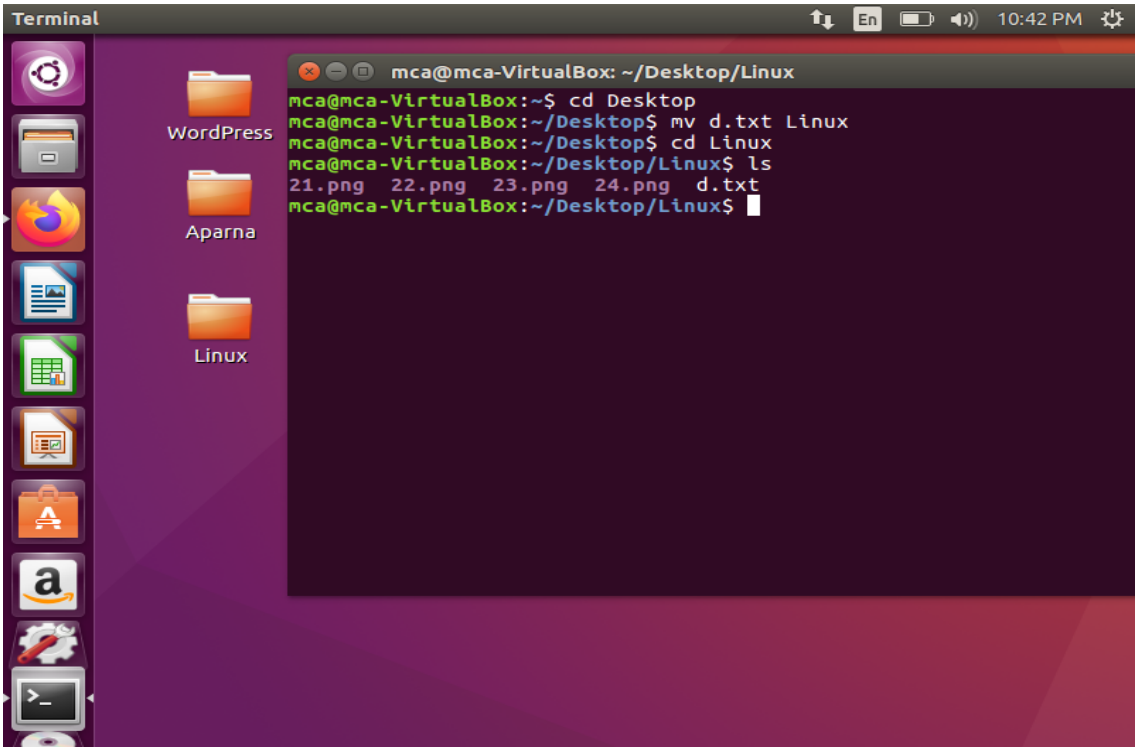


14. mv

mv stands for move. mv is used to move one or more files or directories from one place to another in a file system like UNIX.

Syntax: *mv [Option] source destination*

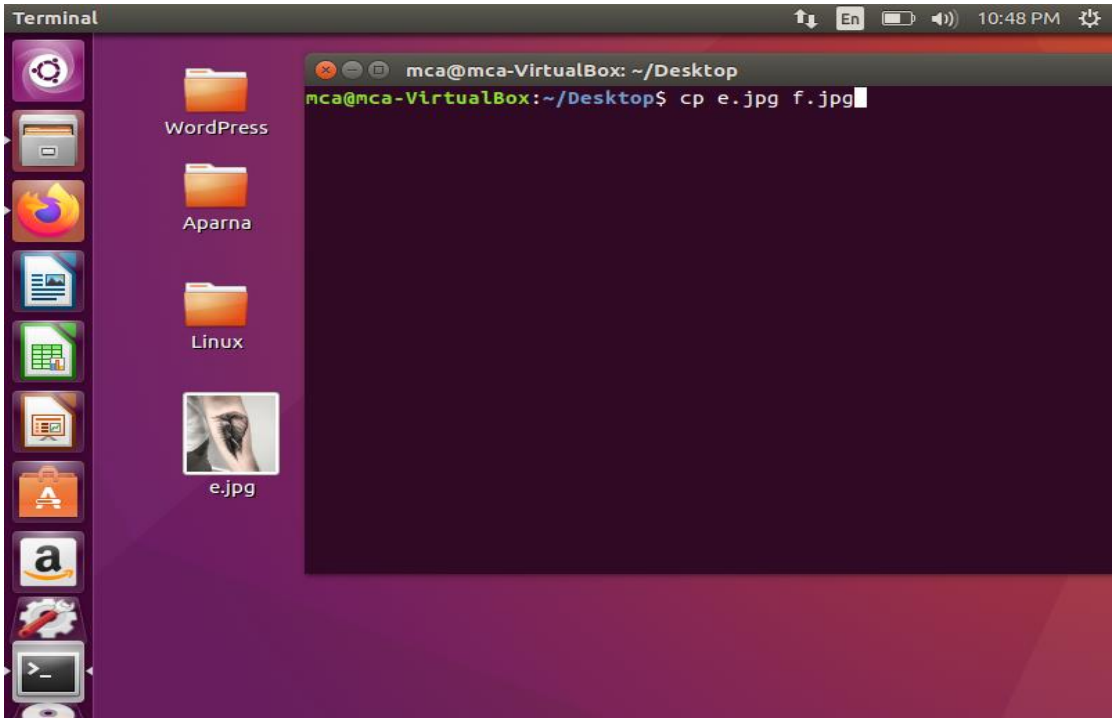


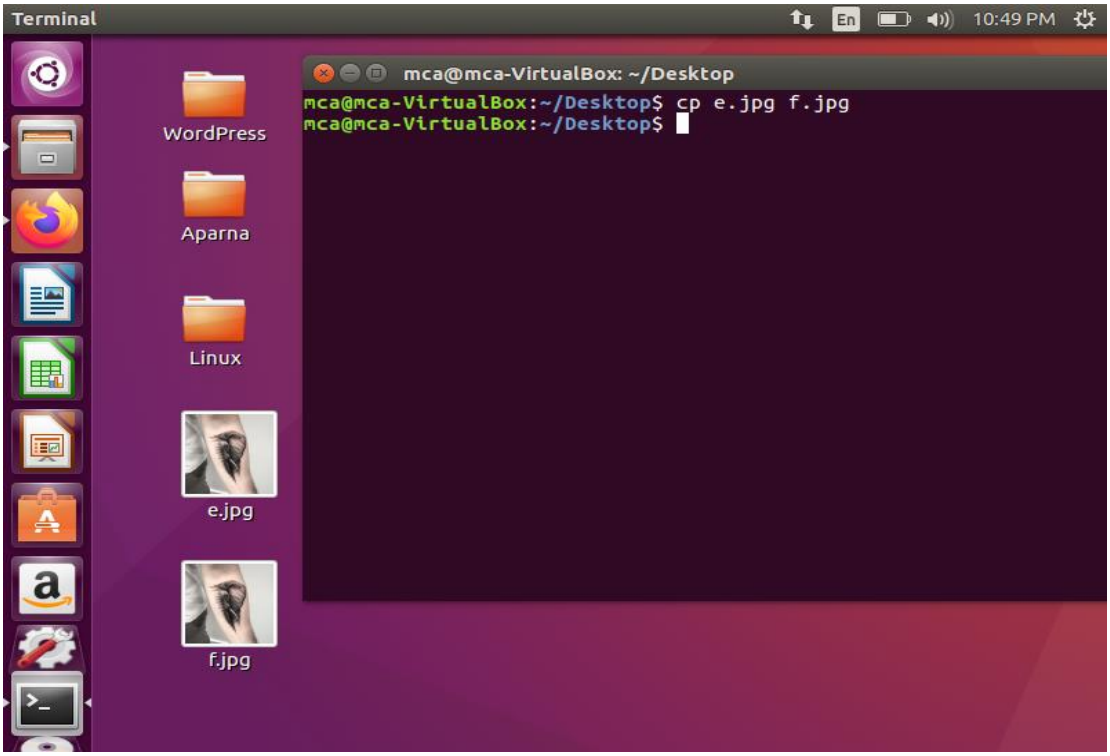


15. cp

cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. *cp* command require at least two filenames in its arguments.

Syntax: *cp [OPTION] Source Destination*

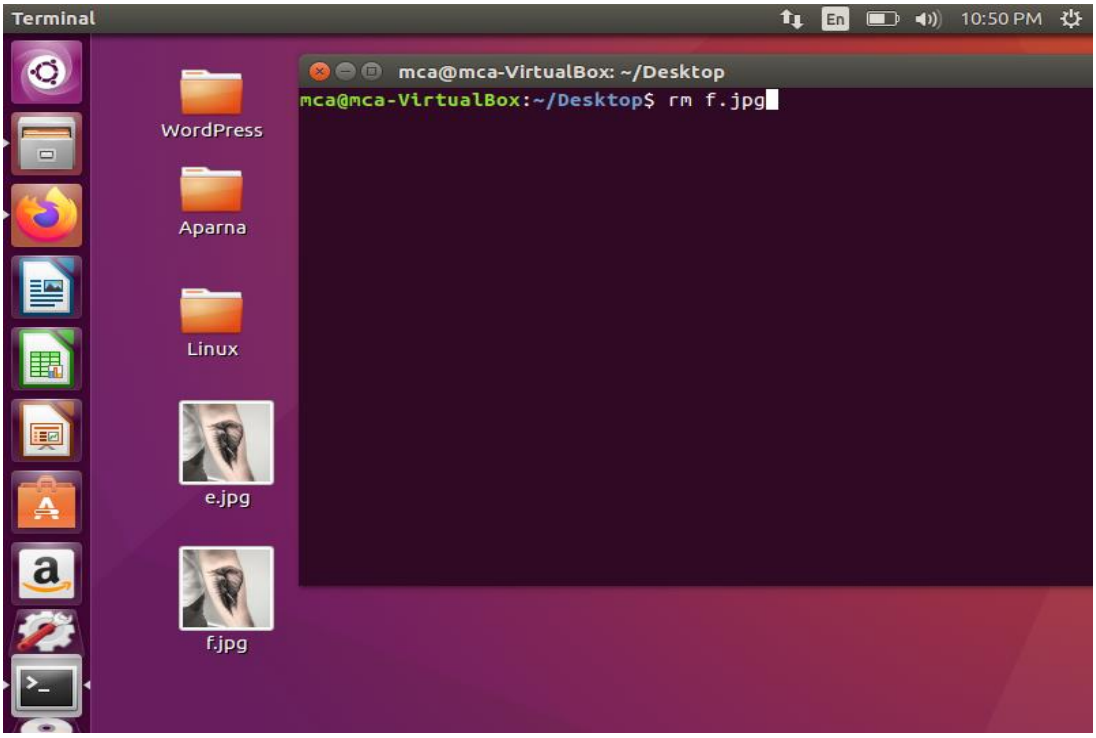


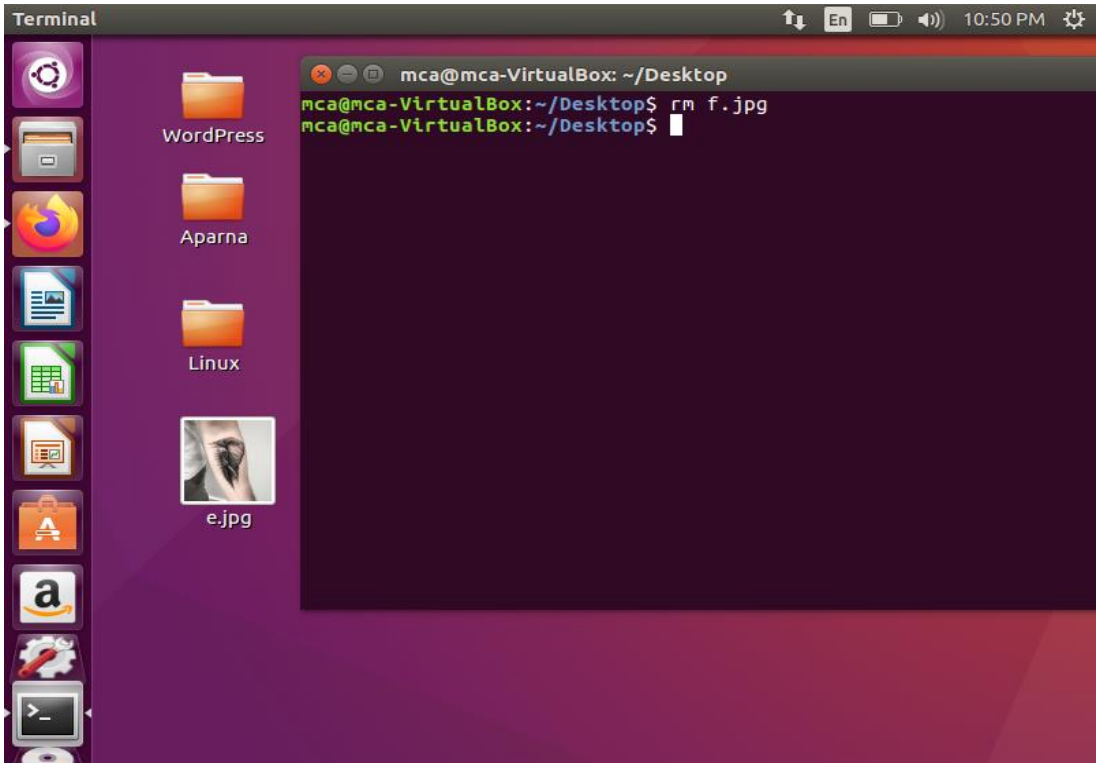


16. rm

rm stands for remove here. rm command is used to remove objects such as files, directories, symbolic links and so on from the file system like UNIX.

Syntax: *rm [OPTION]... FILE...*

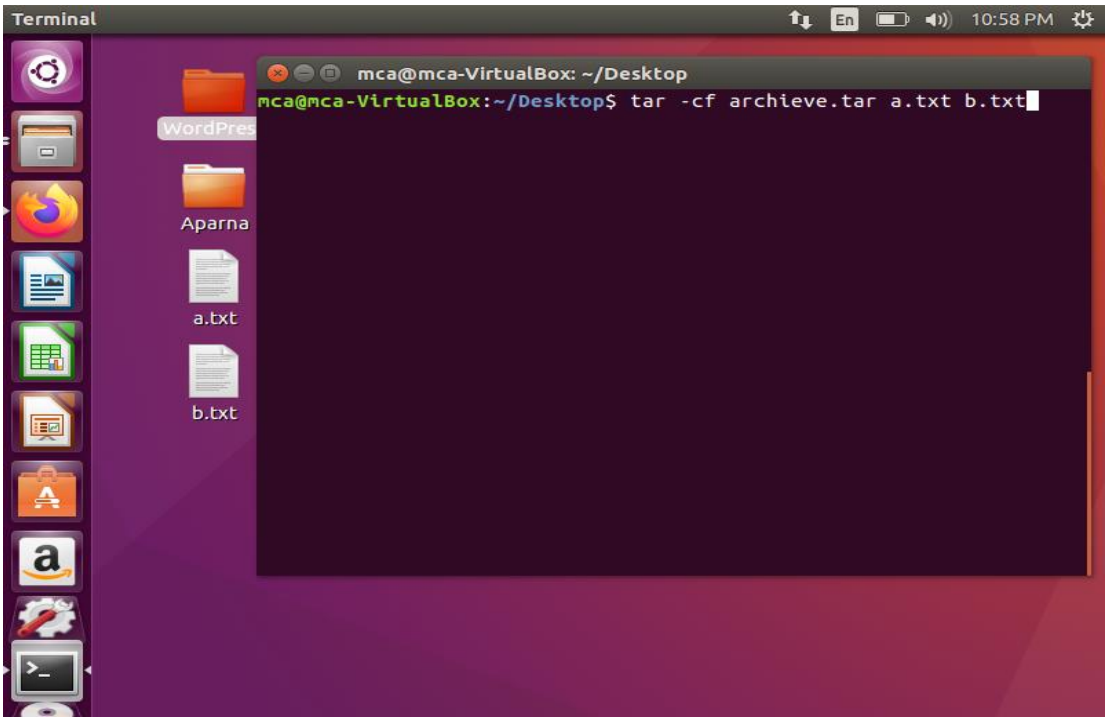


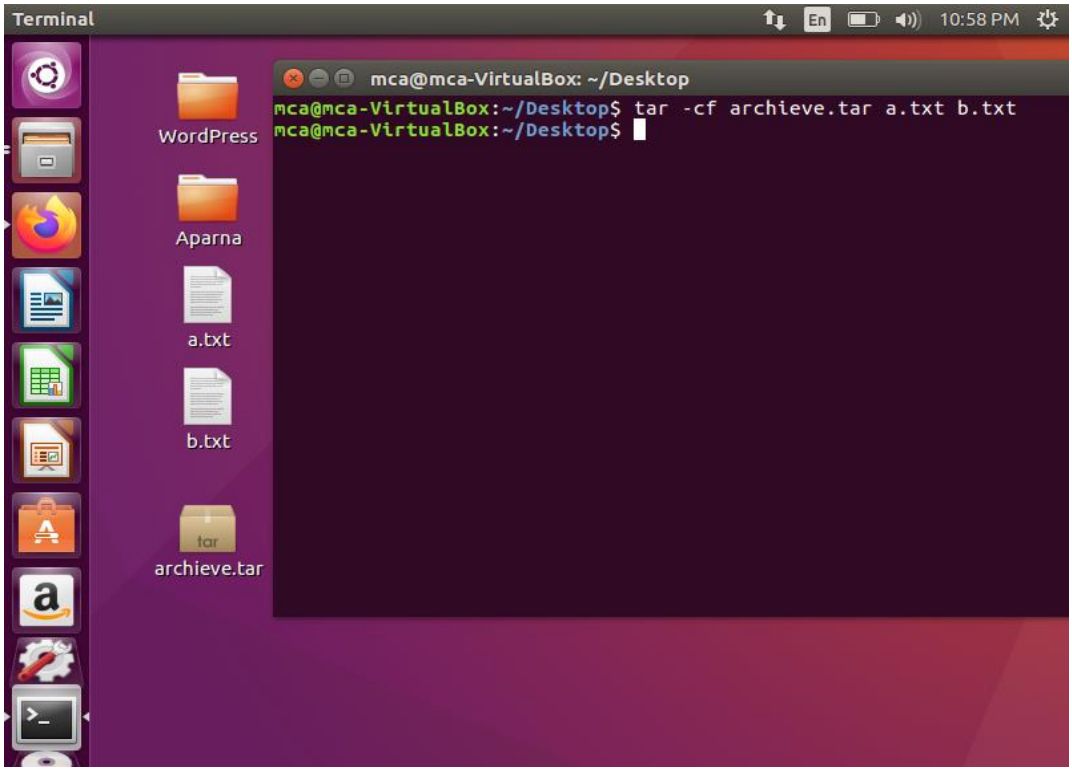


17. tar

The Linux ‘tar’ stands for tape archive, is used to create Archive and extract the Archive files. tar command in Linux is one of the important command which provides archiving functionality in Linux. We can use Linux tar command to create compressed or uncompressed Archive files and also maintain and modify them.

Syntax: *tar [options] [archive-file] [file or directory to be archived]*

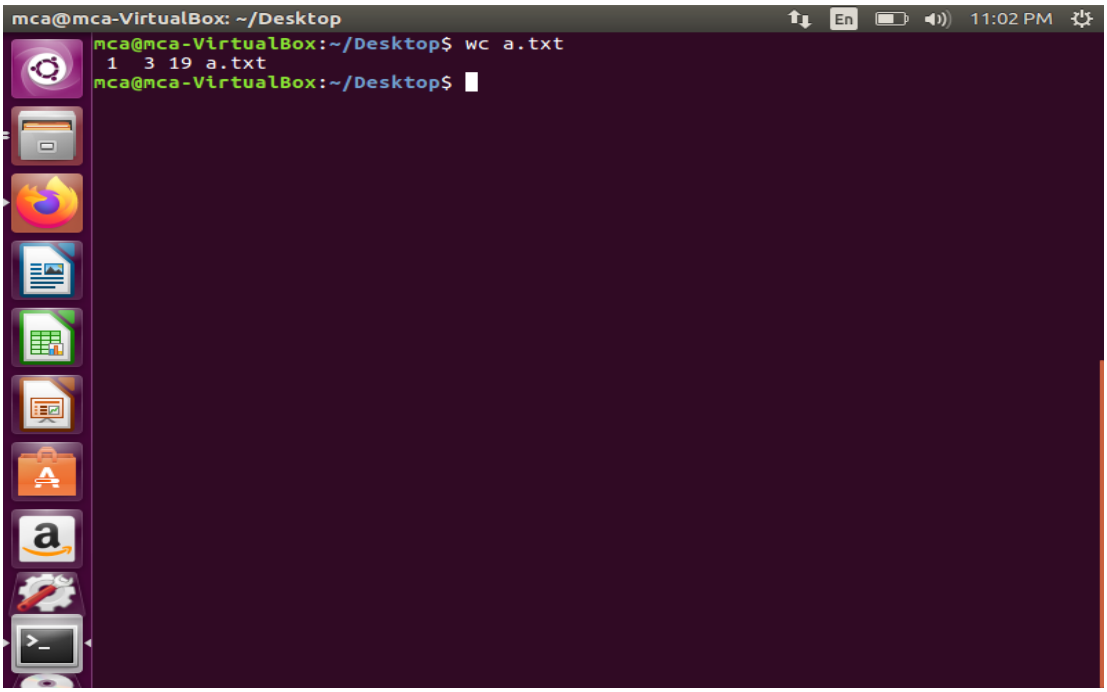




18. wc

wc stands for word count. As the name implies, it is mainly used for counting purpose.It is used to find out number of lines, word count, byte and characters count in the files specified in the file arguments.

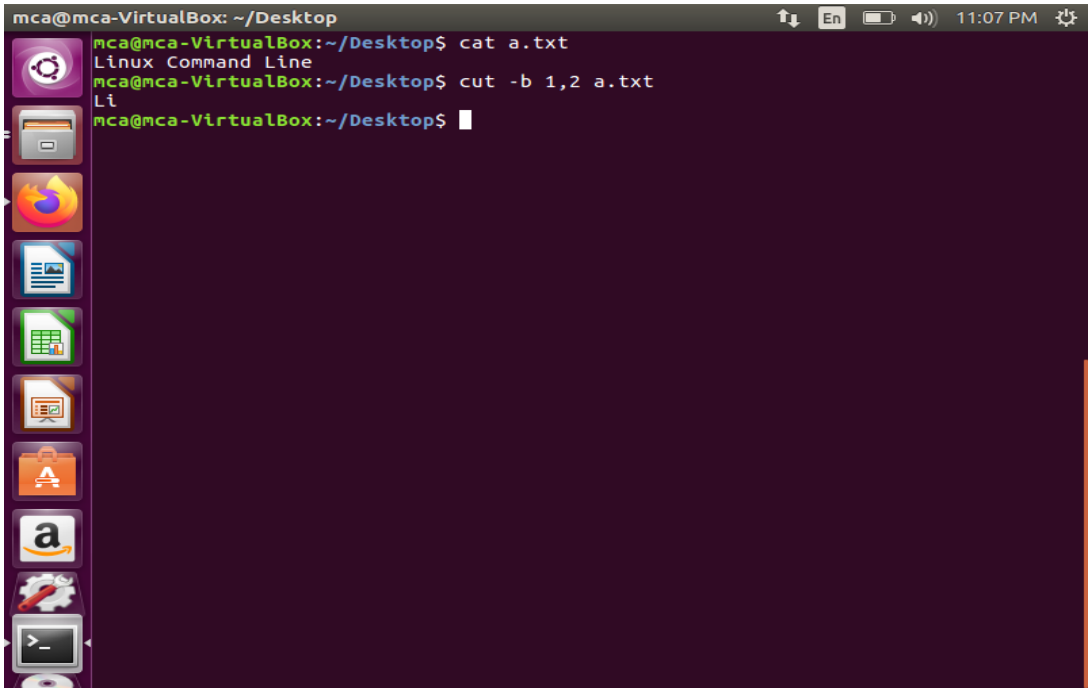
Syntax: *wc [OPTION]... [FILE]...*



19. cut

The cut command in UNIX is a command for cutting out the sections from each line of files and writing the result to standard output. It can be used to cut parts of a line by byte position, character and field.

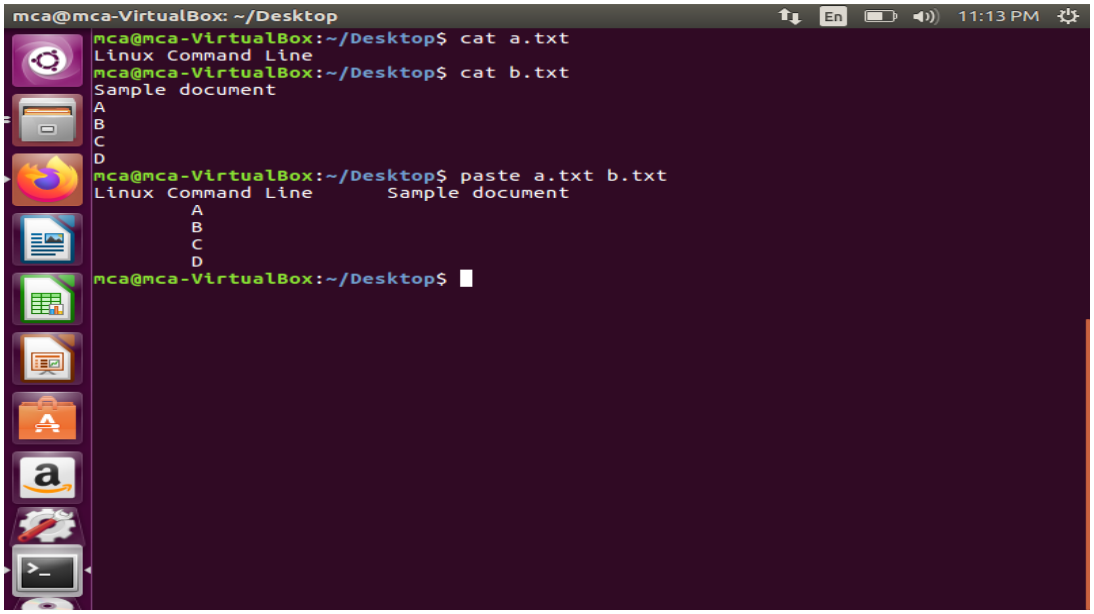
Syntax: *cut OPTION... [FILE]...*



20. paste

Paste command is one of the useful commands in Unix or Linux operating system. It is used to join files horizontally (parallel merging) by outputting lines consisting of lines from each file specified, separated by tab as delimiter, to the standard output.

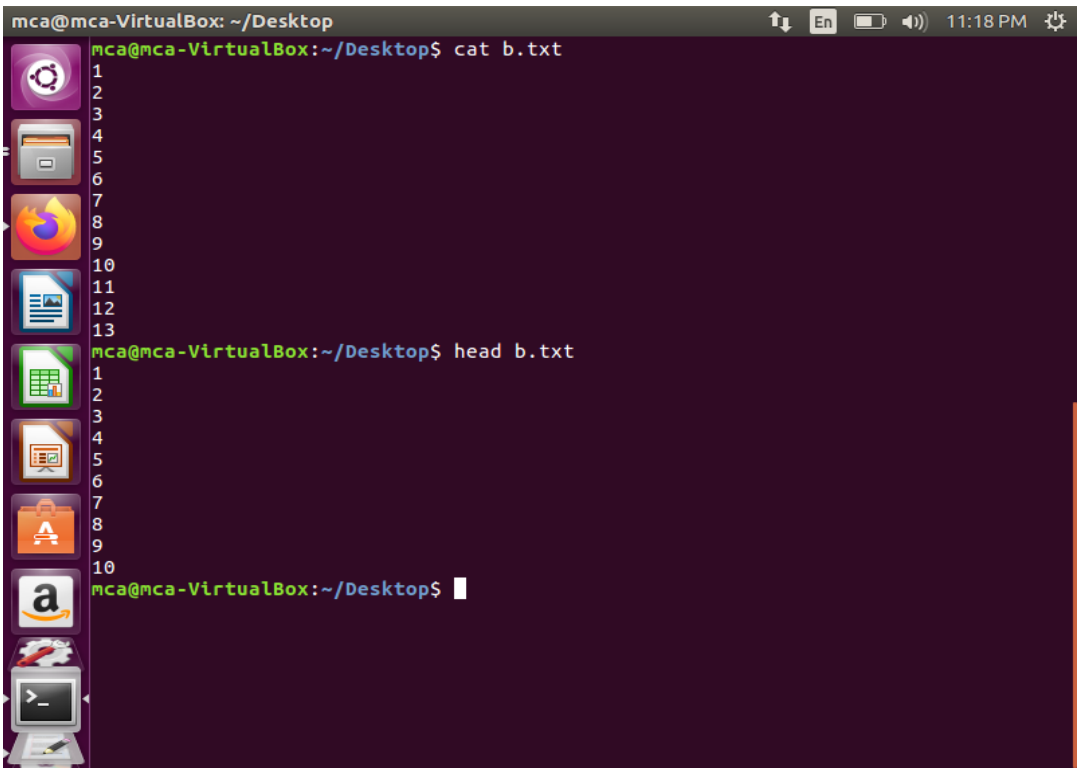
Syntax: *paste [OPTION]... [FILES]...*



21. head

The head command, as the name implies, print the top N number of data of the given input. By default, it prints the first 10 lines of the specified files. If more than one file name is provided then data from each file is preceded by its file name.

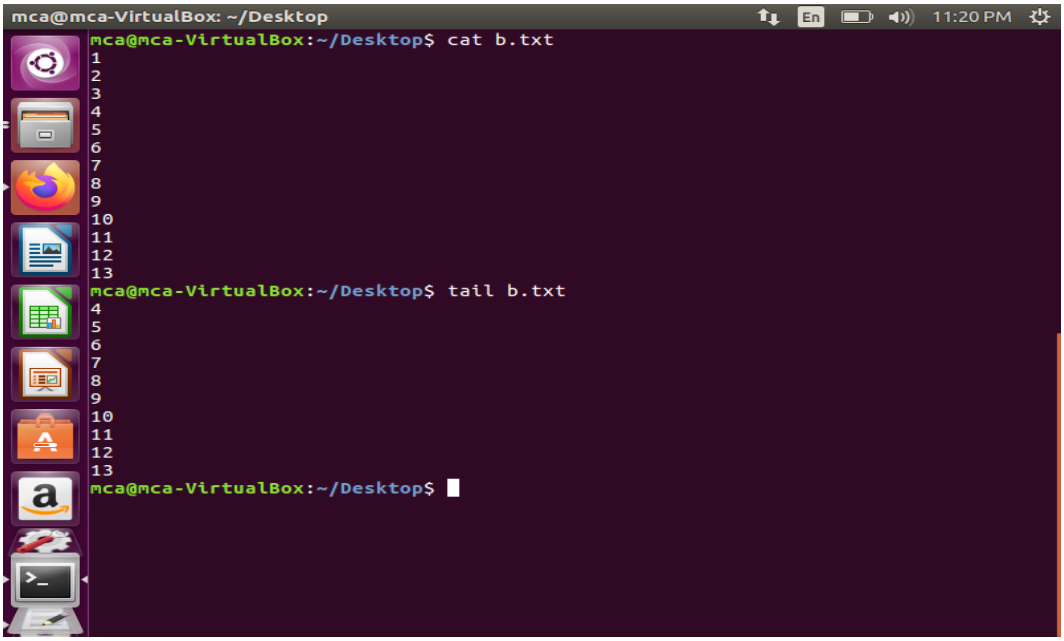
Syntax: *head [OPTION]... [FILE]...*



22. tail

The tail command, as the name implies, print the last N number of data of the given input. By default it prints the last 10 lines of the specified files. If more than one file name is provided then data from each file is precedes by its file name.

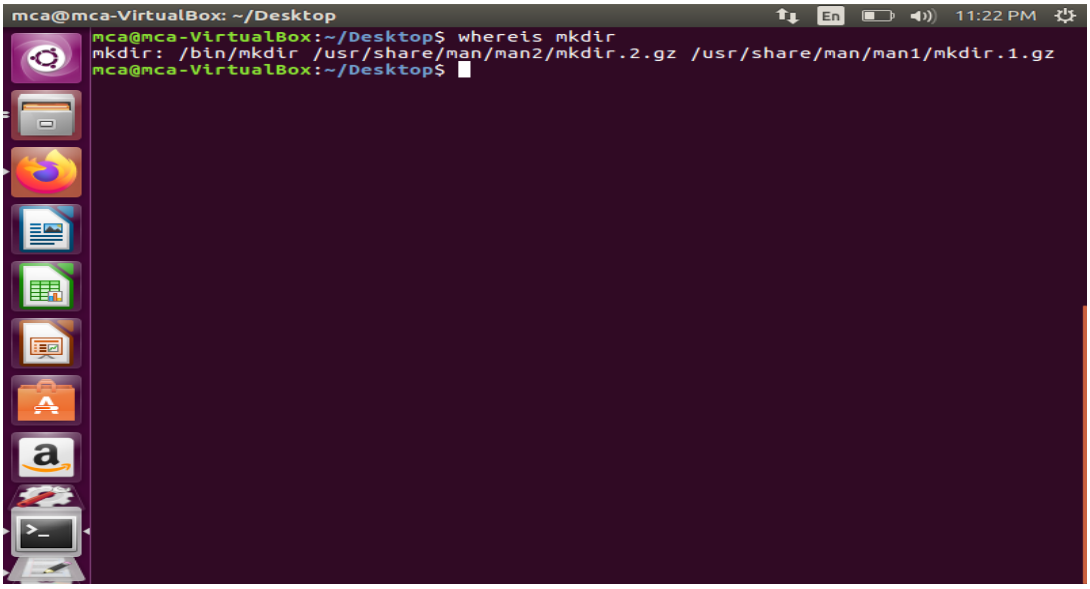
Syntax: *tail [OPTION]... [FILE]...*



23. whereis

whereis command is used to find the location of source/binary file of a command and manuals sections for a specified file in Linux system.

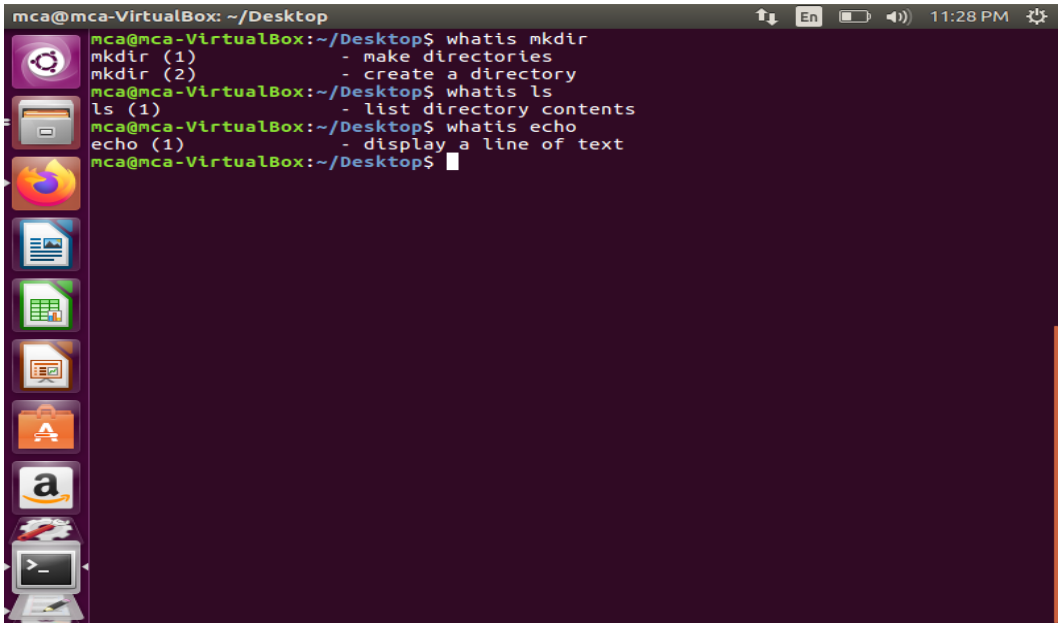
Syntax: *whereis [options] filename...*



24. whatis

whatis command in Linux is used to get a one-line manual page descriptions. In Linux, each manual page has some sort of description within it. So this command search for the manual pages names and show the manual page description of the specified filename or argument.

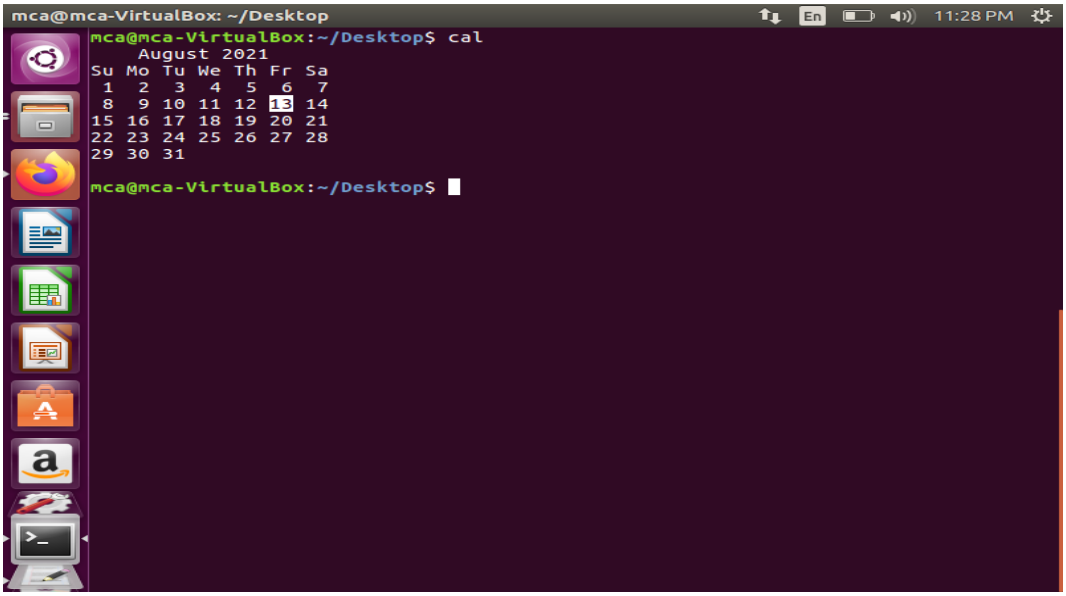
Syntax: *whatis [-dlv?V] [-r|-w] [-s list] [-m system[, ...]] [-M path] [-L locale] [-C file] name ...*



25. cal

cal command is a calendar command in Linux which is used to see the calendar of a specific month or a whole year.

Syntax: *cal [[month] year]*

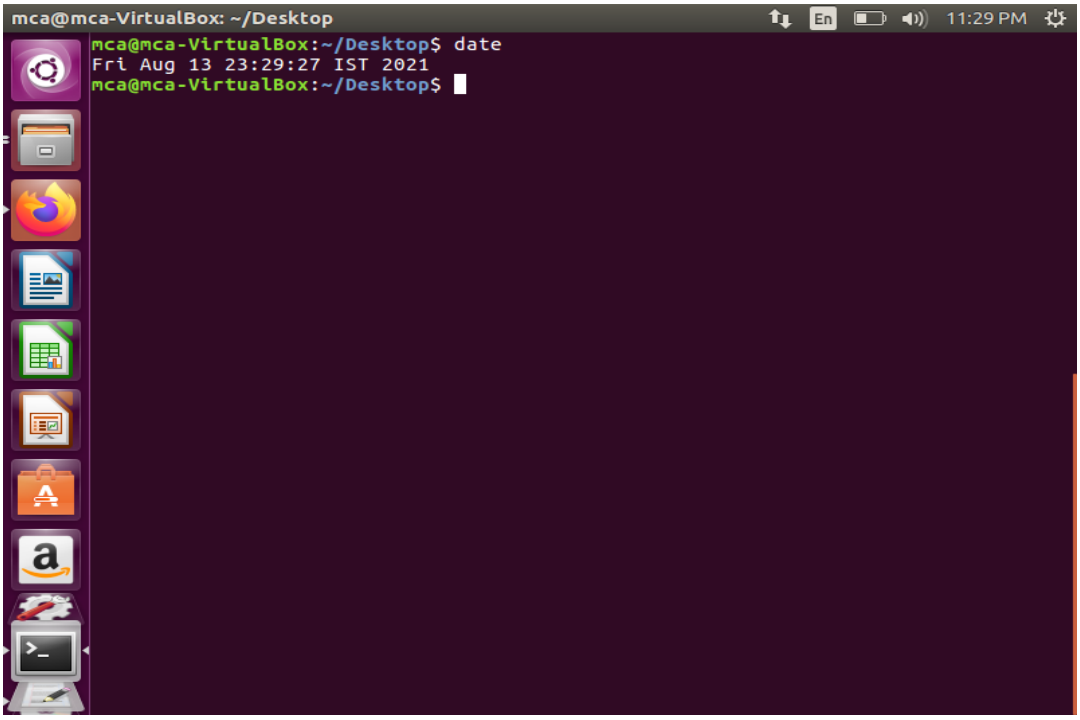


26. date

date command is used to display the system date and time. date command is also used to set date and time of the system.

Syntax: *date [OPTION]... [+FORMAT]*

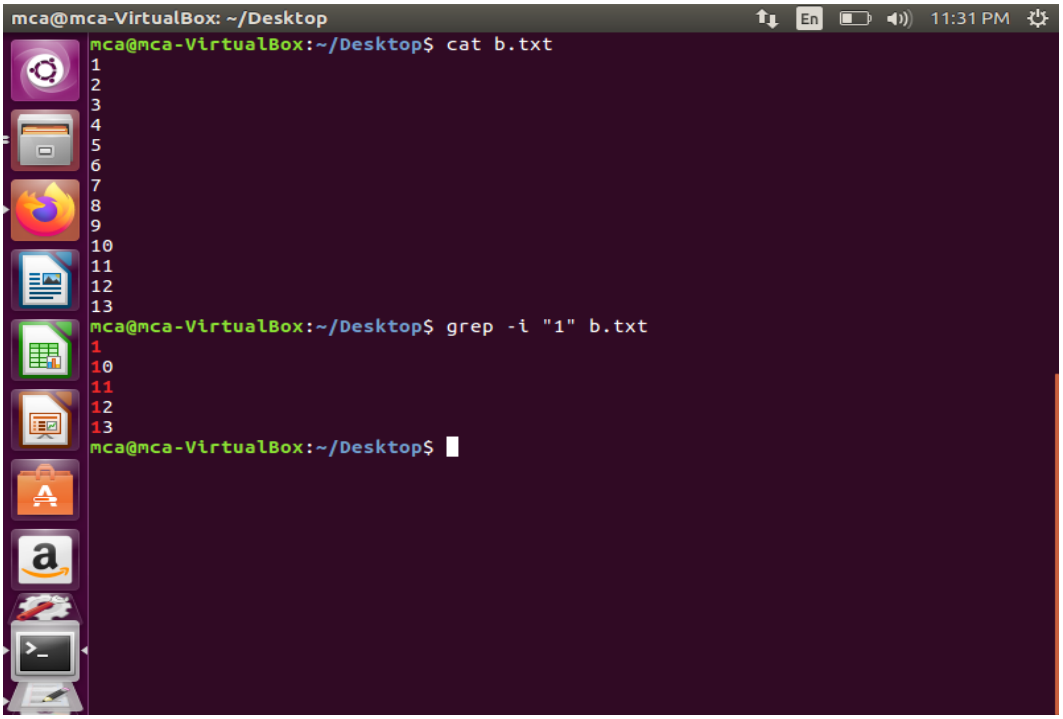
date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]



27. grep

The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern. The pattern that is searched in the file is referred to as the regular expression

Syntax: *grep [options] pattern [files]*

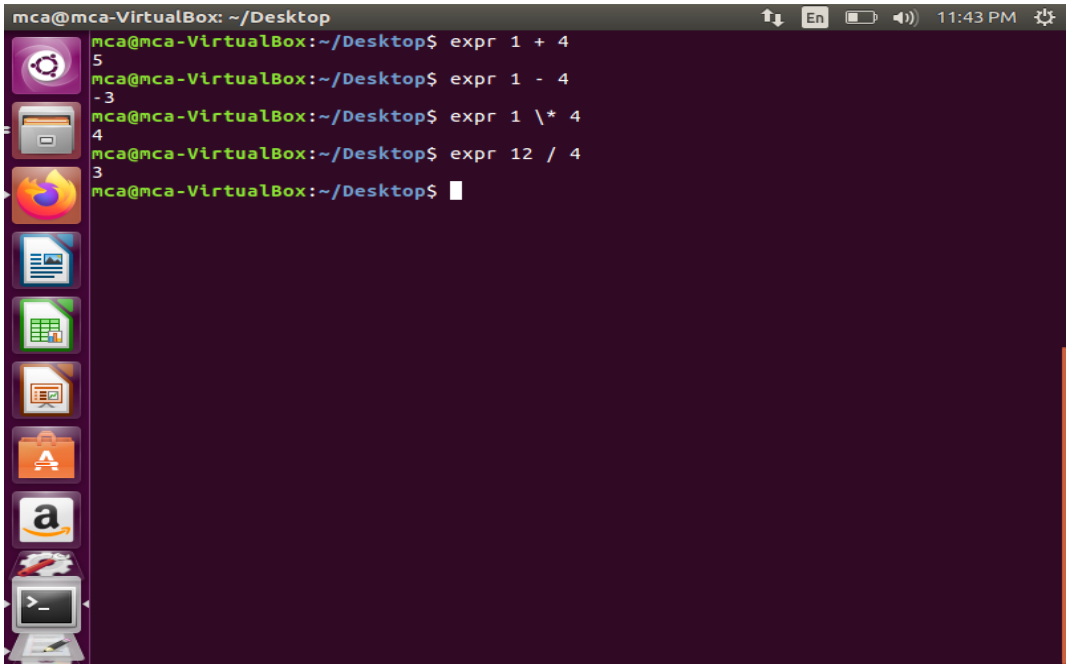


28. expr

The **expr** command in Unix evaluates a given expression and displays its corresponding output. It is used for:

- Basic operations like addition, subtraction, multiplication, division, and modulus on integers.
- Evaluating regular expressions, string operations like substring, length of strings etc.

Syntax: *$expr expression*



29. chmod

In Unix-like operating systems, the chmod command is used to change the access mode of a file. The name is an abbreviation of change mode.

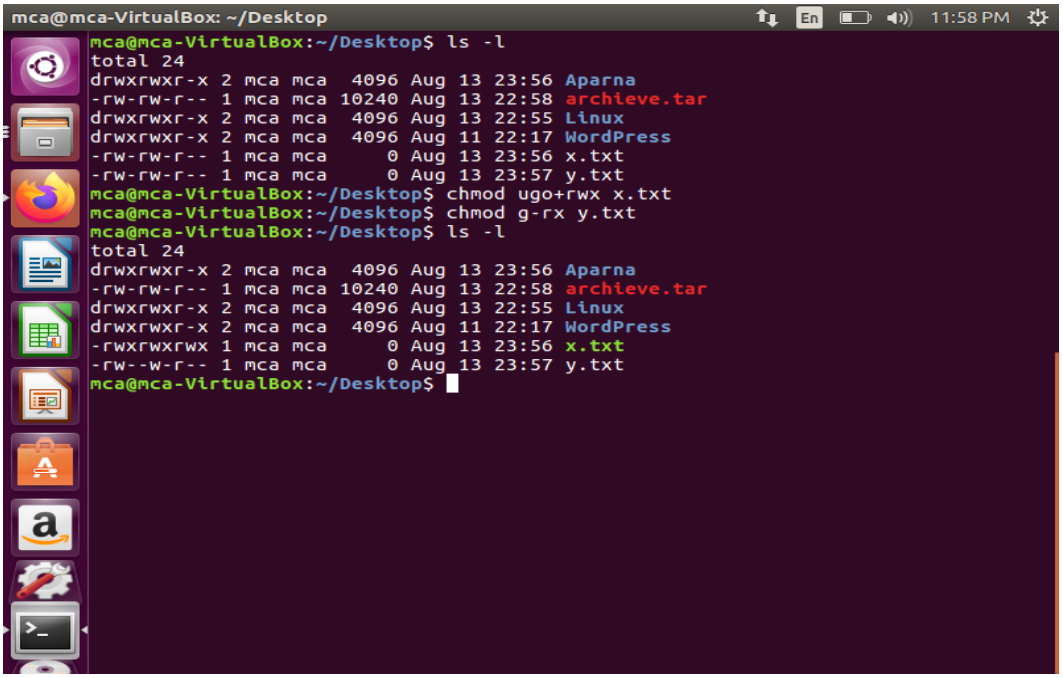
Syntax : *chmod [reference][operator][mode] file...*

The references are used to distinguish the users to whom the permissions apply i.e. they are list of letters that specifies whom to give permissions.

Reference Class	Description
u	owner file's owner
g	group users who are members of the file's group
o	others users who are neither the file's owner nor members of the file's group
a	All three of the above, same as ugo

The operator is used to specify how the modes of a file should be adjusted. The following operators are accepted:

Operator	Description
+	Adds the specified modes to the specified classes
-	Removes the specified modes from the specified classes
=	The modes specified are to be made the exact modes for the specified classes

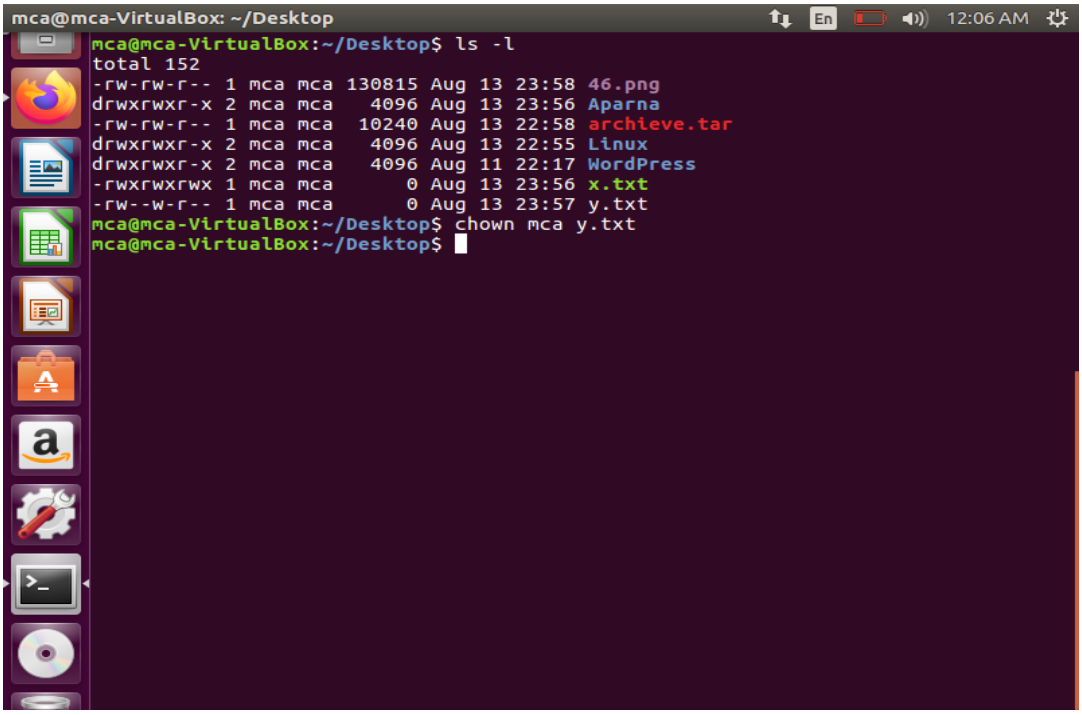


30. chown

chown command is used to change the file Owner or group. Whenever you want to change ownership, you can use chown command.

Syntax: *chown [OPTION]… [OWNER][:[GROUP]] FILE…*

chown [OPTION]… –reference=RFILE FILE…



Experiment No. 3

Aim

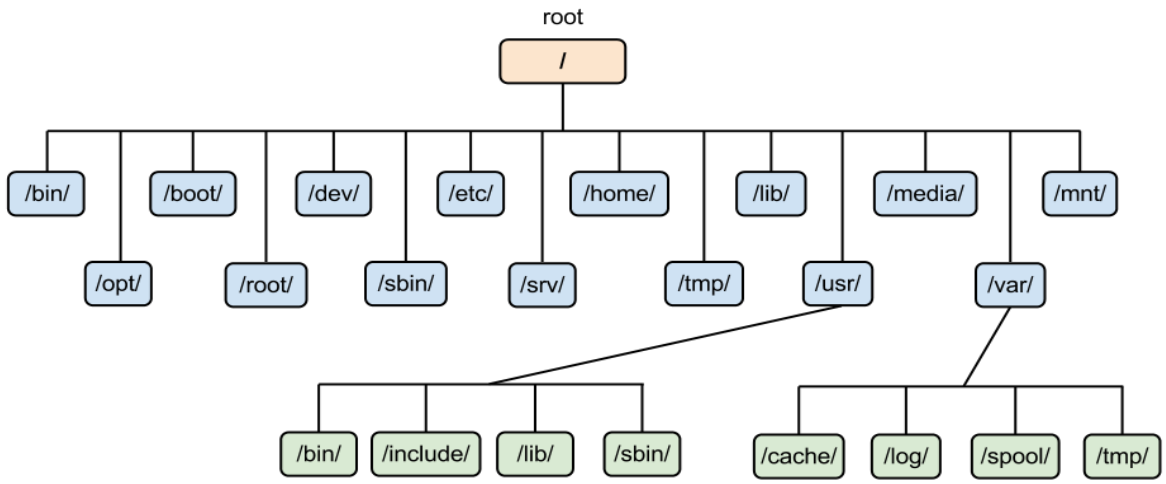
File system hierarchy in a common Linux distribution, file and device permissions, study of system configuration files in /etc, familiarizing log files for system events, user activity, network events.

Result

LINUX File Hierarchy Structure

The Linux File Hierarchy Structure or the Filesystem Hierarchy Standard (FHS) defines the directory structure and directory contents in Unix-like operating systems. It is maintained by the Linux Foundation.

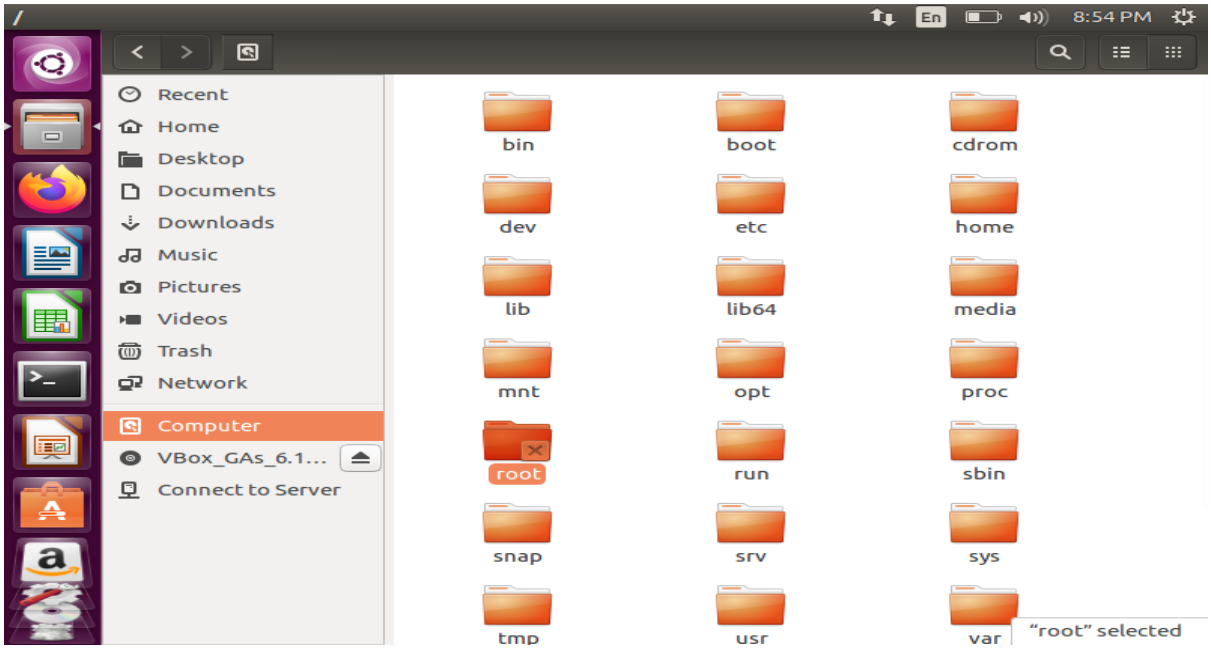
- In the FHS, all files and directories appear under the root directory /, even if they are stored on different physical or virtual devices.
- Some of these directories only exist on a particular system if certain subsystems, such as the X Window System, are installed.
- Most of these directories exist in all UNIX operating systems and are generally used in much the same way; however, the descriptions here are those used specifically for the FHS and are not considered authoritative for platforms other than Linux.



1. / (Root): Primary hierarchy root and root directory of the entire file system hierarchy.

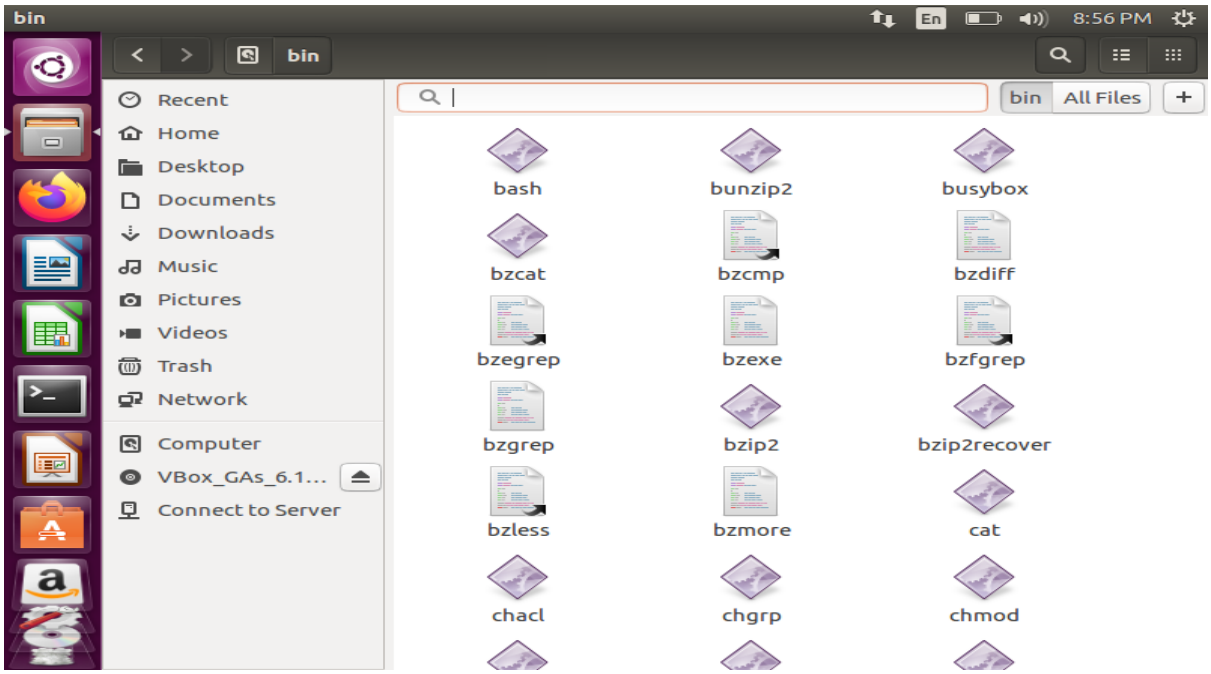
- Every single file and directory starts from the root directory

- The only root user has the right to write under this directory
- /root is the root user's home directory, which is not the same as /



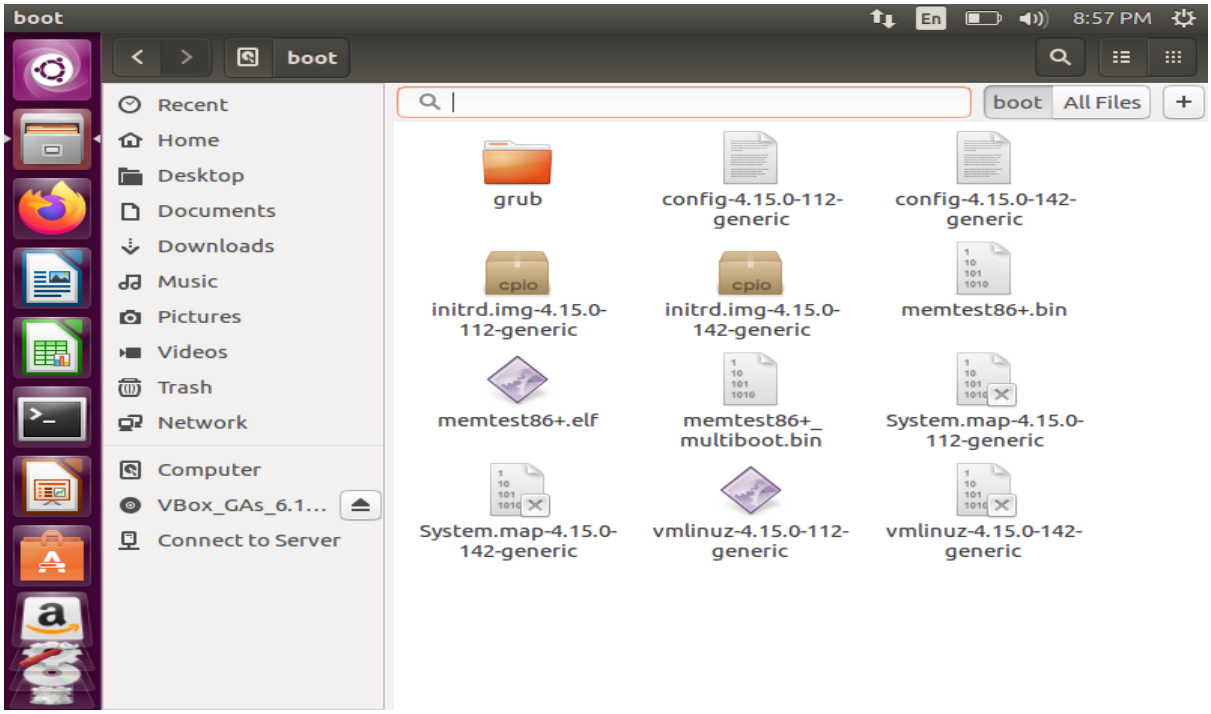
2. /bin : Essential command binaries that need to be available in single-user mode; for all users, e.g., cat, ls, cp.

- Contains binary executables
- Common linux commands you need to use in single-user modes are located under this directory.
- Commands used by all the users of the system are located here e.g. ps, ls, ping, grep, cp



3. /boot : Boot loader files, e.g., kernels, initrd.

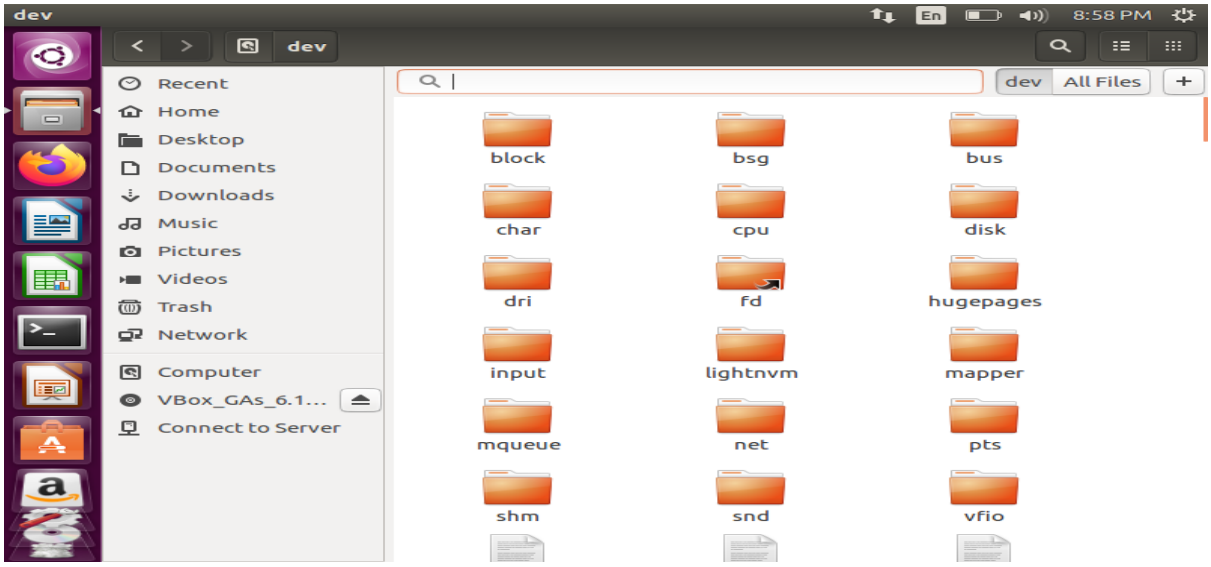
- Kernel initrd, vmlinux, grub files are located under /boot
- Example: initrd.img-2.6.32-24-generic, vmlinuz-2.6.32-24-generic



4. /dev : Essential device files, e.g., /dev/null.

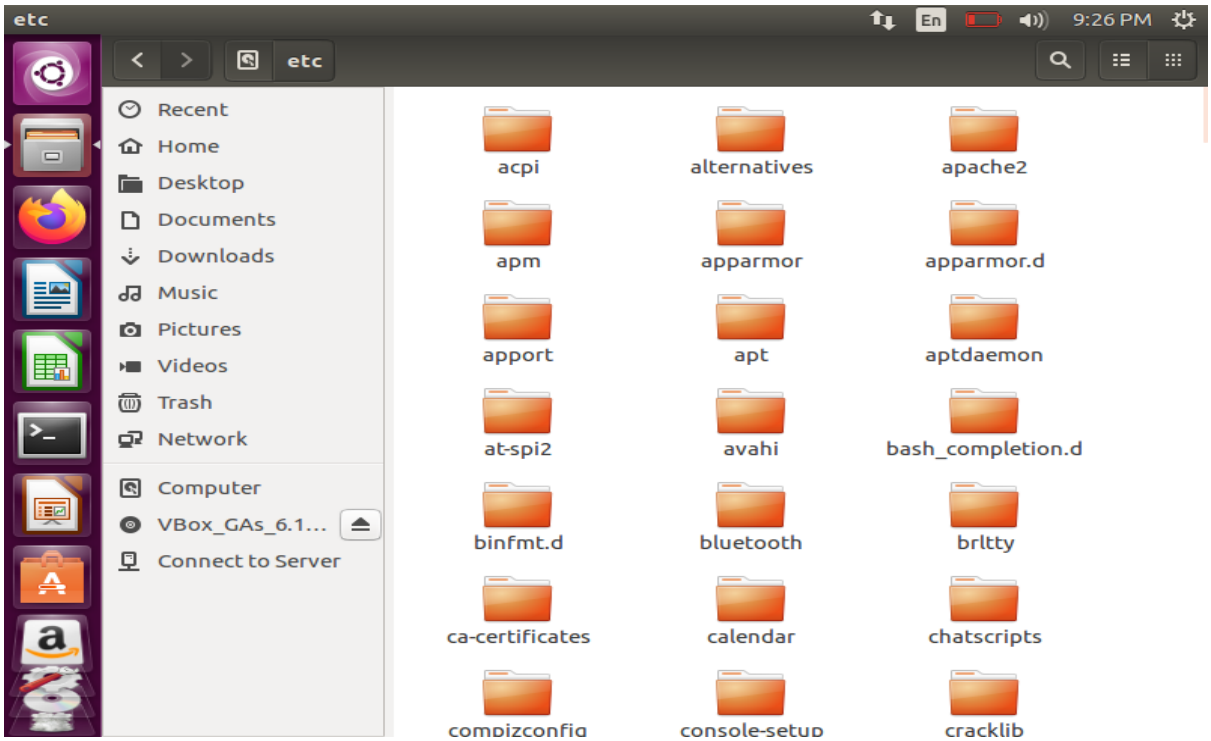
- These include terminal devices, usb, or any device attached to the system.

- Example: /dev/tty1, /dev/usbmon0



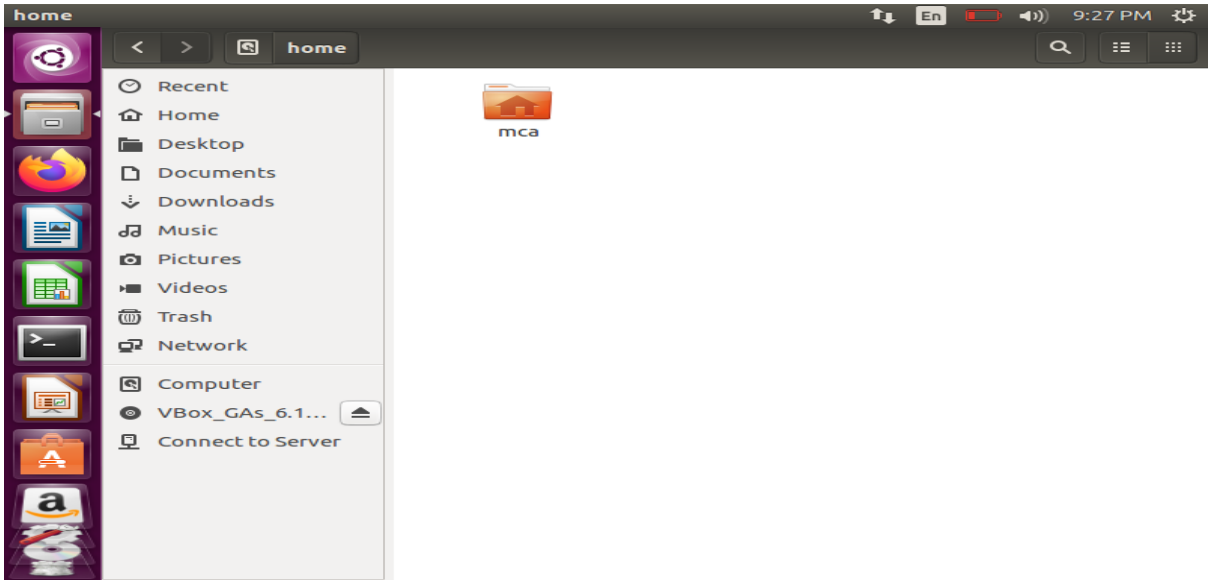
5. /etc : Host-specific system-wide configuration files.

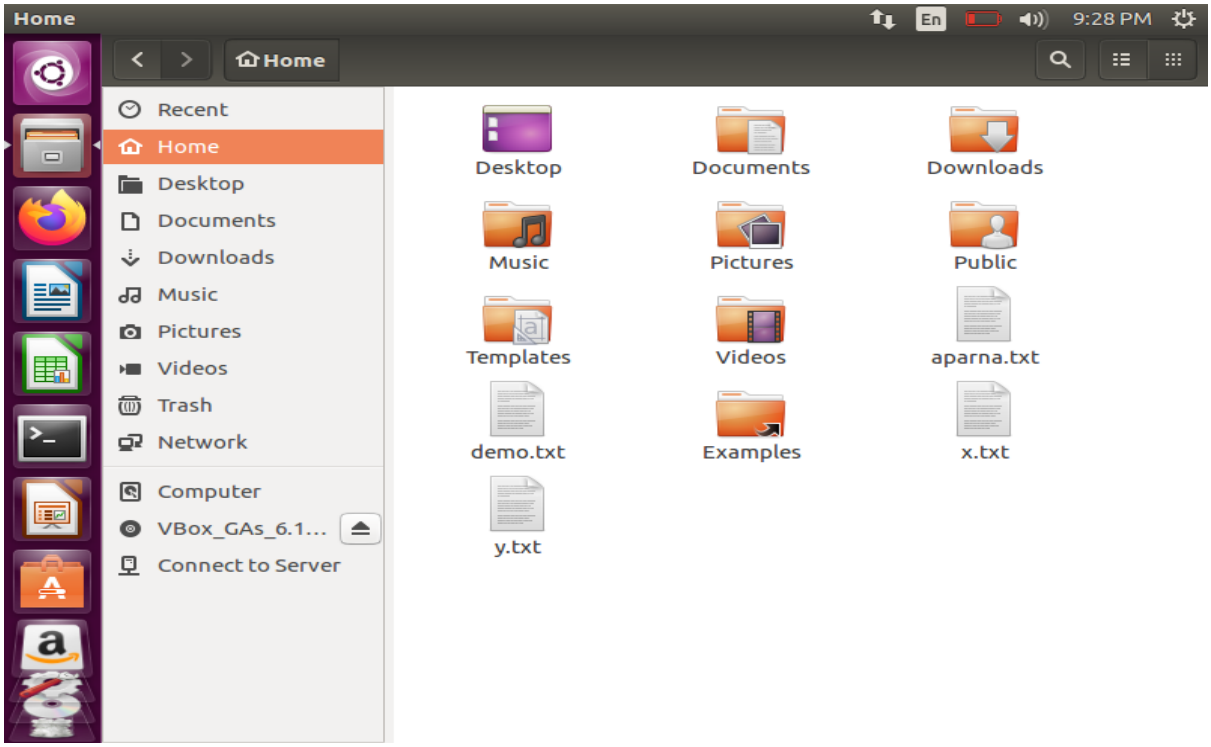
- Contains configuration files required by all programs.
- This also contains startup and shutdown shell scripts used to start/stop individual programs.
- Example: /etc/resolv.conf, /etc/logrotate.conf.



6. /home : Users' home directories, containing saved files, personal settings, etc.

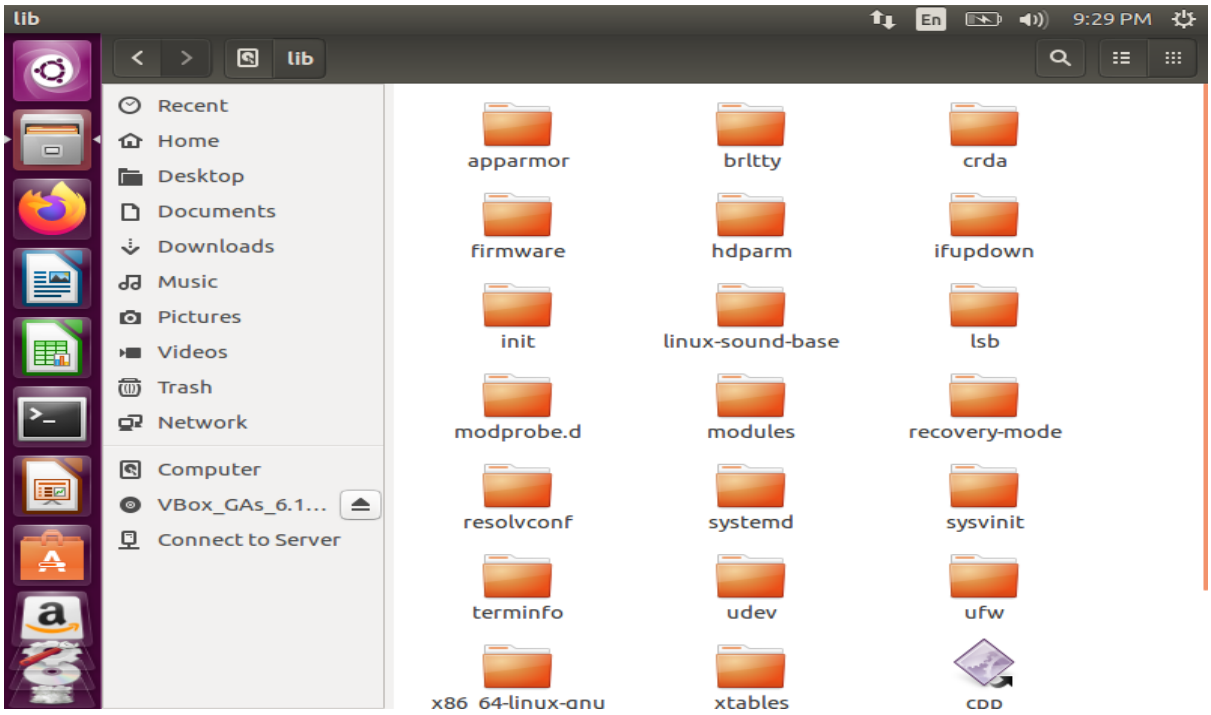
- Home directories for all users to store their personal files.
- example: /home/kishlay, /home/kv





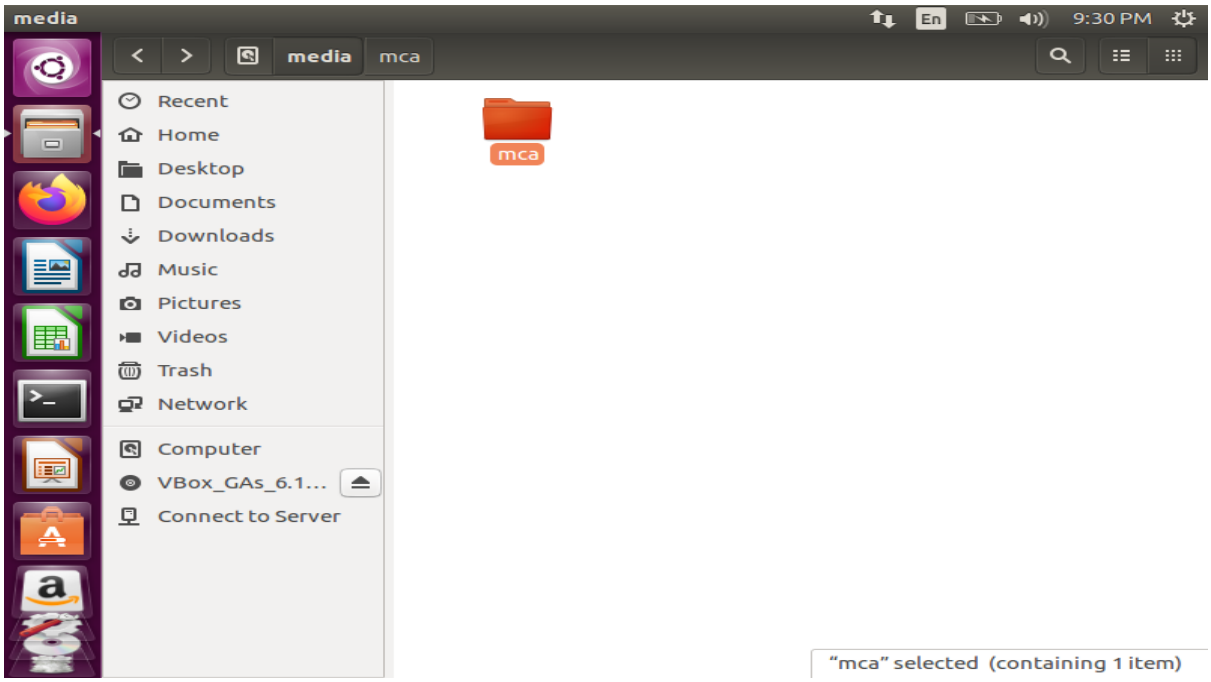
7. /lib : Libraries essential for the binaries in /bin/ and /sbin/.

- Library filenames are either ld* or lib*.so.*
- Example: ld-2.11.1.so, libncurses.so.5.7



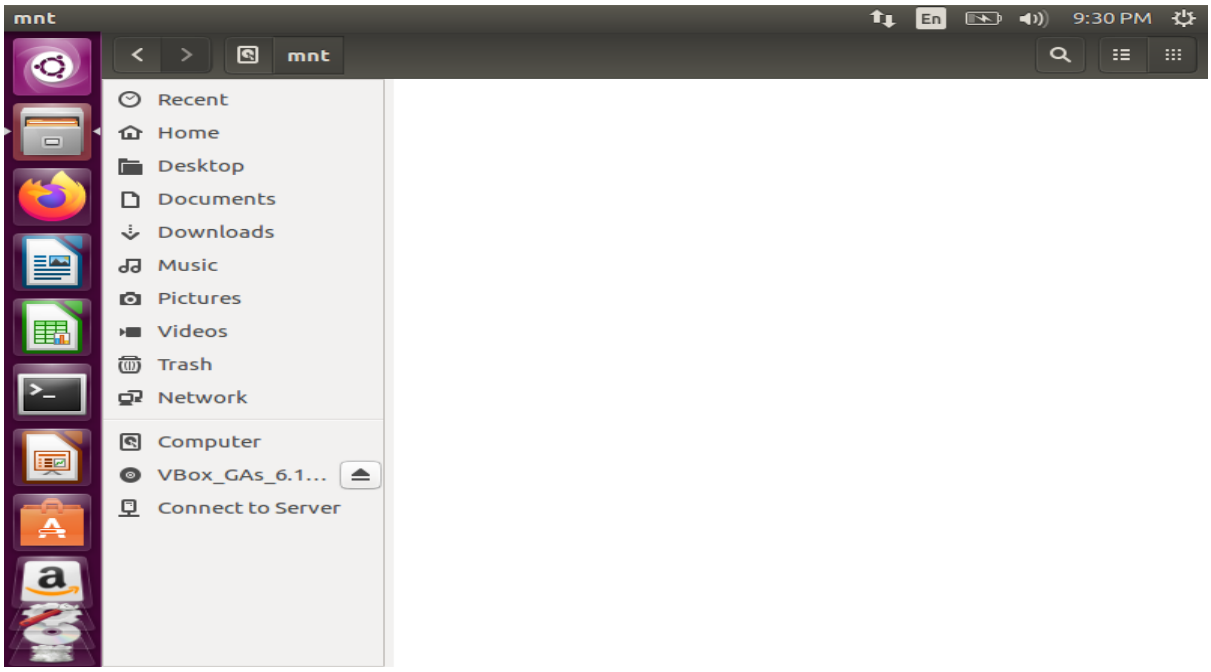
8. /media : Mount points for removable media such as CD-ROMs (appeared in FHS-2.3).

- Temporary mount directory for removable devices.
- Examples, /media/cdrom for CD-ROM; /media/floppy for floppy drives; /media/cdrecorder for CD writer



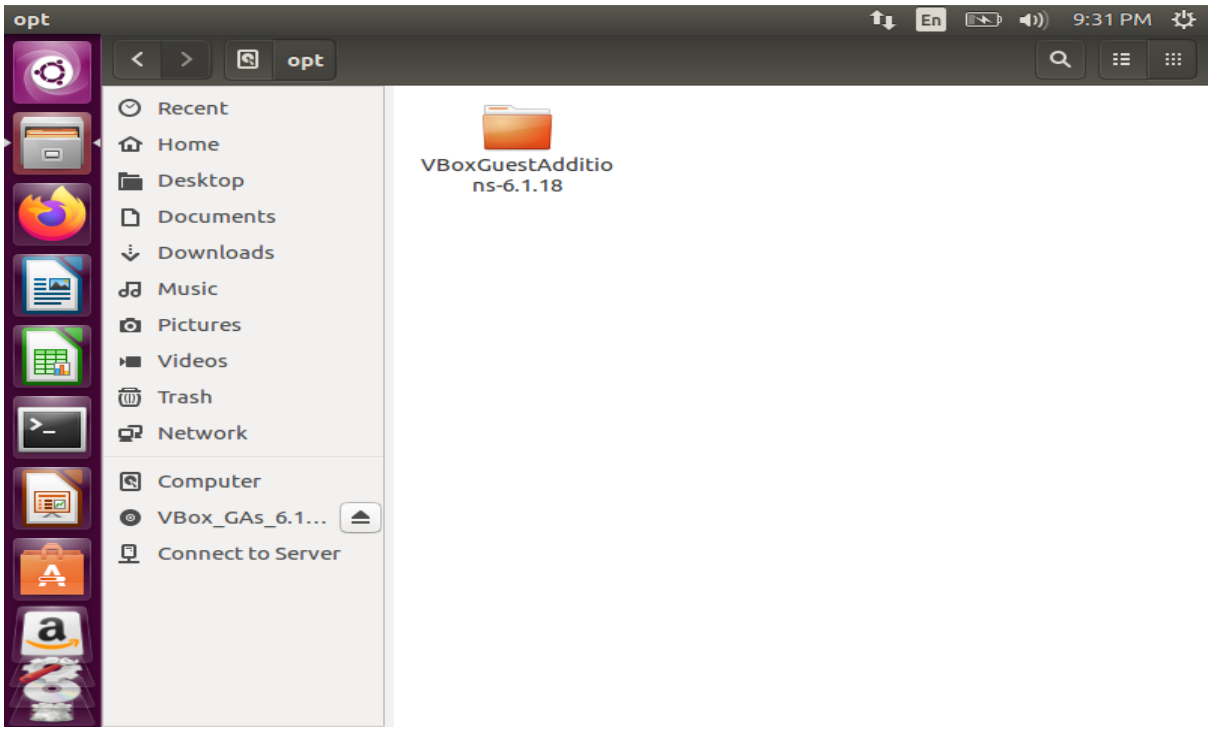
9. /mnt : Temporarily mounted filesystems.

- Temporary mount directory where sysadmins can mount filesystems.



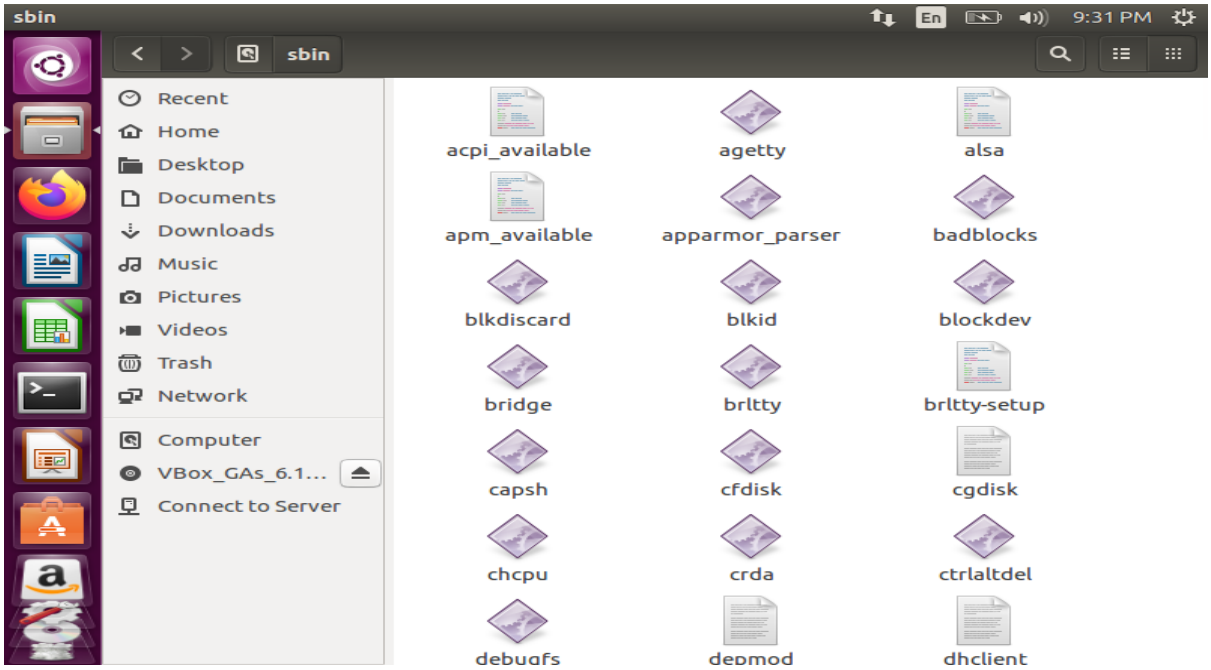
10. /opt : Optional application software packages.

- Contains add-on applications from individual vendors.
- Add-on applications should be installed under either /opt/ or /opt/ sub-directory.



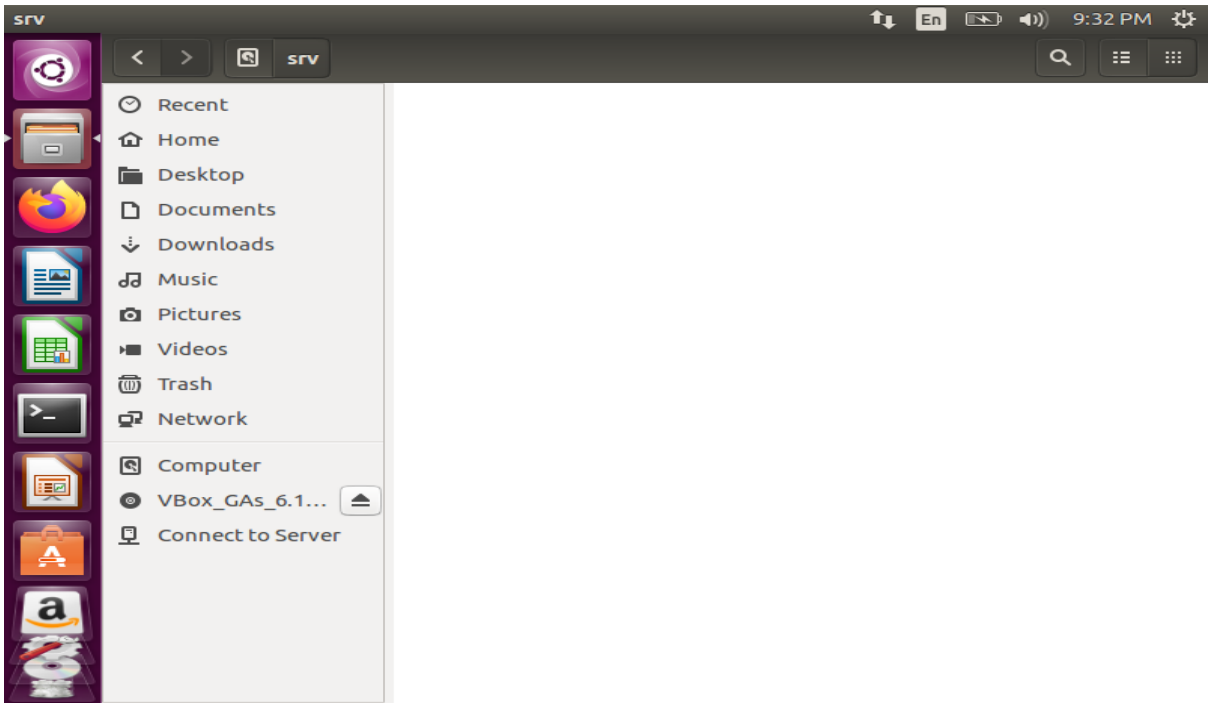
11. /sbin : Essential system binaries, e.g., fsck, init, route.

- Just like /bin, /sbin also contains binary executables.
- The linux commands located under this directory are used typically by system administrator, for system maintenance purpose.
- Example: iptables, reboot, fdisk, ifconfig, swapon



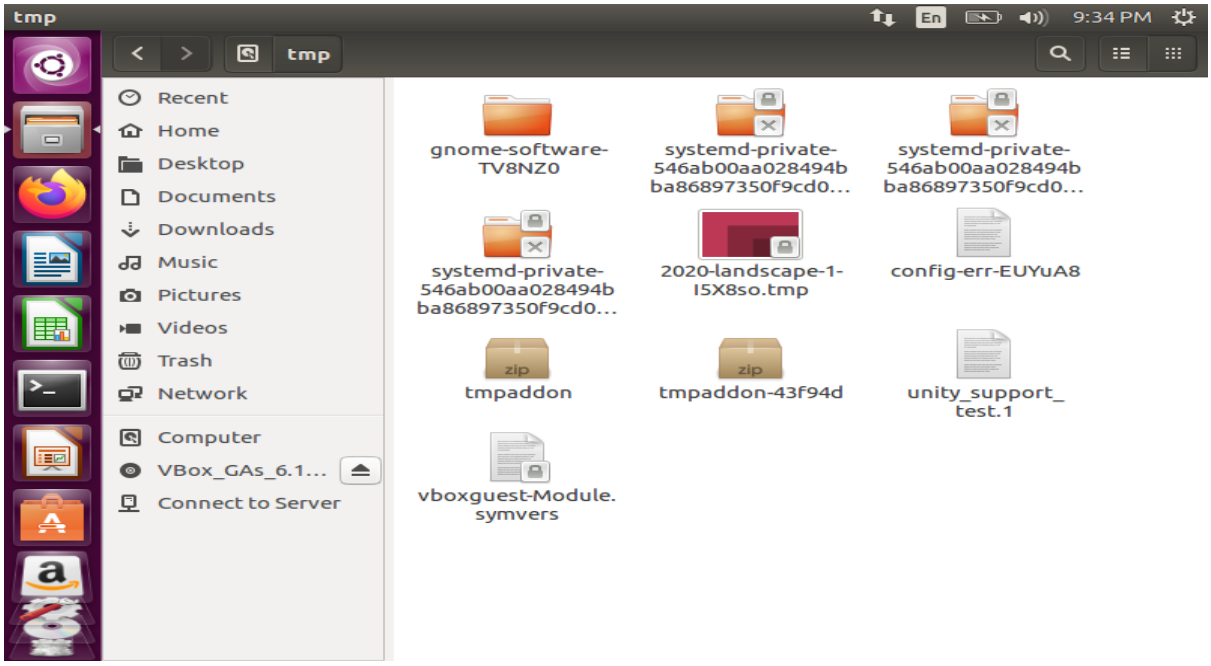
12. /srv : Site-specific data served by this system, such as data and scripts for web servers, data offered by FTP servers, and repositories for version control systems.

- srv stands for service.
- Contains server specific services related data.
- Example, /srv/cvs contains CVS related data.



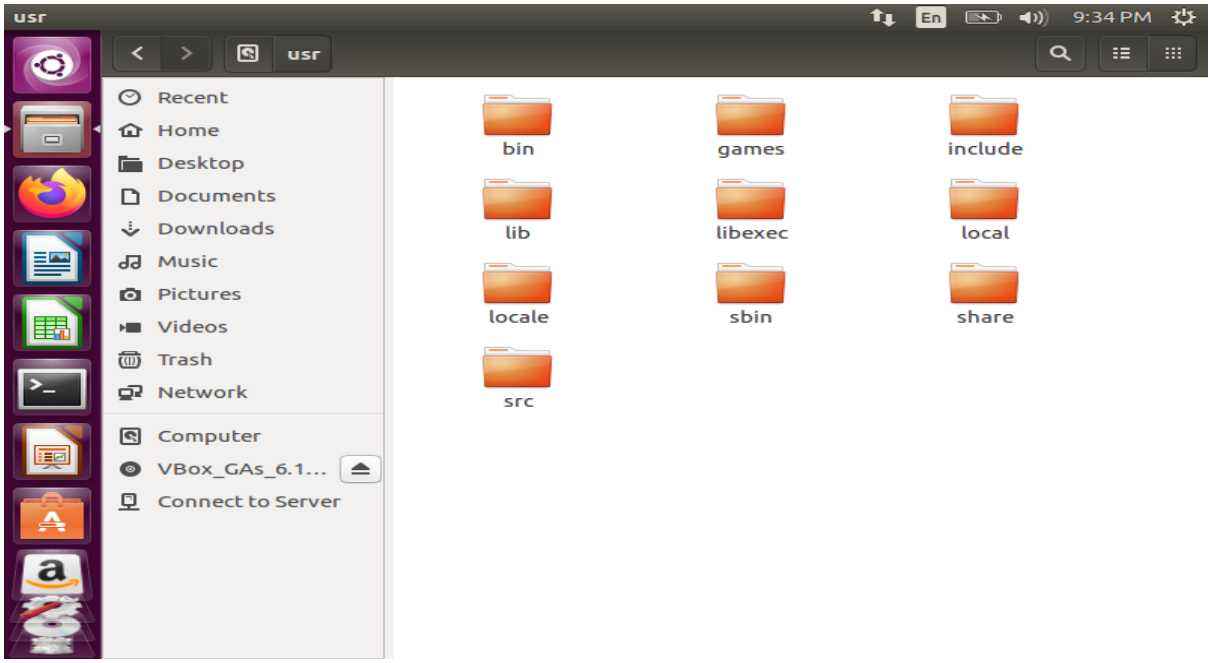
13. /tmp : Temporary files. Often not preserved between system reboots, and may be severely size restricted.

- Directory that contains temporary files created by system and users.
- Files under this directory are deleted when system is rebooted.



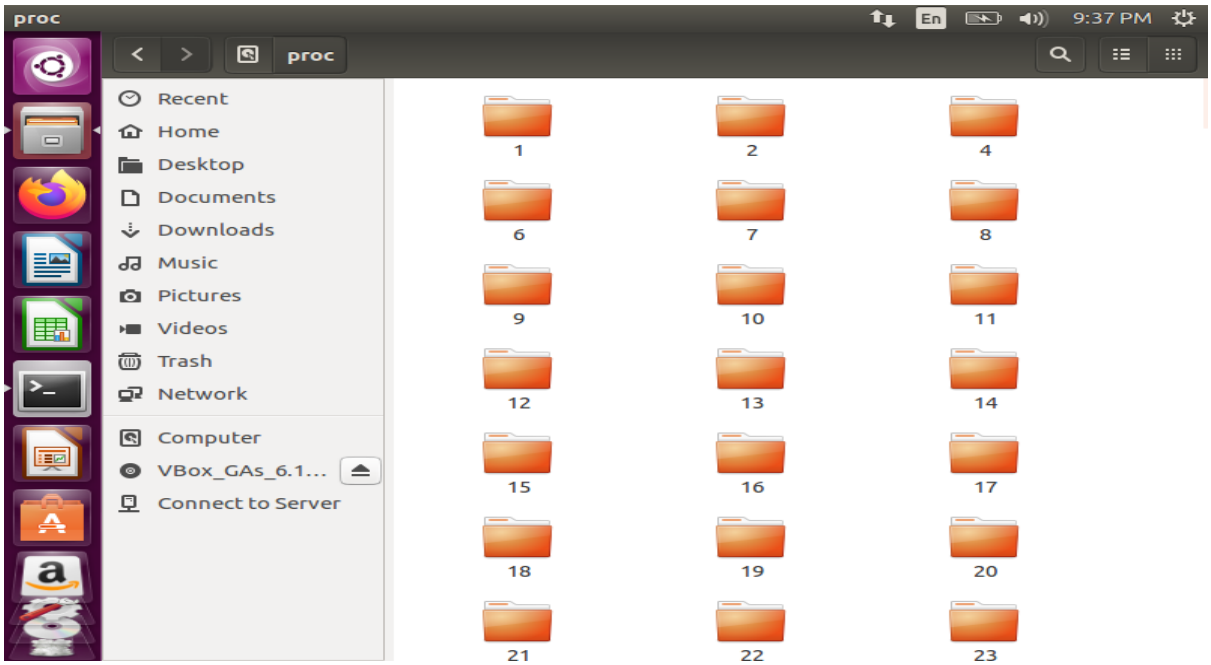
14. /usr : Secondary hierarchy for read-only user data; contains the majority of (multi-)user utilities and applications.

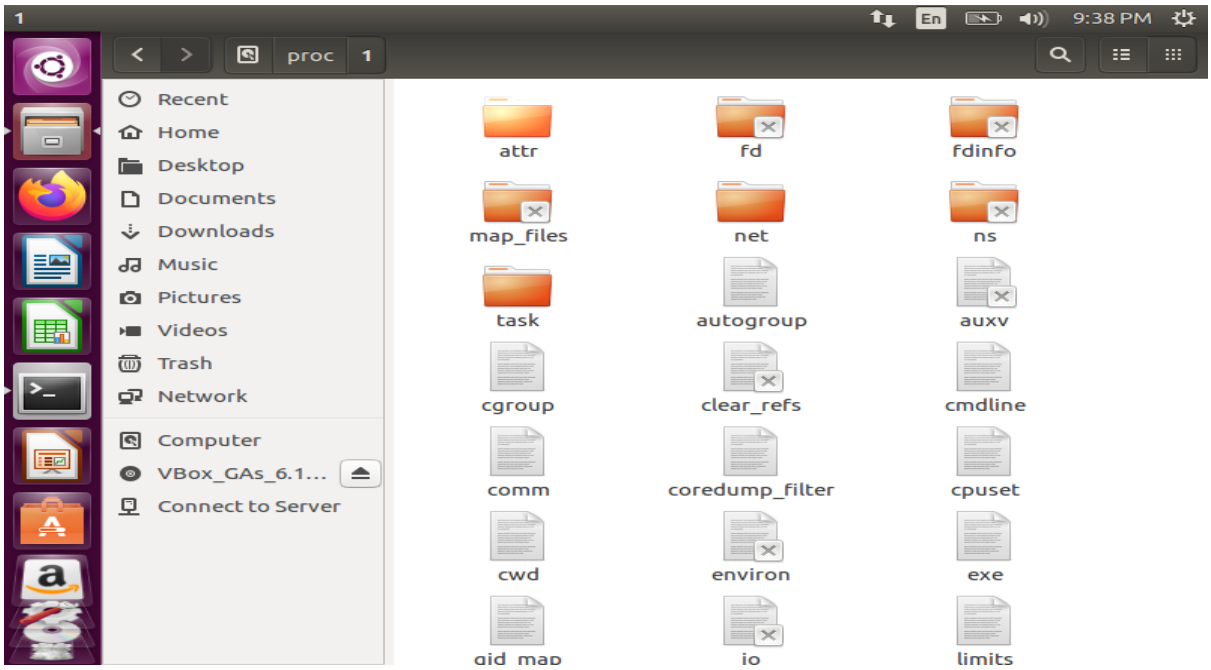
- Contains binaries, libraries, documentation, and source-code for second level programs.
- /usr/bin contains binary files for user programs. If you can't find a user binary under /bin, look under /usr/bin. For example: at, awk, cc, less, scp
- /usr/sbin contains binary files for system administrators. If you can't find a system binary under /sbin, look under /usr/sbin. For example: atd, cron, sshd, useradd, userdel
- /usr/lib contains libraries for /usr/bin and /usr/sbin
- /usr/local contains users programs that you install from source. For example, when you install apache from source, it goes under /usr/local/apache2
- /usr/src holds the Linux kernel sources, header-files and documentation.



15. /proc : Virtual filesystem providing process and kernel information as files. In Linux, corresponds to a procfs mount. Generally automatically generated and populated by the system, on the fly.

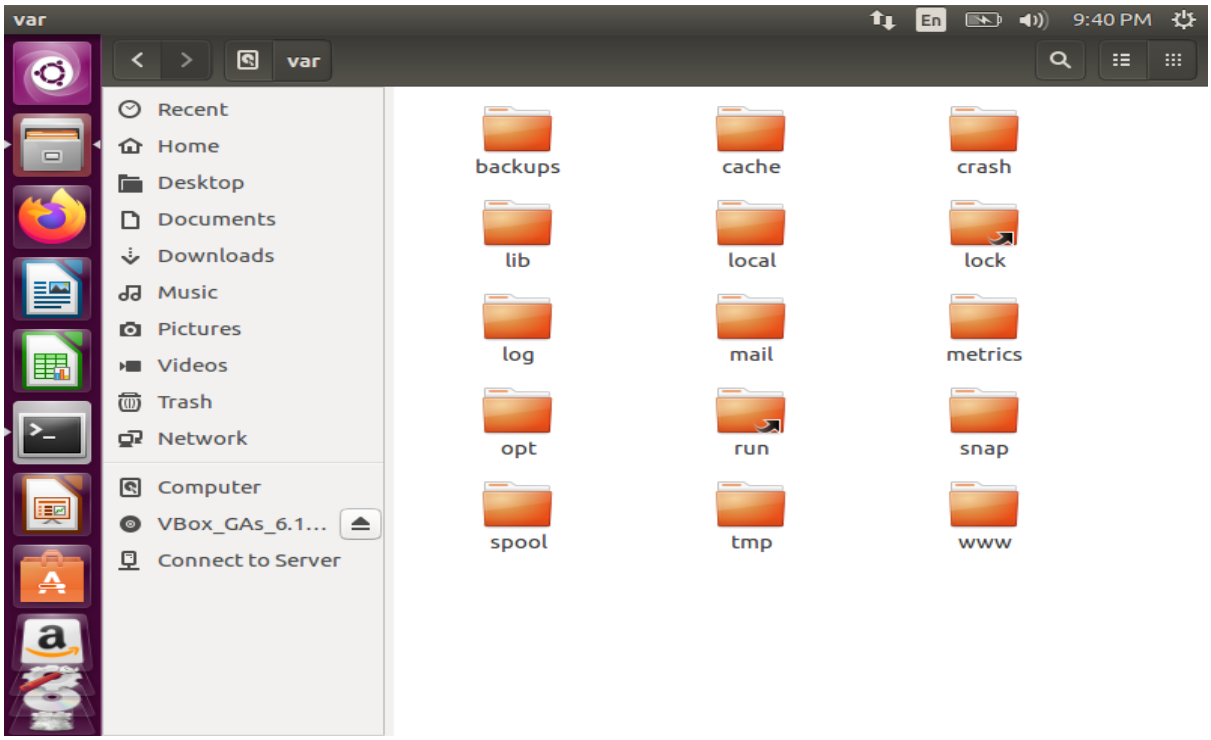
- Contains information about system process.
- This is a pseudo filesystem contains information about running process. For example: /proc/{pid} directory contains information about the process with that particular pid.
- This is a virtual filesystem with text information about system resources. For example: /proc/uptime





16. /var – Variable Files

- var stands for variable files.
- Content of the files that are expected to grow can be found under this directory.
- This includes — system log files (/var/log); packages and database files (/var/lib); emails (/var/mail); print queues (/var/spool); lock files (/var/lock); temp files needed across reboots (/var/tmp);



FILE AND DEVICE PERMISSIONS

Although there are already a lot of good security features built into Linux-based systems, one very important potential vulnerability can exist when local access is granted that is file permission-based issues resulting from a user not assigning the correct permissions to files and directories. So based upon the need for proper permissions, I will go over the ways to assign permissions and show you some examples where modification may be necessary.

Ownership of Linux files

Every file and directory on your Unix/Linux system is assigned 3 types of owner, given below.

User

A user is the owner of the file. By default, the person who created a file becomes its owner. Hence, a user is also sometimes called an owner.

Group

A user- group can contain multiple users. All users belonging to a group will have the same Linux group permissions access to the file. Suppose you have a project where a number of people require access to a file. Instead of manually assigning permissions to each user, you could add all users to a group, and assign group permission to file such that only this group members and no one else can read or modify the files.

Other

Any other user who has access to a file. This person has neither created the file, nor he belongs to a usergroup who could own the file. Practically, it means everybody else. Hence, when you set the permission for others, it is also referred as set permissions for the world.

Permissions

Every file and directory in your UNIX/Linux system has following 3 permissions defined for all the 3 owners discussed above.

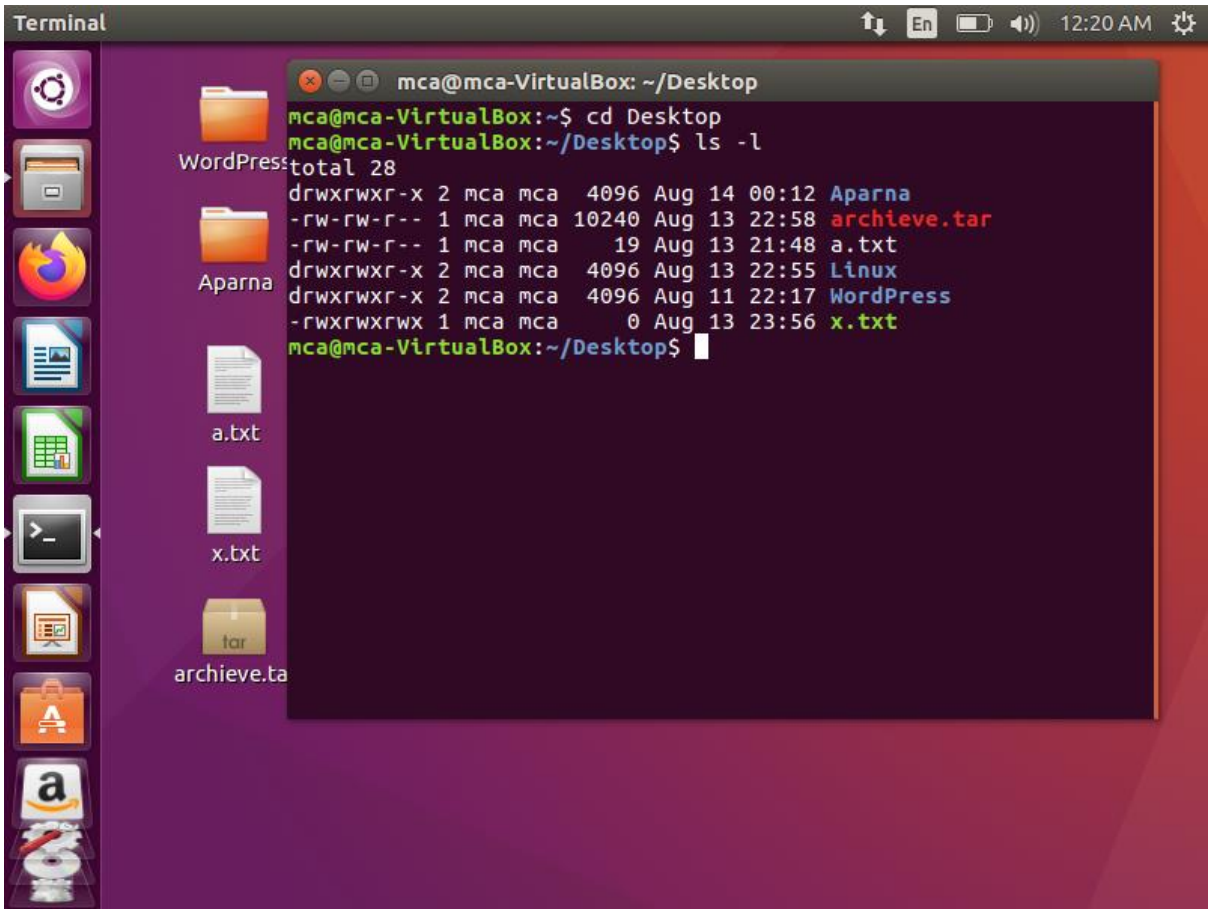
- Read: This permission give you the authority to open and read a file. Read permission on a directory gives you the ability to lists its content.

- Write: The write permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory. Consider a scenario where you have to write permission on file but do not have write permission on the directory where the file is stored. You will be able to modify the file contents. But you will not be able to rename, move or remove the file from the directory.

- Execute: In Windows, an executable program usually has an extension ".exe" and which you can easily run. In Unix/Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code(provided read & write permissions are set), but not run it.

Let's see file permissions in Linux with examples:

ls -l used to list information about files and directories within the file system.



If the first '-' implies that we have selected a file. Else, if it were a directory, d would have been shown.

The characters are pretty easy to remember.

r read permission
w write permission
x execute permission
- no permission

The first part of the code is 'rw-'. This suggests that the owner 'Home' can:

- Read the file
- Write or edit the file
- He cannot execute the file since the execute bit is set to '-'.

The second part is 'rw-'. It for the user group 'Home' and group-members can:

- Read the file
- Write or edit the file

The third part is for the world which means any user. It says 'r--'. This means the user can only:

- Read the file

Changing file/directory permissions with 'chmod' command

We can use the 'chmod' command which stands for 'change mode'. Using the command, we can set permissions (read, write, execute) on a file/directory for the owner, group and the world.

Syntax: *chmod permissions filename*

There are 2 ways to use the command:

1. Absolute mode
2. Symbolic mode

Absolute (Numeric) Mode

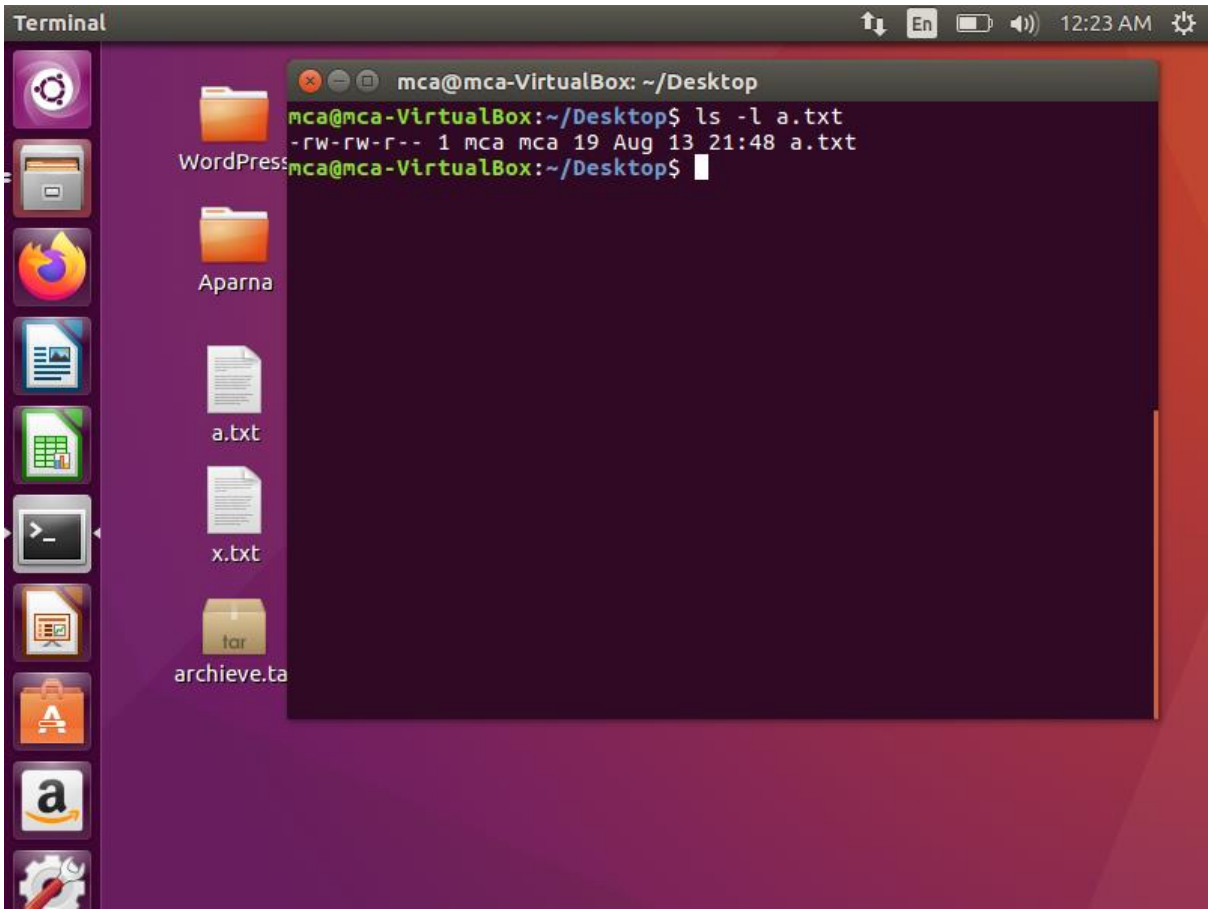
In this mode, file permissions are not represented as characters but a three-digit octal number.

The table below gives numbers for all for permissions types.\

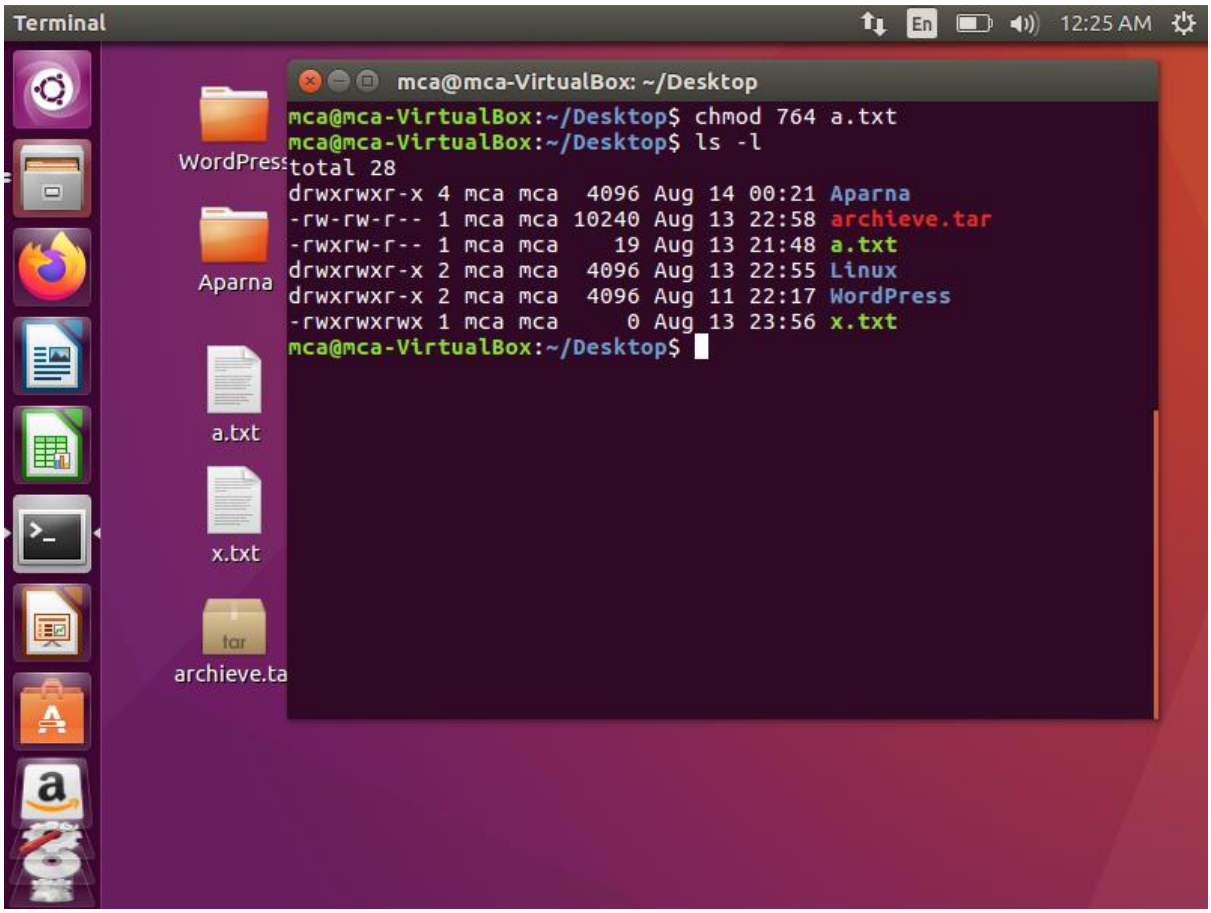
Number	Permission Type	Symbol
0	No Permission	---
1	Execute	--x
2	Write	-w-
3	Execute + Write	-wx
4	Read	r--
5	Read + Execute	r-x
6	Read +Write	rw-
7	Read + Write +Execute	rwx

Let's see the chmod permissions command in action.

1. Checking current file permissions



2. chmod 764 and checking file permission again



In the above-given terminal window, we have changed the permissions of the file 'sample to '764'.

'764' absolute code says the following:

Owner can read, write and execute , Usergroup can read and write , World can only read

This is shown as '-rwxrw-r—

Symbolic Mode

In the Absolute mode, you change permissions for all 3 owners. In the symbolic mode, you can modify permissions of a specific owner. It makes use of mathematical symbols to modify the Unix file permissions.

Operator	Description
+	Adds a permission to a file or directory
-	Removes the permission

=	Sets the permission and overrides the permissions set earlier.

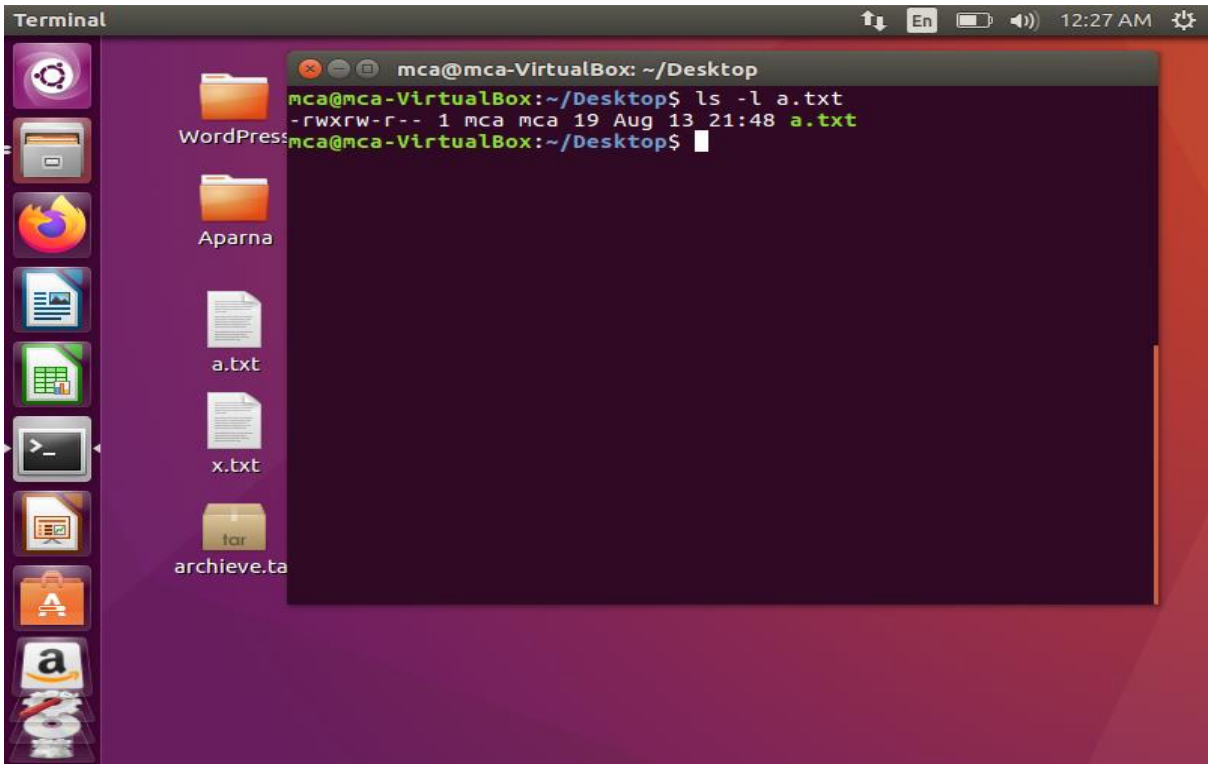
The various owners are represented as:

User Denotations:

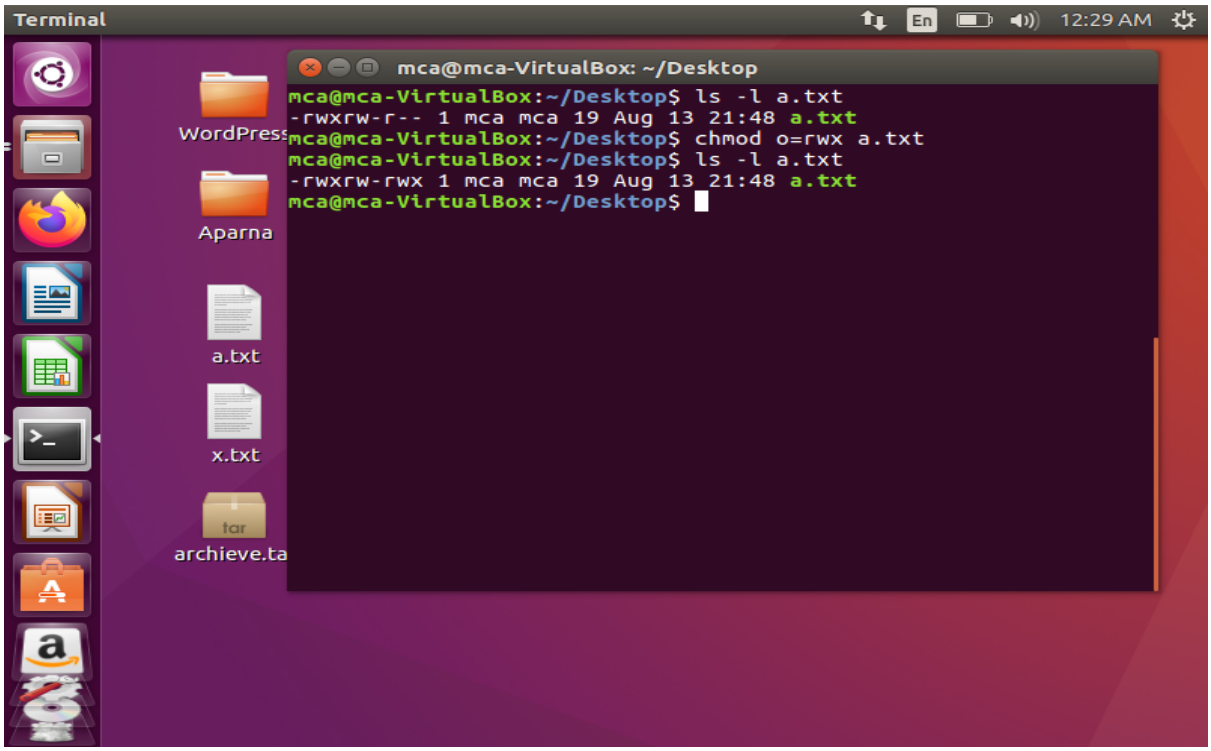
u	user/owner
g	group
o	other
a	all

We will not be using permissions in numbers like 755 but characters like rwx. Let's look into an example:

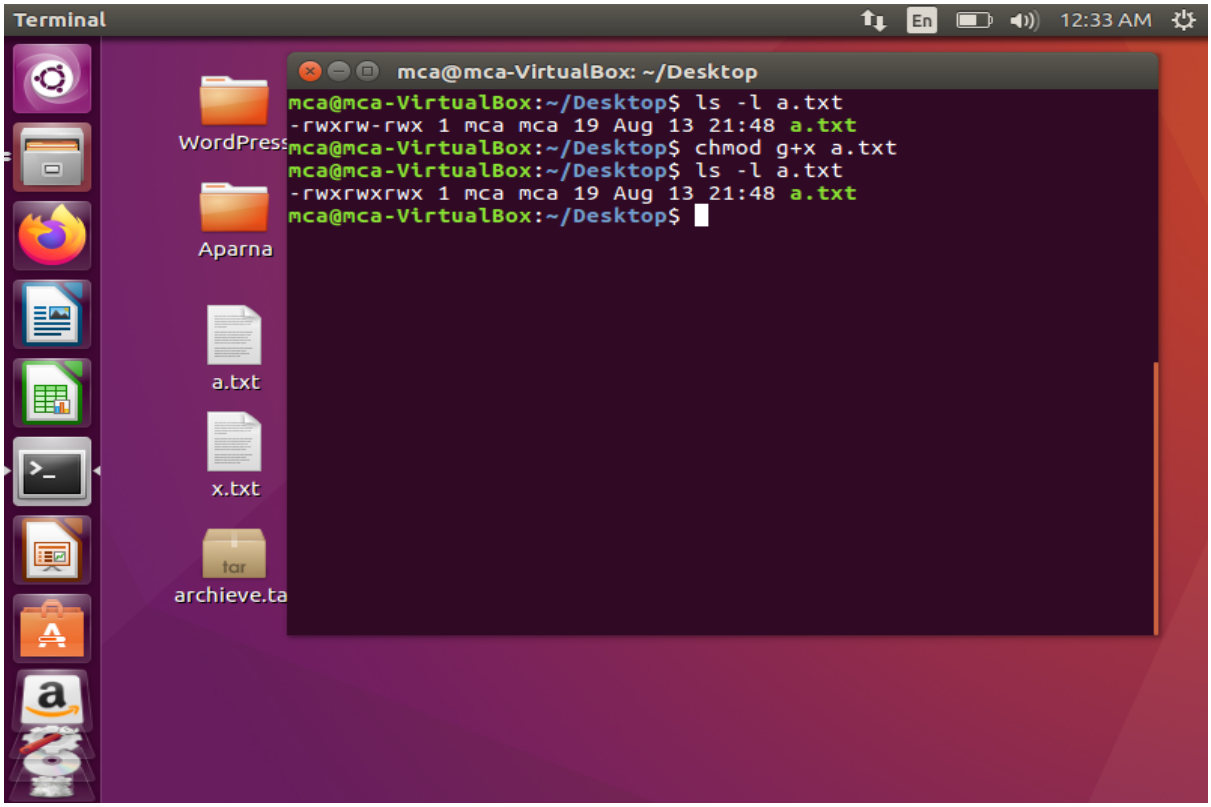
1. Current file permissions



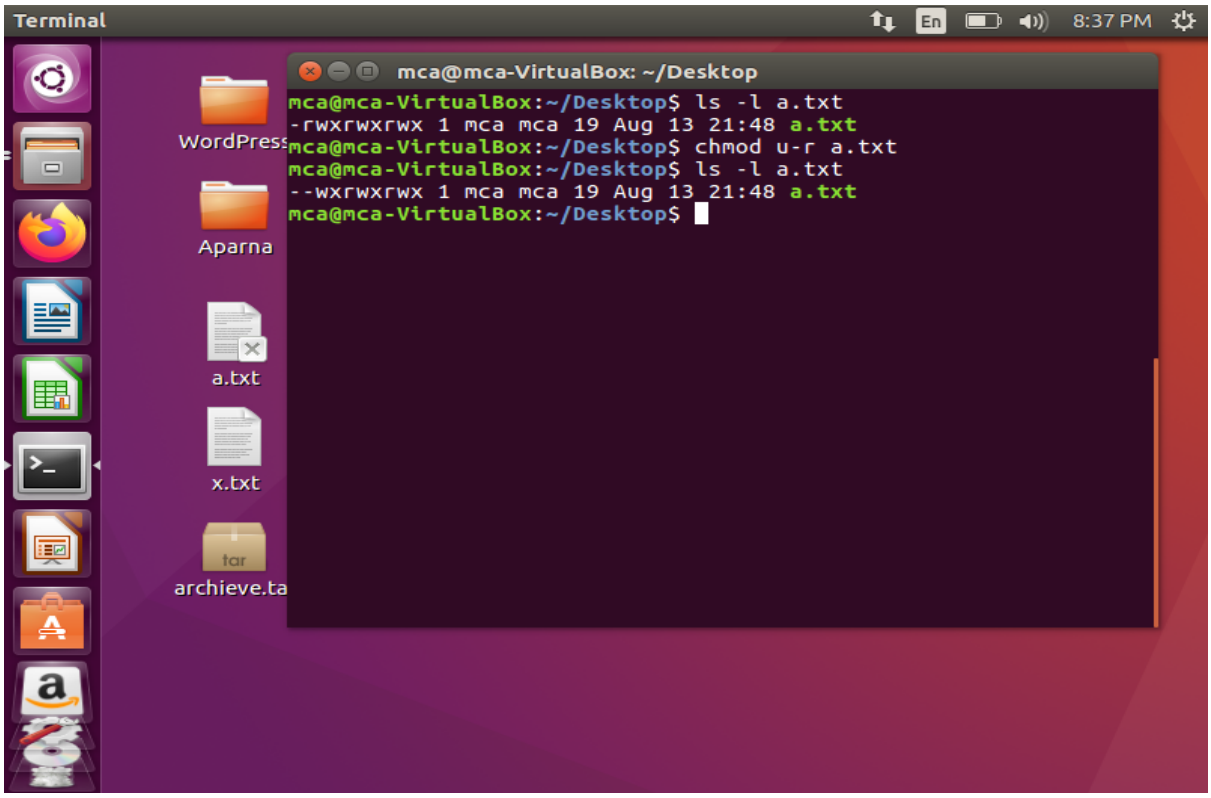
2. Setting permission to ‘other’ users



3. Adding ‘execute’ permission to usergroups



4. Removing ‘read’ permission for ‘user’



/etc : Host-specific system configuration

Purpose

The /etc hierarchy contains configuration files. A "configuration file" is a local file used to control the operation of a program; it must be static and cannot be an executable binary. It is recommended that files be stored in subdirectories of /etc rather than directly in /etc.

Requirements

No binaries may be located under /etc. The following directories, or symbolic links to directories are required in /etc:

Directory	Description
opt	Configuration for /opt

Specific Options

The following directories, or symbolic links to directories must be in /etc, if the corresponding subsystem is installed:

Directory	Description
X11	Configuration for the X Window system (optional)
sgml	Configuration for SGML (optional)
xml	Configuration for XML (optional)

The following files, or symbolic links to files, must be in /etc if the corresponding subsystem is installed:

File	Description
csh.login	Systemwide initialization file for C shell logins (optional)
exports	NFS filesystem access control list (optional)
fstab	Static information about filesystems (optional)
ftpusers	FTP daemon user access control list (optional)
gateways	File which lists gateways for routed (optional)
gettydefs	Speed and terminal settings used by getty (optional)
group	User group file (optional)
host.conf	Resolver configuration file (optional)
hosts	Static information about host names (optional)
hosts.allow	Host access file for TCP wrappers (optional)
hosts.deny	Host access file for TCP wrappers (optional)
hosts.equiv	List of trusted hosts for rlogin, rsh, rcp (optional)
hosts.lpd	List of trusted hosts for lpd (optional)
inetd.conf	Configuration file for inetd (optional)
inittab	Configuration file for init (optional)
issue	Pre-login message and identification file (optional)
ld.so.conf	List of extra directories to search for shared libraries (optional)
motd	Post-login message of the day file (optional)
mtab	Dynamic information about filesystems (optional)

File	Description
mtools.conf	Configuration file for mtools (optional)
networks	Static information about network names (optional)
passwd	The password file (optional)
printcap	The lpd printer capability database (optional)
profile	Systemwide initialization file for sh shell logins (optional)
protocols	IP protocol listing (optional)
resolv.conf	Resolver configuration file (optional)
rpc	RPC protocol listing (optional)
securetty	TTY access control for root login (optional)
services	Port names for network services (optional)
shells	Pathnames of valid login shells (optional)
syslog.conf	Configuration file for syslogd (optional)

Linux log files

Log files are the records that Linux stores for administrators to keep track and monitor important events about the server, kernel, services, and applications running on it. In this post, we'll go over the top Linux log files server administrators should monitor.

Log files are a set of records that Linux maintains for the administrators to keep track of important events. They contain messages about the server, including the kernel, services and applications running on it.

Linux provides a centralized repository of log files that can be located under the /var/log directory.

The log files generated in a Linux environment can typically be classified into four different categories:

- Application Logs
- Event Logs
- Service Logs
- System Logs\

Common Linux log files names and usage

/var/log/messages : General message and system related stuff
/var/log/auth.log : Authenication logs

/var/log/kern.log : Kernel logs
/var/log/cron.log : Crond logs (cron job)
/var/log/maillog : Mail server logs
/var/log/qmail/ : Qmail log directory (more files inside this directory)
/var/log/httpd/ : Apache access and error logs directory
/var/log/lighttpd/ : Lighttpd access and error logs directory
/var/log/nginx/ : Nginx access and error logs directory
/var/log/apt/ : Apt/apt-get command history and logs directory
/var/log/boot.log : System boot log
/var/log/mysqld.log : MySQL database server log file
/var/log/secure or /var/log/auth.log : Authentication log
/var/log/utmp or /var/log/wtmp : Login records file
/var/log/yum.log or /var/log/dnf.log : Yum/Dnf command log file.

Experiment No.4

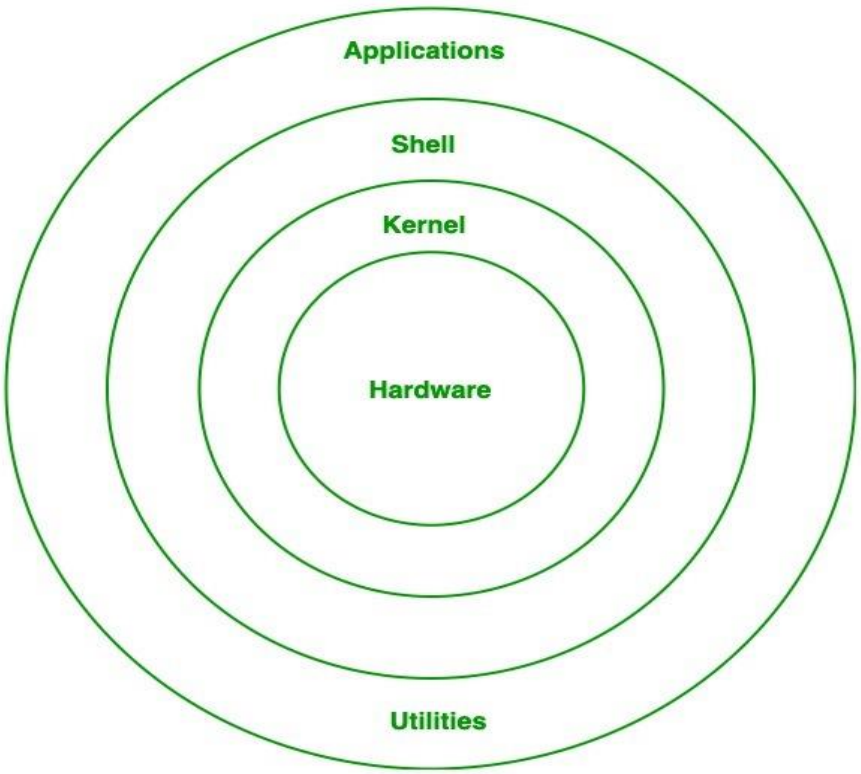
Aim

Shell scripting: study bash syntax, environment variables, variables, control constructs such as if, for and while, aliases and functions, accessing command line arguments passed to shell scripts. Study of startup scripts login and logout scripts, familiarity with systemd and system 5 init scripts is expected.

Result

Shell Scripting

A shell is special user program which provide an interface to user to use operating system services. Shell accept human readable commands from user and convert them into something which kernel can understand. It is a command language interpreter that execute commands read from input devices such as keyboards or from files. The shell gets started when the user logs in or start the terminal.



An Operating is made of many components, but its two prime components are:

- Kernel
- Shell

A Kernel is at the nucleus of a computer. It makes the communication between the hardware and software possible. While the Kernel is the innermost part of an operating system, a shell is the outermost one.A shell in a Linux operating system takes input from you in the form of commands, processes it, and then gives an output. It is the interface through which a user works on the programs, commands, and scripts. A shell is accessed by a terminal which runs it.

When you run the terminal, the Shell issues a command prompt (usually $), where you can type your input, which is then executed when you hit the Enter key. The output or the result is thereafter displayed on the terminal.The Shell wraps around the delicate interior of an Operating system protecting it from accidental damage. Hence the name Shell.

Shell is broadly classified into two categories –

- Command Line Shell
- Graphical shell

Command Line Shell : Shell can be accessed by user using a command line interface. A special program called Terminal in linux/macOS or Command Prompt in Windows OS is provided to type in the human readable commands such as “cat”, “ls” etc. and then it is being execute.

Graphical Shells : Graphical shells provide means for manipulating programs based on graphical user interface (GUI), by allowing for operations such as opening, closing, moving and resizing windows, as well as switching focus between windows. Window OS or Ubuntu OS can be considered as good example which provide GUI to user for interacting with program. User do not need to type in command for every actions.

Shell Prompt

The prompt, $, which is called the command prompt, is issued by the shell. While the prompt is displayed, you can type a command.

Shell reads your input after you press Enter. It determines the command you want executed by looking at the first word of your input. A word is an unbroken set of characters. Spaces and tabs separate words.

Shell Types

In Unix, there are two major types of shells −

- Bourne shell − If you are using a Bourne-type shell, the **$** character is the default prompt.
- C shell − If you are using a C-type shell, the % character is the default prompt.

The Bourne Shell has the following subcategories −

- Bourne shell (sh)
- Korn shell (ksh)
- Bourne Again shell (bash)
- POSIX shell (sh)

The different C-type shells follow −

- C shell (csh)
- TENEX/TOPS C shell (tcsh)

Bash

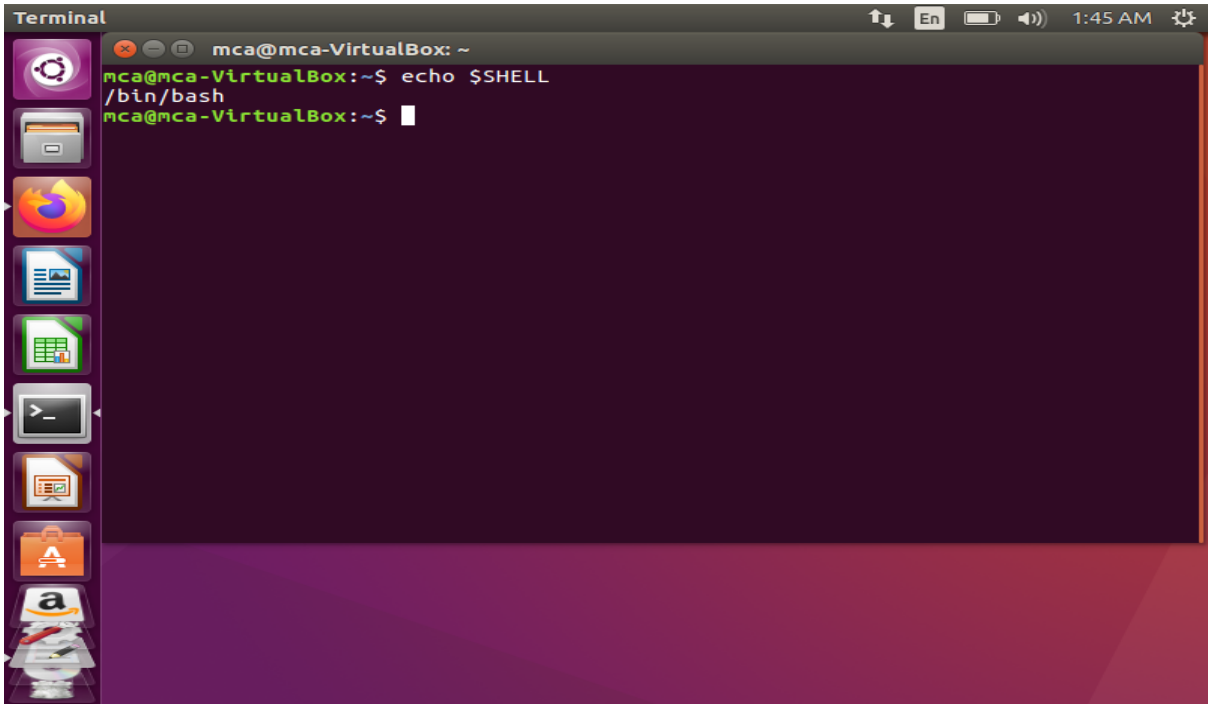
BASH is an acronym for Bourne Again Shell. Bash is a shell program written by Brian Fox as an upgraded version of Bourne Shell program '**sh**'. It is an open-source GNU project. It was released in 1989 as one of the most popular shell distribution of GNU/Linux operating systems. It provides functional improvements over Bourne Shell for both programming and interactive uses. It includes command line editing, key bindings, command history with unlimited size, etc.

In basic terms, Bash is a command line interpreter that typically runs in a text window where user can interpret commands to carry out various actions. The combination of these commands as a series within a file is known as a Shell Script. Bash can read and execute the commands from a Shell Script.Bash is the default login shell for most Linux distributions and Apple's mac OS. It is also accessible for Windows 10 with a version and default user shell in Solaris 11.

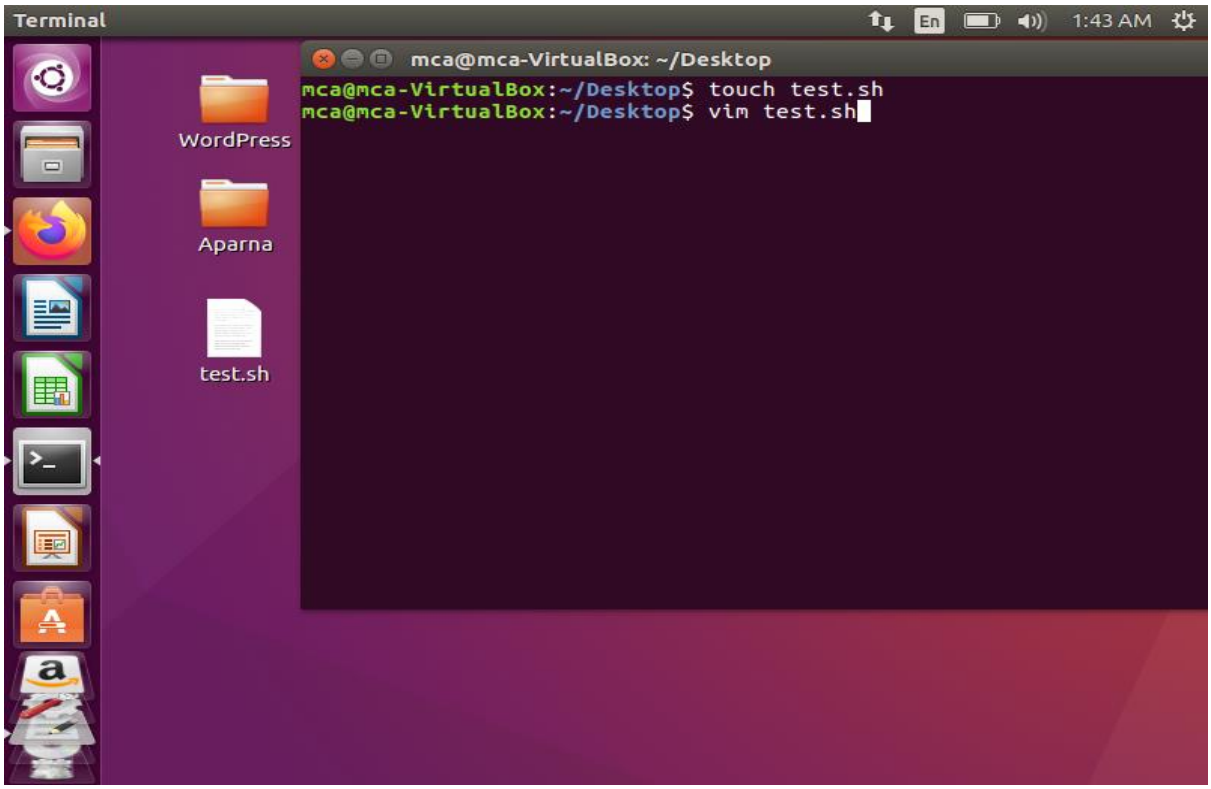
Example Script

In a Bash Shell Script First you need to find out where is your bash interpreter located. Enter the following into your command line:

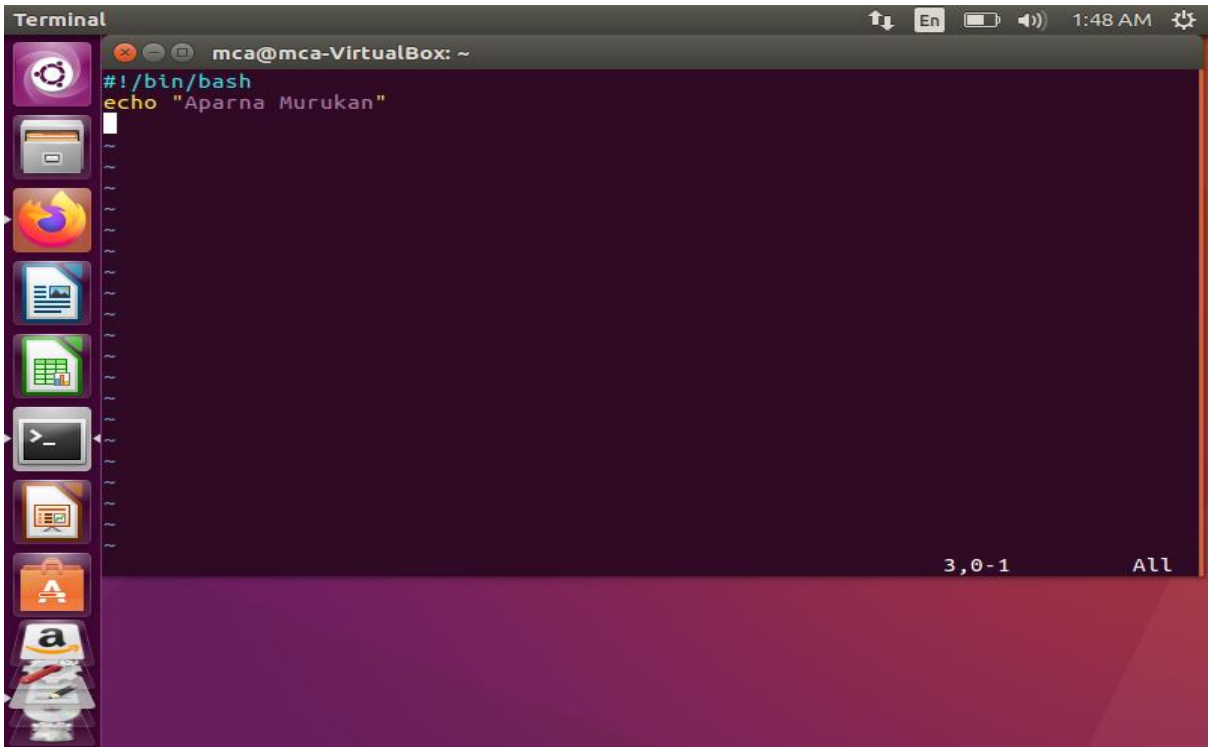
echo $shell



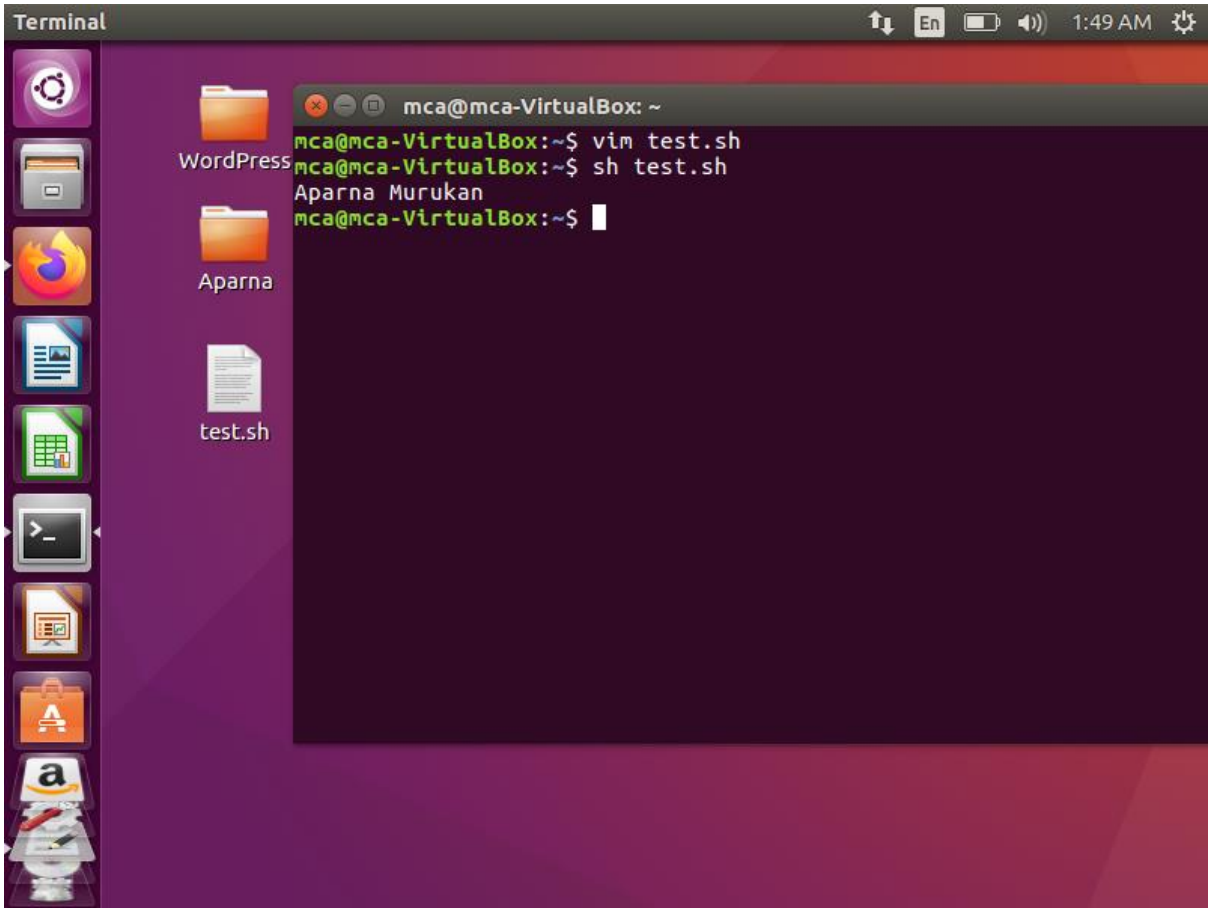
- Open up vim text editor and create file called test.sh



- Insert the following lines to a file:



- Execute the bash script:



Shell Variables

A variable is a character string to which we assign a value. The value assigned could be a number, text, filename, device, or any other type of data.A variable is nothing more than a pointer to the actual data. The shell enables you to create, assign, and delete variables.

Variable Names

The name of a variable can contain only letters (a to z or A to Z), numbers (0 to 9) or the underscore character (_). By convention, Unix shell variables will have their names in UPPERCASE.

The following examples are valid variable names −

_ALI
TOKEN_A
VAR_1
VAR_2

Defining Variables

Variables are defined as follows −

variable_name=variable_value

For example −

NAME="Zara Ali"

The above example defines the variable NAME and assigns the value "Zara Ali" to it. Variables of this type are called scalar variables. A scalar variable can hold only one value at a time.

Accessing Values

To access the value stored in a variable, prefix its name with the dollar sign (**$**) –

For example, the following script will access the value of defined variable NAME and print it on STDOUT −

```
#!/bin/sh
```

```
NAME="Zara Ali"
echo $NAME
```

The above script will produce the following value −

```
Zara Ali
```

Read-only Variables

Shell provides a way to mark variables as read-only by using the read-only command. After a variable is marked read-only, its value cannot be changed.

For example, the following script generates an error while trying to change the value of NAME −

```
#!/bin/sh

NAME="Zara Ali"
readonly NAME
NAME="Qadiri"
```

The above script will generate the following result −

/bin/sh: NAME: This variable is read only.

Variable Types

When a shell is running, three main types of variables are present −

- Local Variables − A local variable is a variable that is present within the current instance of the shell. It is not available to programs that are started by the shell. They are set at the command prompt.
- Environment Variables − An environment variable is available to any child process of the shell. Some programs need environment variables in order to function correctly. Usually, a shell script defines only those environment variables that are needed by the programs that it runs.
- Shell Variables − A shell variable is a special variable that is set by the shell and is required by the shell in order to function correctly. Some of these variables are environment variables whereas others are local variables

Conditional Statements in Shell Script

There are total 5 conditional statements which can be used in bash programming

1. if statement
2. if-else statement
3. if..elif..else..fi statement (Else If ladder)
4. if..then..else..if..then..fi..fi..(Nested if)
5. switch statement

Their description with syntax is as follows:

- if statement : This block will process if specified condition is true.
  Syntax:

  if [expression]

  then

  statement

  fi

- if-else statement : If specified condition is not true in if part then else part will be execute.
  Syntax

  if [expression]

  then

  statement1

  else

  statement2

  fi

- if..elif..else..fi statement (Elif If ladder) : To use multiple conditions in one if-else block, then elif keyword is used in shell. If expression1 is true then it executes statement 1 and 2, and this process continues.
  If none of the condition is true then it processes else part.

  Syntax:

```
if [ expression1 ]
then
   statement1
   statement2
   .
   .
elif [ expression2 ]
then
   statement3
   statement4
   .
   .
else
   statement5
fi
```

- if..then..else..if..then..fi..fi..(Nested if) : Nested if-else block can be used when, one condition is satisfies then it again checks another condition. In the syntax, if expression1 is false then it processes else part, and again expression2 will be check.
  Syntax:

```
if [ expression1 ]
then
   statement1
   statement2
   .
else
   if [ expression2 ]
   then
      statement3
      .
   fi
fi
```

- switch statement : Case statement works as a switch statement if specified value match with the pattern then it will execute a block of that particular pattern. When a match is found all of the associated statements until the double semicolon (;;) is executed. A case will be terminated when the last command is executed. If there is no match, the exit status of the case is zero.

Syntax:

```
case  in
  Pattern 1) Statement 1;;
  Pattern n) Statement n;;
esac
```

Looping Statements in Shell Script

There are total 3 looping statements which can be used in bash programming

1. while statement
2. for statement
3. until statement

To alter the flow of loop statements, two commands are used they are,

1. break
2. continue

Their descriptions and syntax are as follows:

- while statement : Here command is evaluated and based on the result loop will executed, if command raise to false then loop will be terminated
  Syntax

```
while command
do
  Statement to be executed
done
```

- for statement : The for loop operate on lists of items. It repeats a set of commands for every item in a list.Here var is the name of a variable and word1 to wordN are sequences of characters separated by spaces (words). Each time the for loop executes, the value of the variable var is set to the next word in the list of words, word1 to wordN.
  Syntax

```
for var in word1 word2 ...wordn
do
  Statement to be executed
done
```

- until statement: The until loop is executed as many as times the condition/command evaluates to false.
  The loop terminates when the condition/command becomes true.

  Syntax

  ```
  until command
  do
    Statement to be executed until command is true
  Done
  ```

Shell functions

Shell functions are similar to subroutines, procedures, and functions in other programming languages.

Creating Functions

To declare a function, simply use the following syntax −

```
function_name () {

  list of commands

}
```

Example

Following example shows the use of function −

```
#!/bin/sh

# Define your function here
Hello () {
  echo "Hello World"
}

# Invoke your function
Hello
```

Upon execution, you will receive the following output −

```
$./test.sh
Hello World
```

Pass Parameters to a Function

You can define a function that will accept parameters while calling the function. These parameters would be represented by **$1**, **$2** and so on.

Following is an example where we pass two parameters *Zara* and *Ali* and then we capture and print these parameters in the function.

```
#!/bin/sh

# Define your function here
Hello () {
  echo "Hello World $1 $2"
}

# Invoke your function
Hello Zara Ali
```

Upon execution, you will receive the following result −

```
$./test.sh

Hello World Zara Ali
```

Returning Values from Functions

If you execute an **exit** command from inside a function, its effect is not only to terminate execution of the function but also of the shell program that called the function.If you instead want to just terminate execution of the function, then there is way to come out of a defined function.

Based on the situation you can return any value from your function using the return command whose syntax is as follows −

```
return code
```

Here code can be anything you choose here, but obviously you should choose something that is meaningful or useful in the context of your script as a whole.

Example

Following function returns a value 10 −

```
#!/bin/sh

# Define your function here
Hello () {
  echo "Hello World $1 $2"
  return 10
}
```

```
# Invoke your function
Hello Zara Ali

# Capture value returnd by last command
ret=$?

echo "Return value is $ret"
```

Upon execution, you will receive the following result −

```
$./test.sh

Hello World Zara Ali

Return value is 10
```

Nested Functions

One of the more interesting features of functions is that they can call themselves and also other functions. A function that calls itself is known as a *recursive function*.

Following example demonstrates nesting of two functions −

```
#!/bin/sh

# Calling one function from another
number_one () {
  echo "This is the first function speaking..."
  number_two
}

number_two () {
  echo "This is now the second function speaking..."
}

# Calling function one.
number_one
```

Upon execution, you will receive the following result −

```
This is the first function speaking...

This is now the second function speaking...
```

Function Call from Prompt

You can put definitions for commonly used functions inside your **.***profile*. These definitions will be available whenever you log in and you can use them at the command prompt.Alternatively, you can group the definitions in a file, say *test.sh*, and then execute the file in the current shell by typing −

```
$. test.sh
```

This has the effect of causing functions defined inside *test.sh* to be read and defined to the current shell as follows −

$ number_one

This is the first function speaking...

This is now the second function speaking...

$

To remove the definition of a function from the shell, use the unset command with the .f option. This command is also used to remove the definition of a variable to the shell.

$ unset -f function_name

Experiment No.5

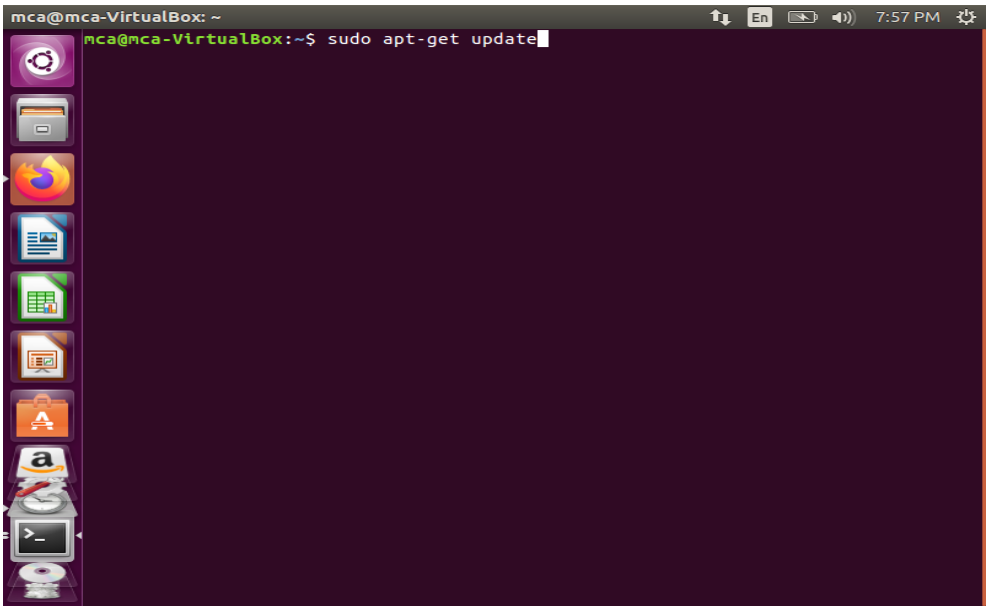
Aim

Installation and configuration of LAMP stack. Deploy an open source application such as phpmyadmin and WordPress.

Procedure

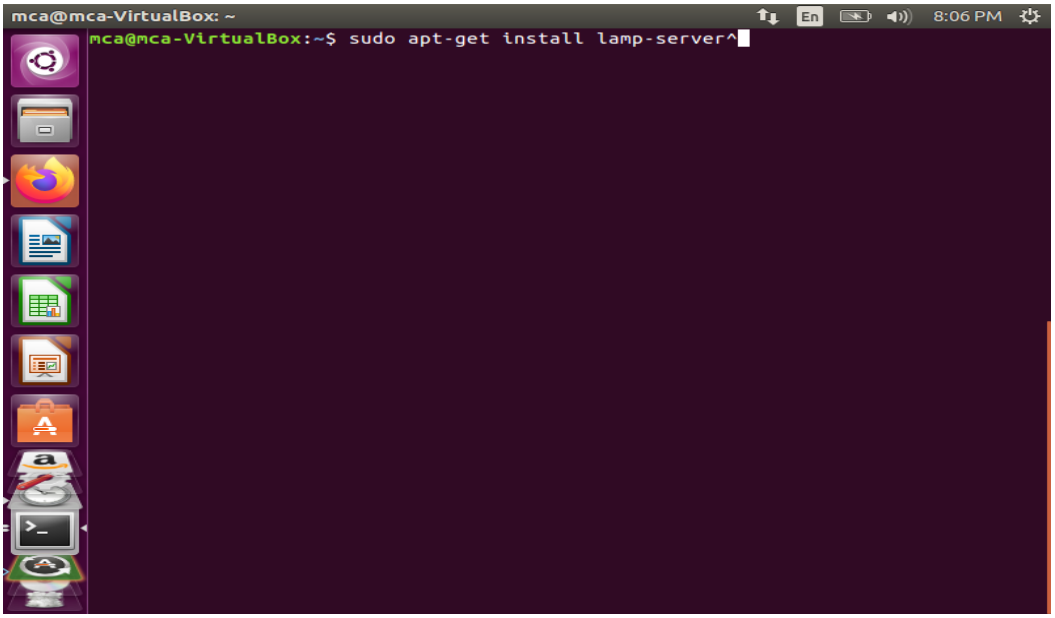
Step 1 : First refresh your package index

$ sudo apt-get update

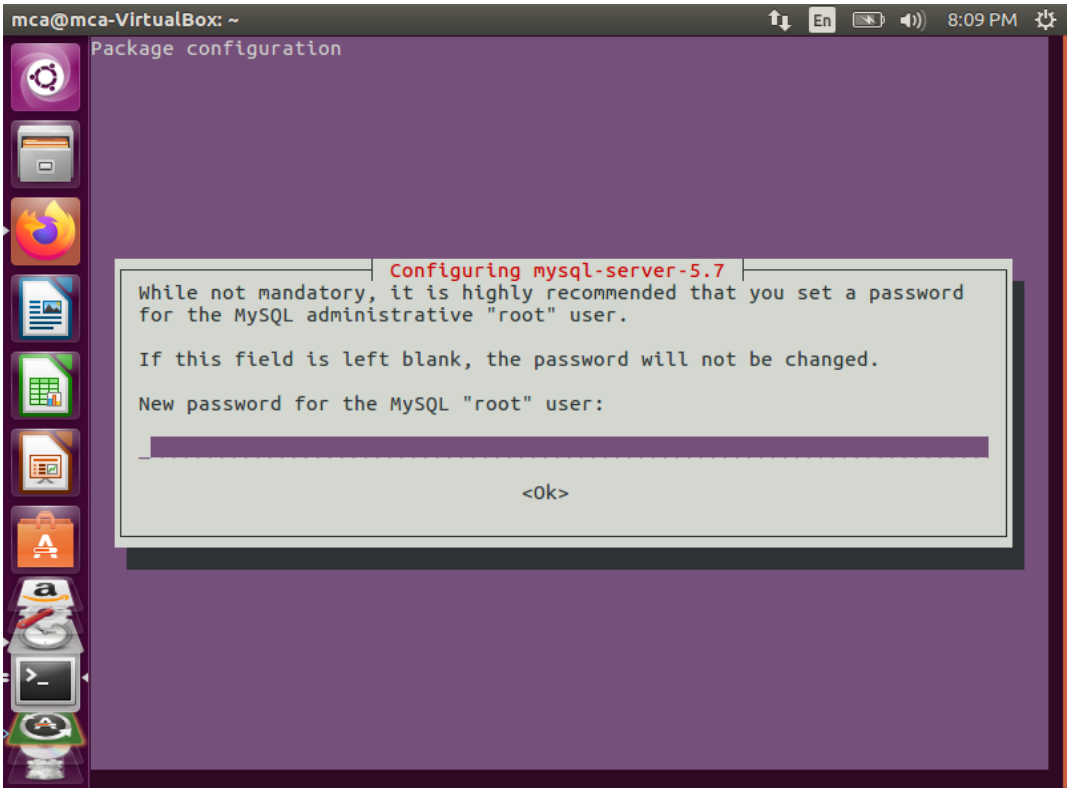


Step 2 : Install the LAMP stack

*$ sudo apt-get install lamp-server^*

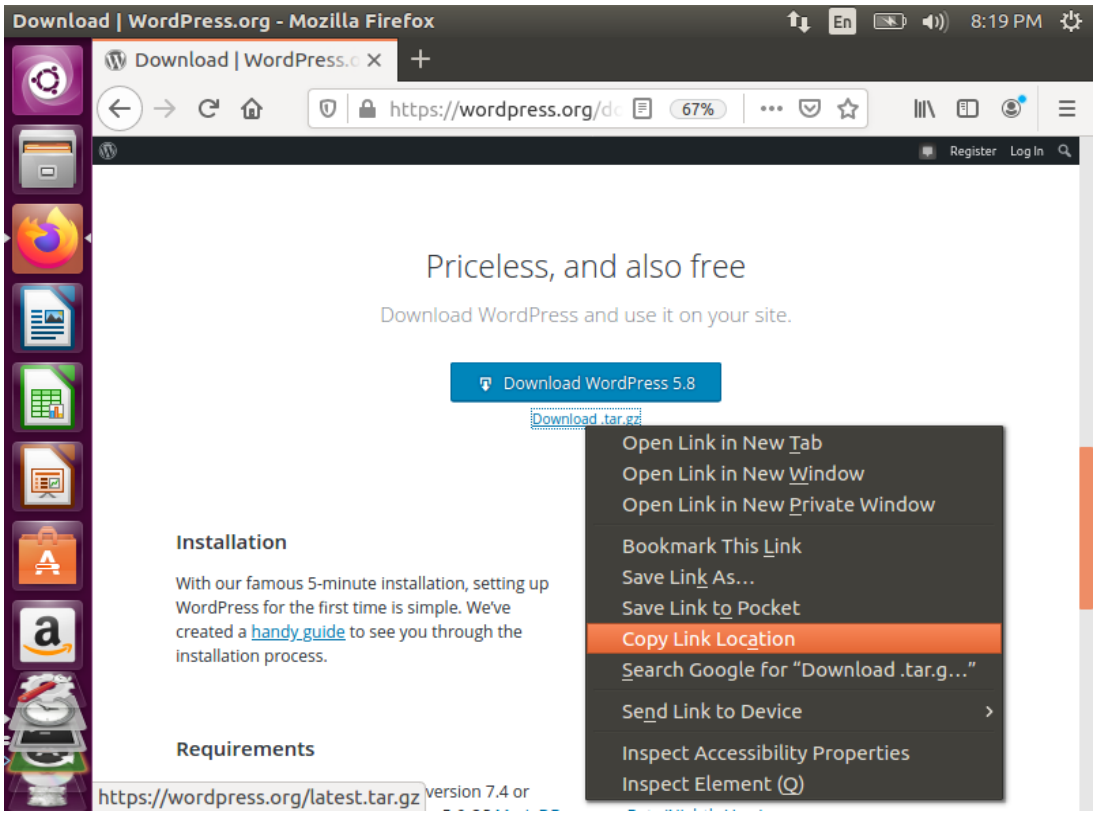


- Set the password for MySQL root user



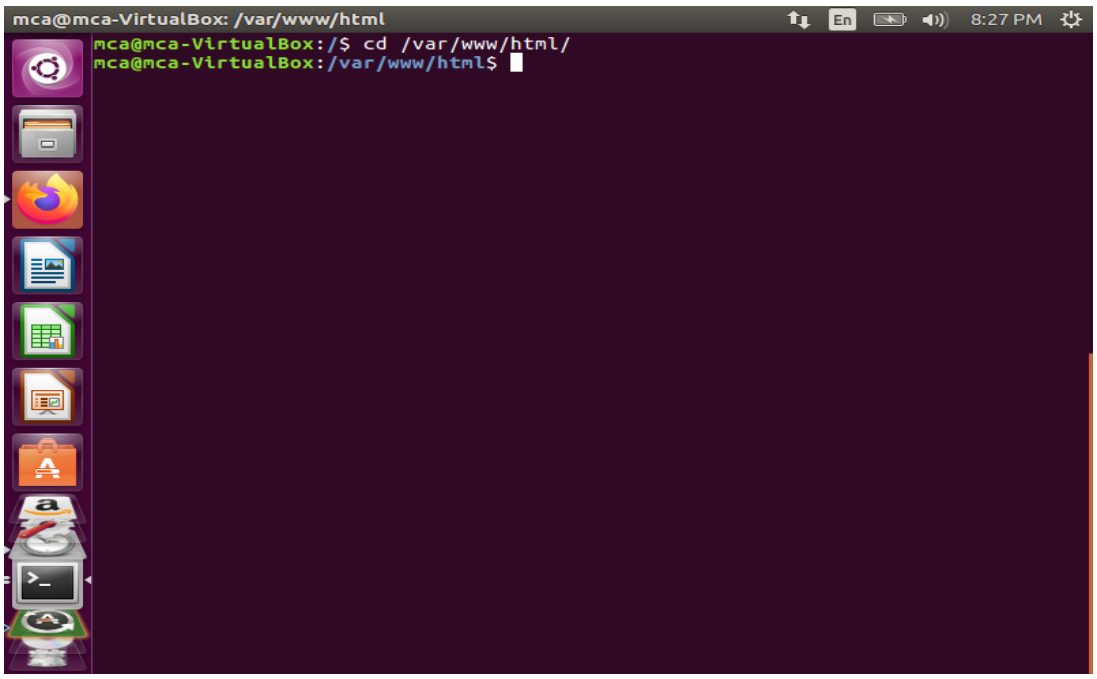
Step 3 : Copy the link address for WordPress from the link

https://wordpress.org/download/



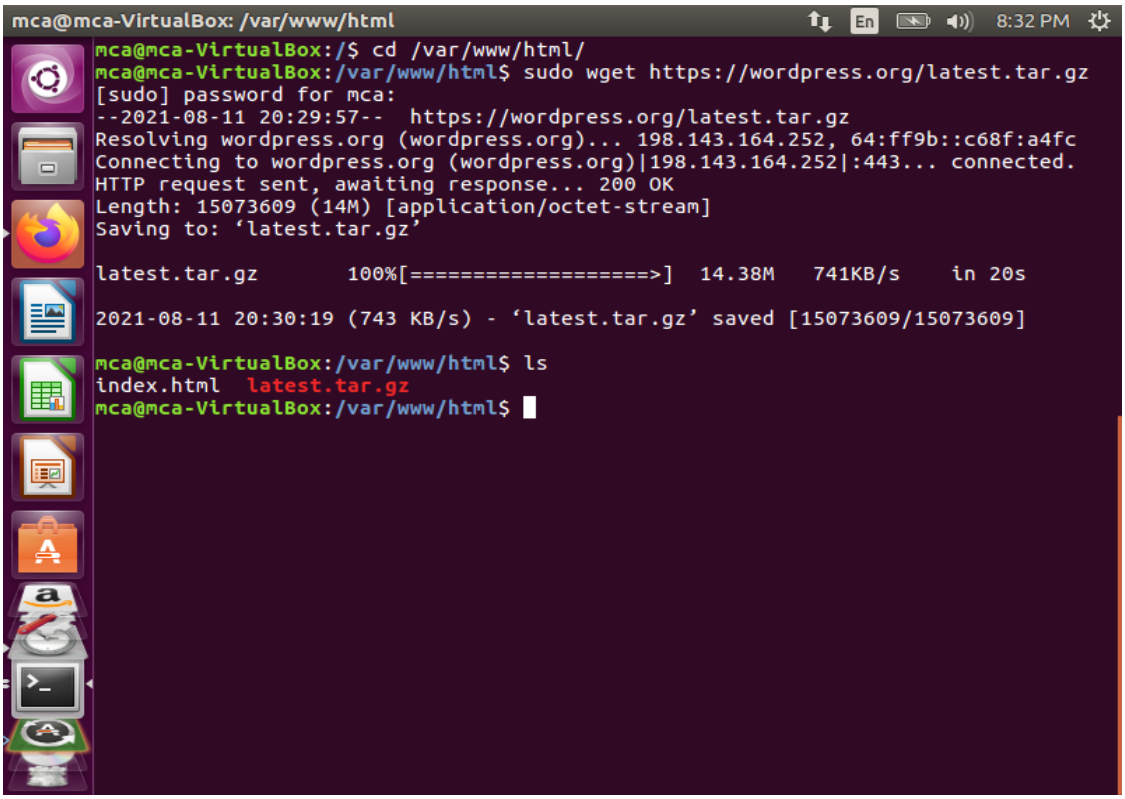
Step 4 : Change the directory to root directory of Apache

$ cd /var/www/html/



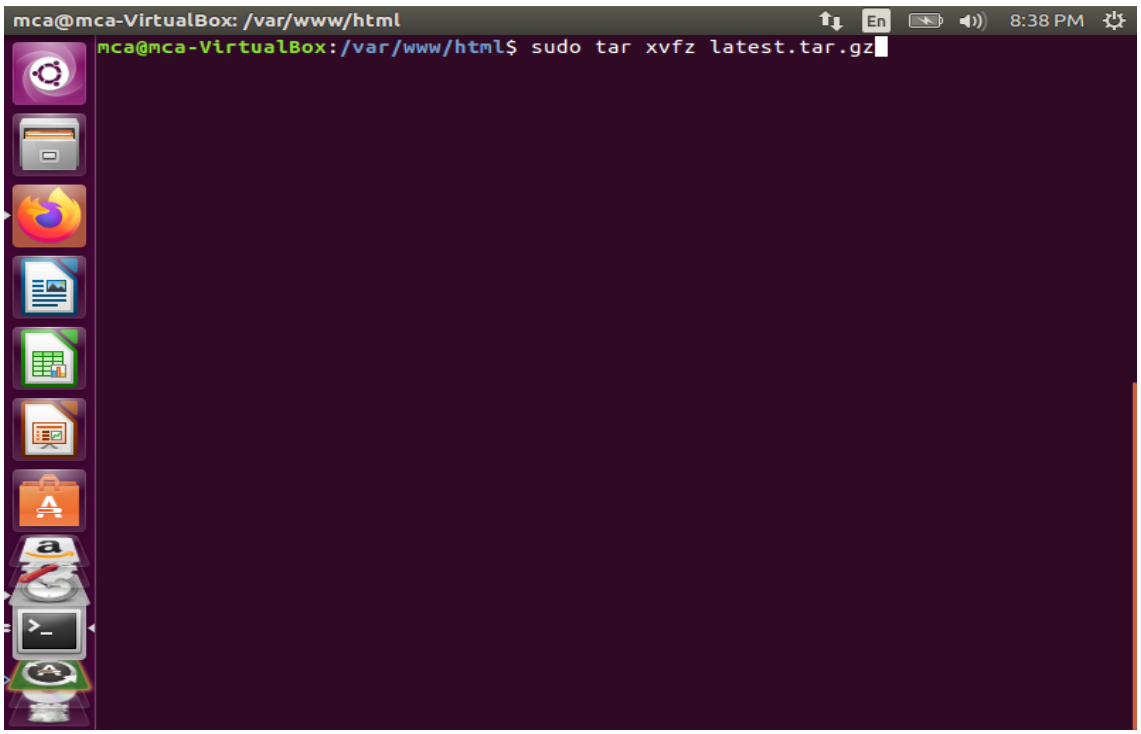
Step 5 : Download WordPress

$ sudo wget https://wordpress.org/latest.tar.gz

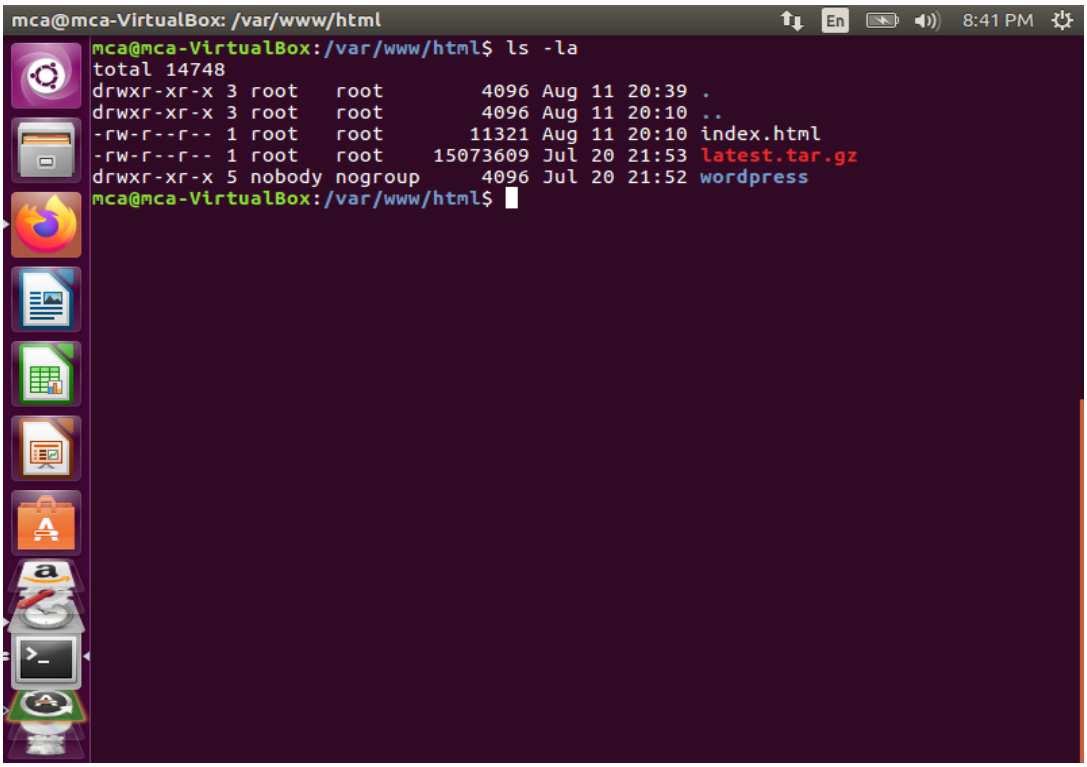


Step 6 : Unzip latest.tar.gz file

$ sudo tar xvfz latest.tar.gz



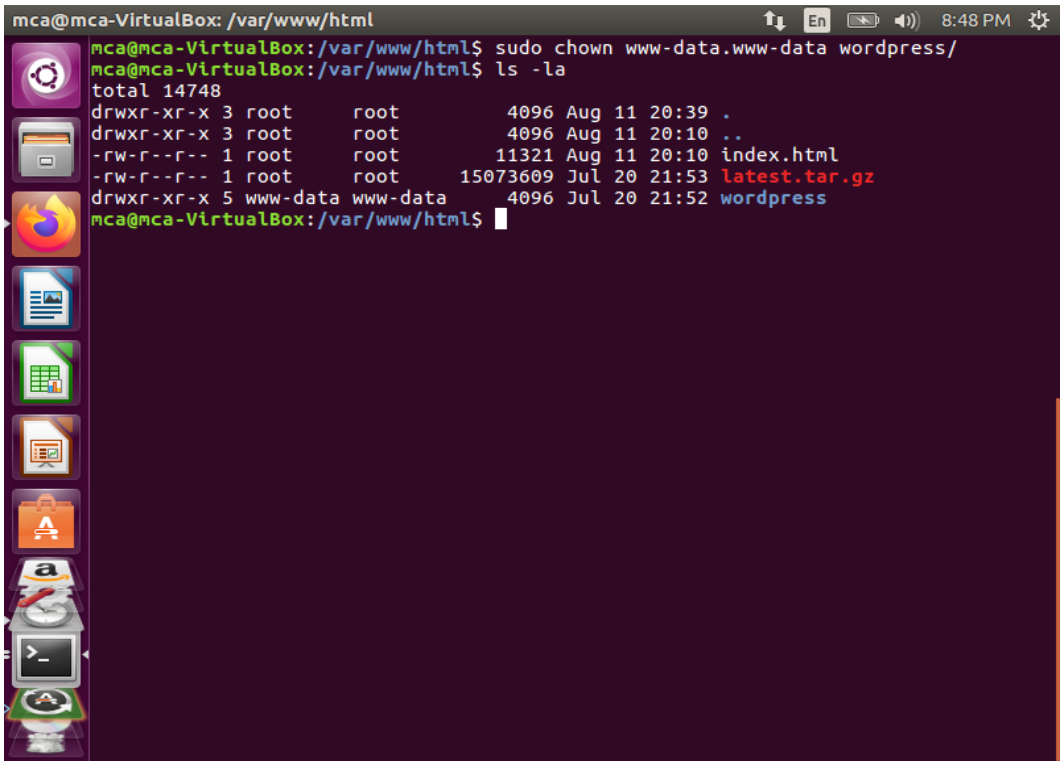
Step 7 : Check the rights on the current directory files *$ ls –la*



Step 8 : Change the owner of the WordPress file

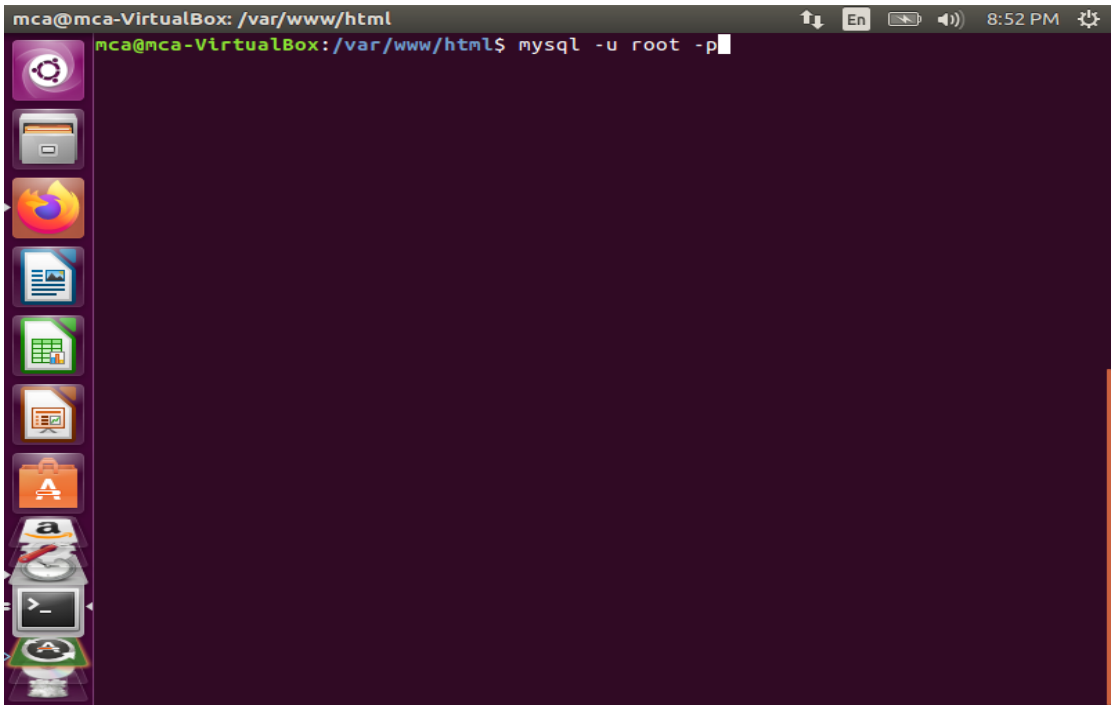
$ sudo chown www-data.www-data wordpress/

$ ls –la

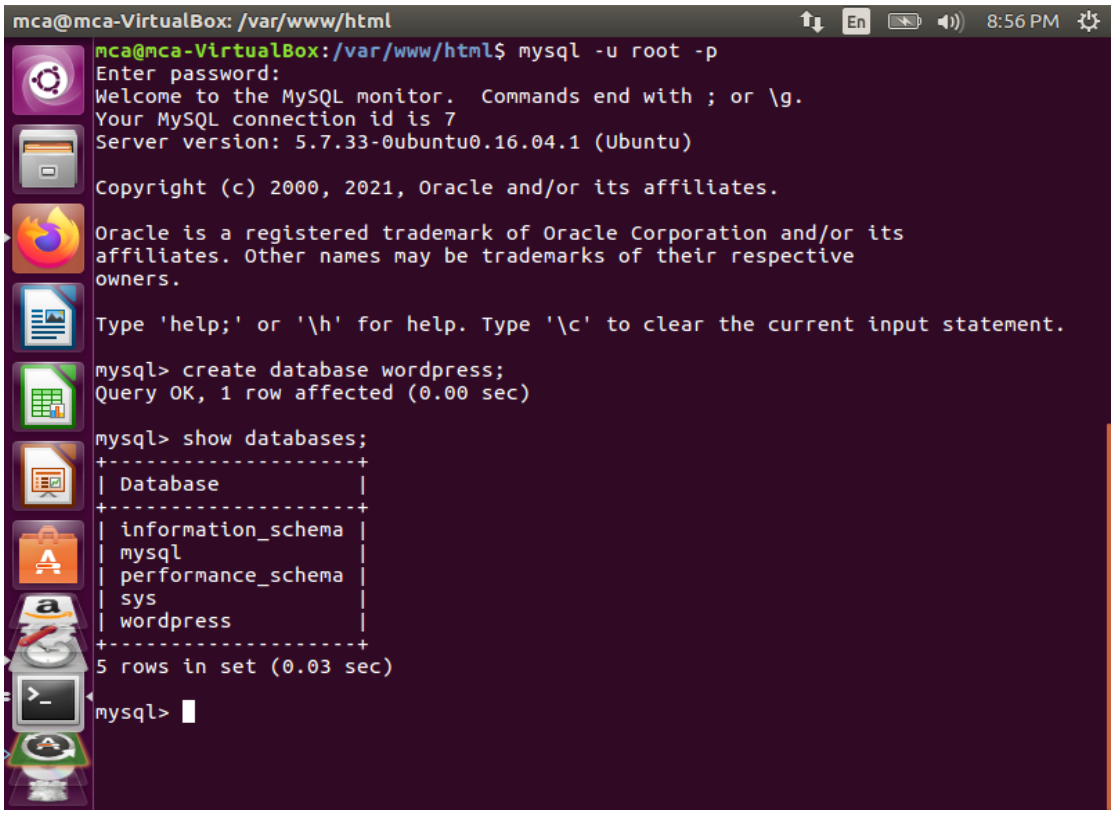


Step 9 : Open MySQL

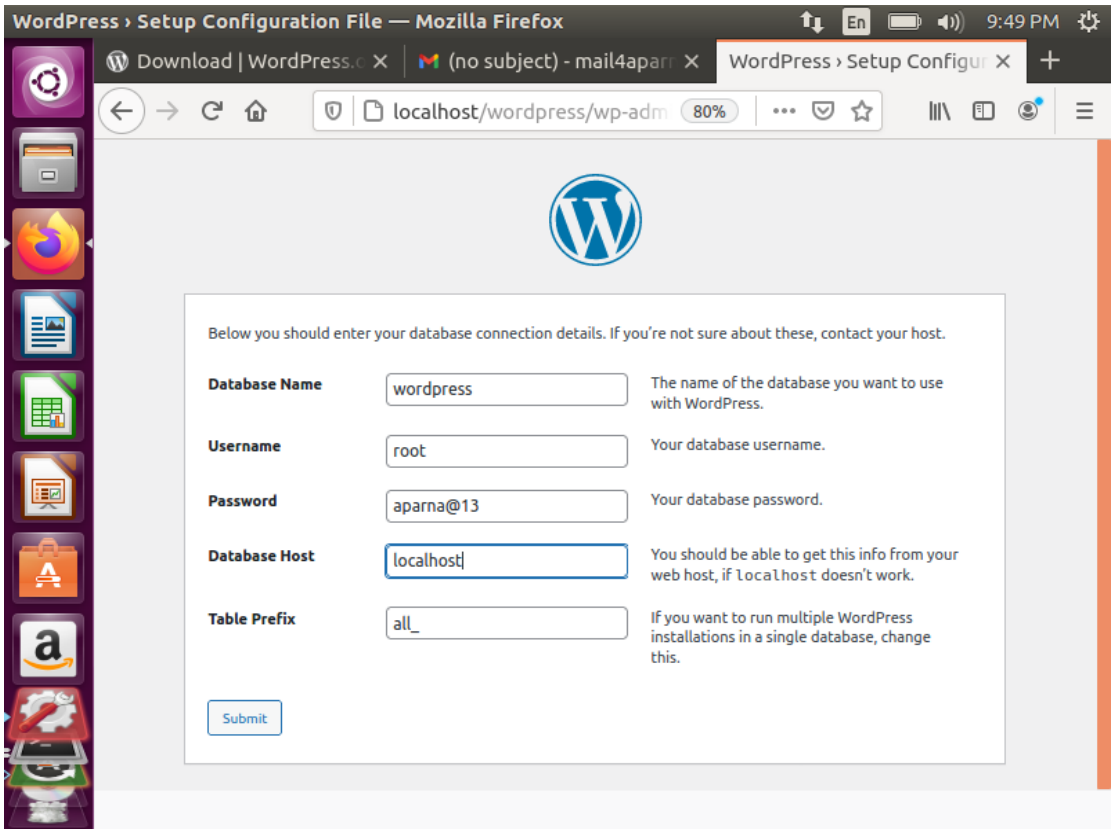
$ mysql -u root –p



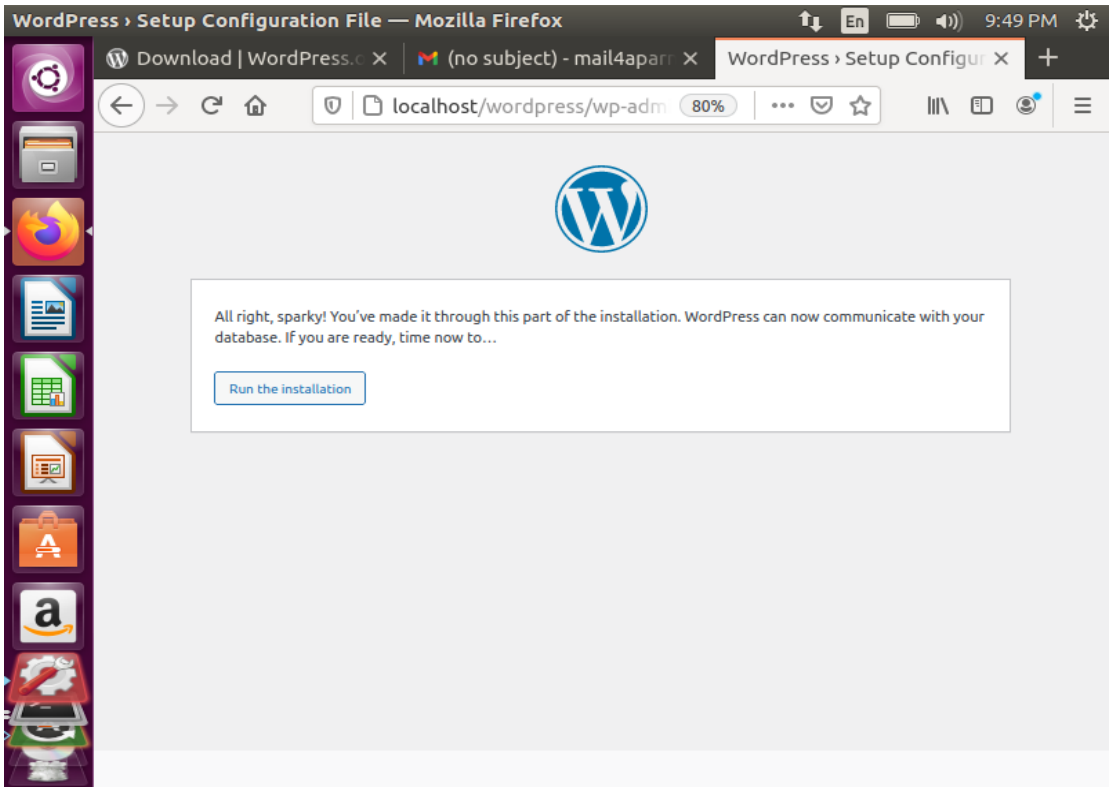
Step 10 : Create a database named wordpress



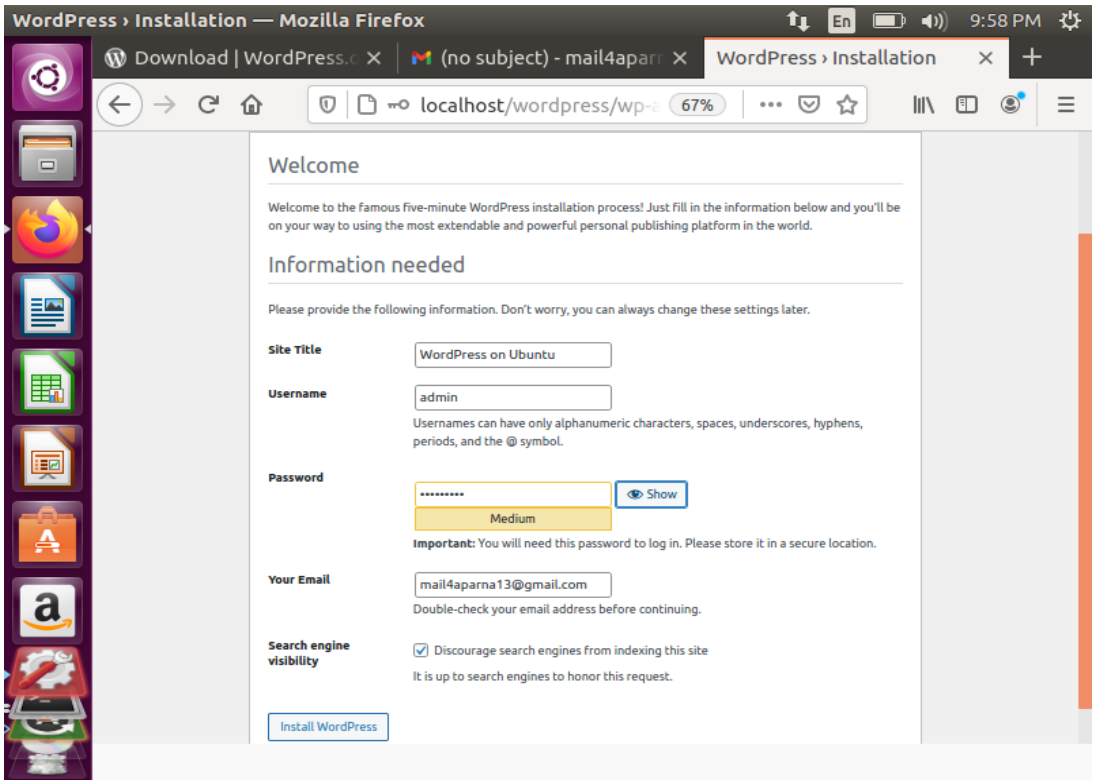
Step 11 : Open WordPress configuration in browser (localhost/wordpress/) ,then enter the database connection details and click submit.

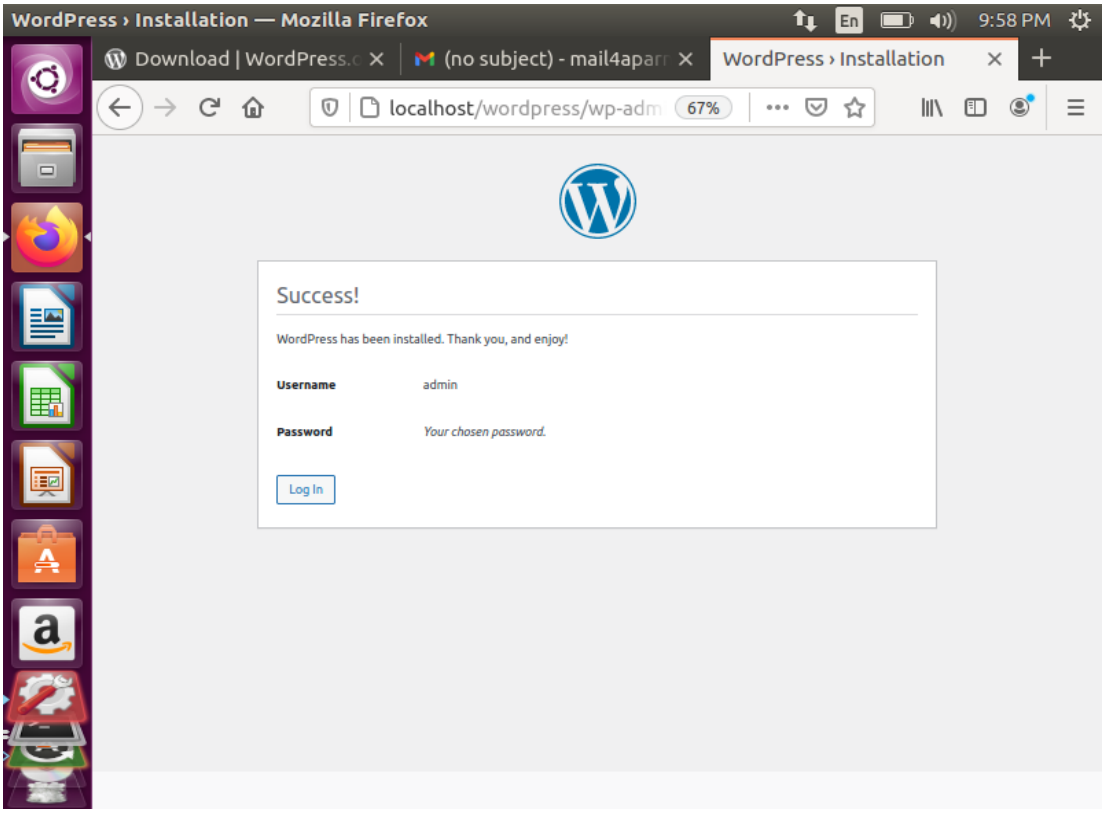


- Click Run the installation



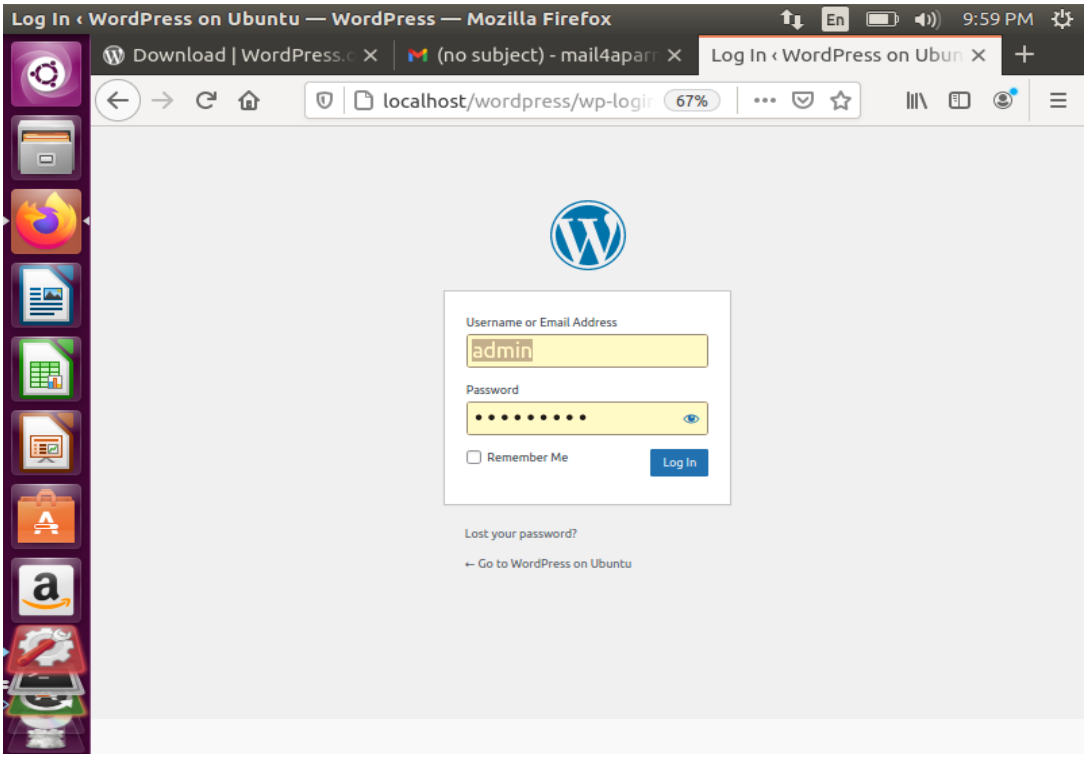
Step 12 : Give informations required and click install WordPress



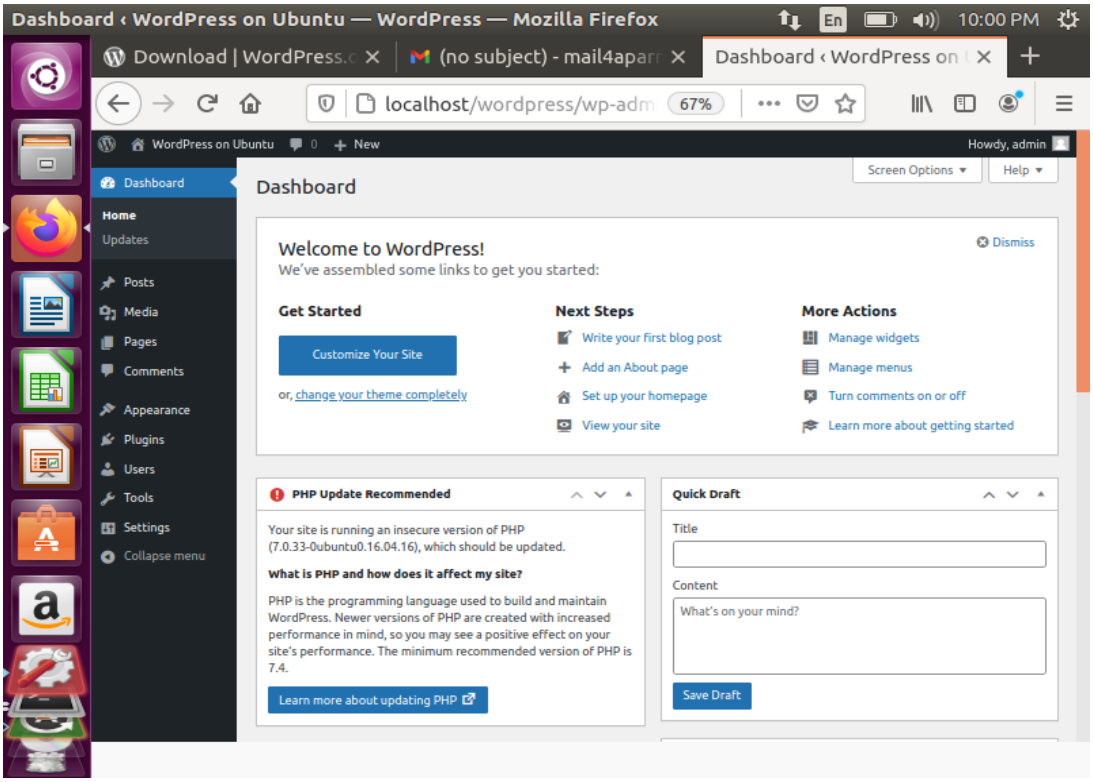


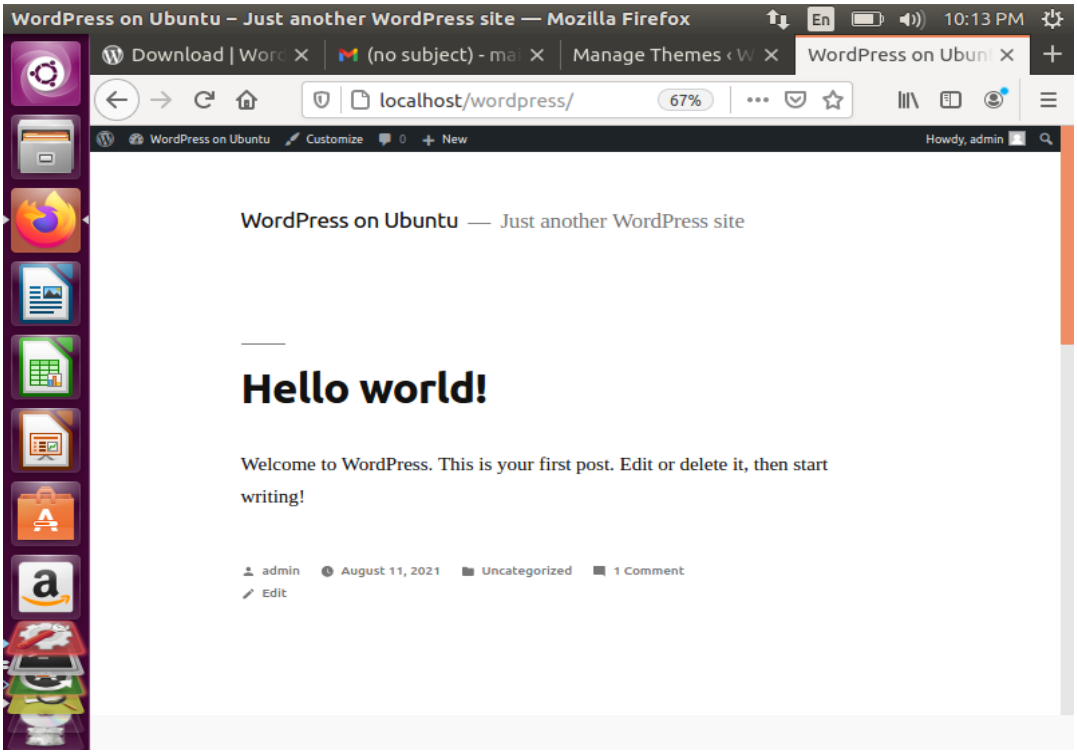
- Click Log In

Step 13 : Enter username and password and click log in



- Successfully enter into WordPress dashboard





- WordPress is installed successfully.

Experiment No.6

Aim

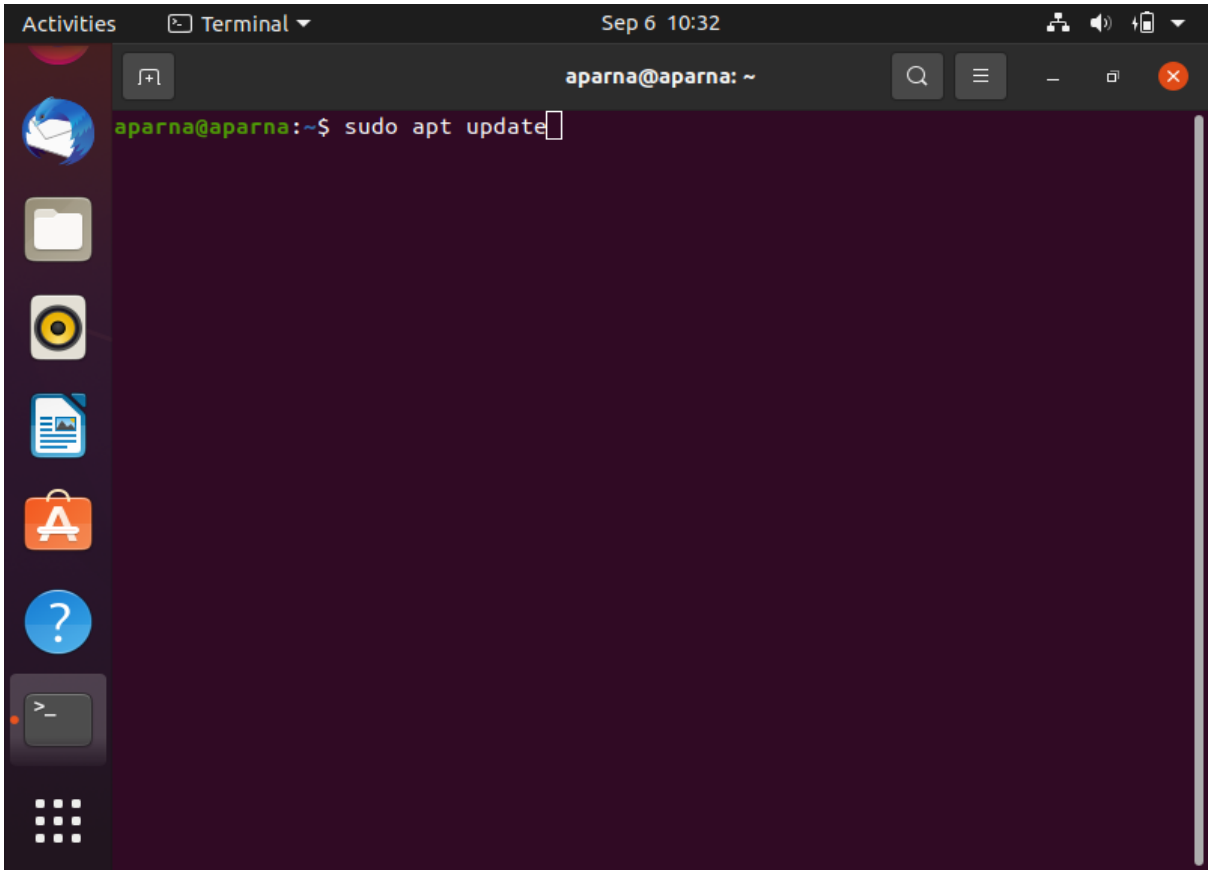
Installation and configuration of common software frame works such as Laravel.

Procedure

Step 1 – Install Apache Web Server

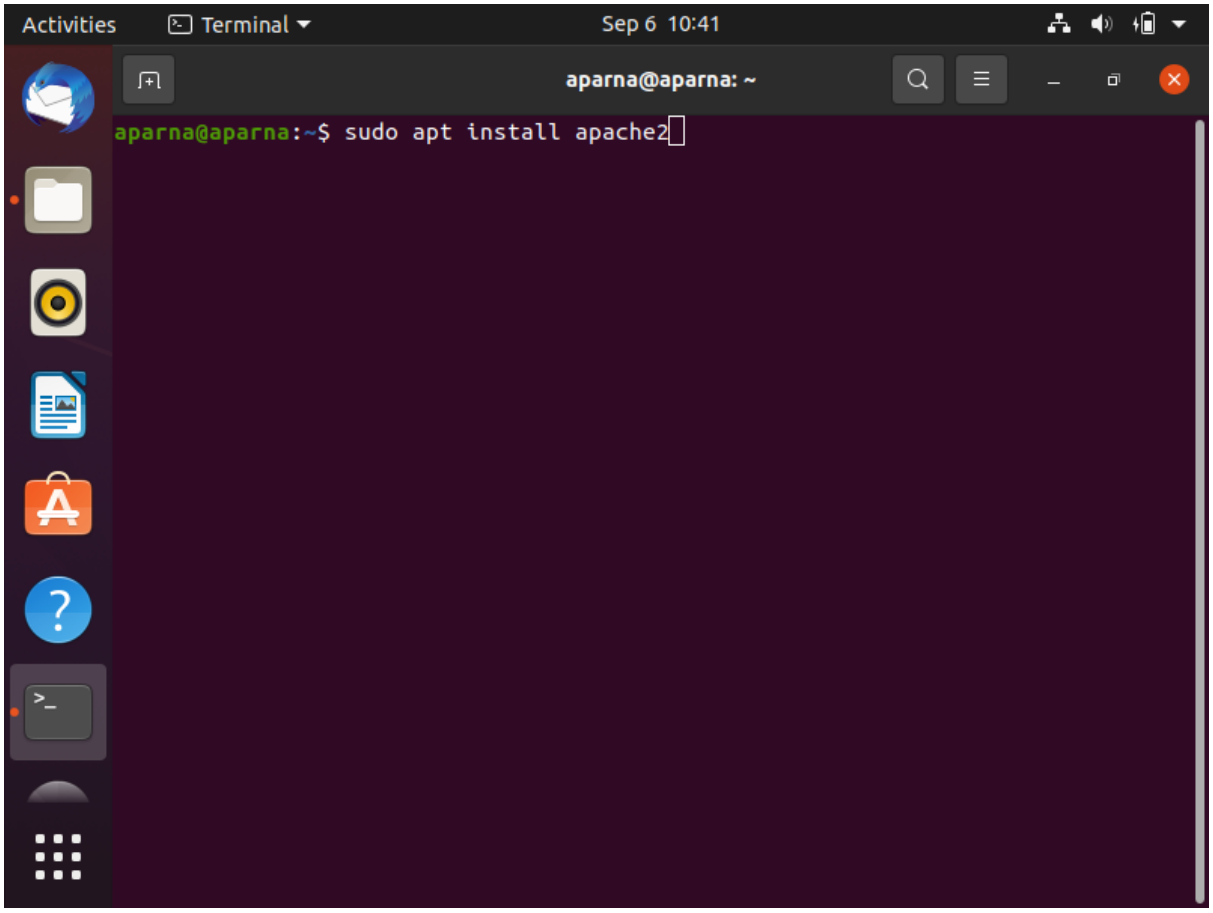
- Let's open up a Terminal and do first thing first update your package list using Sudo apt update command.

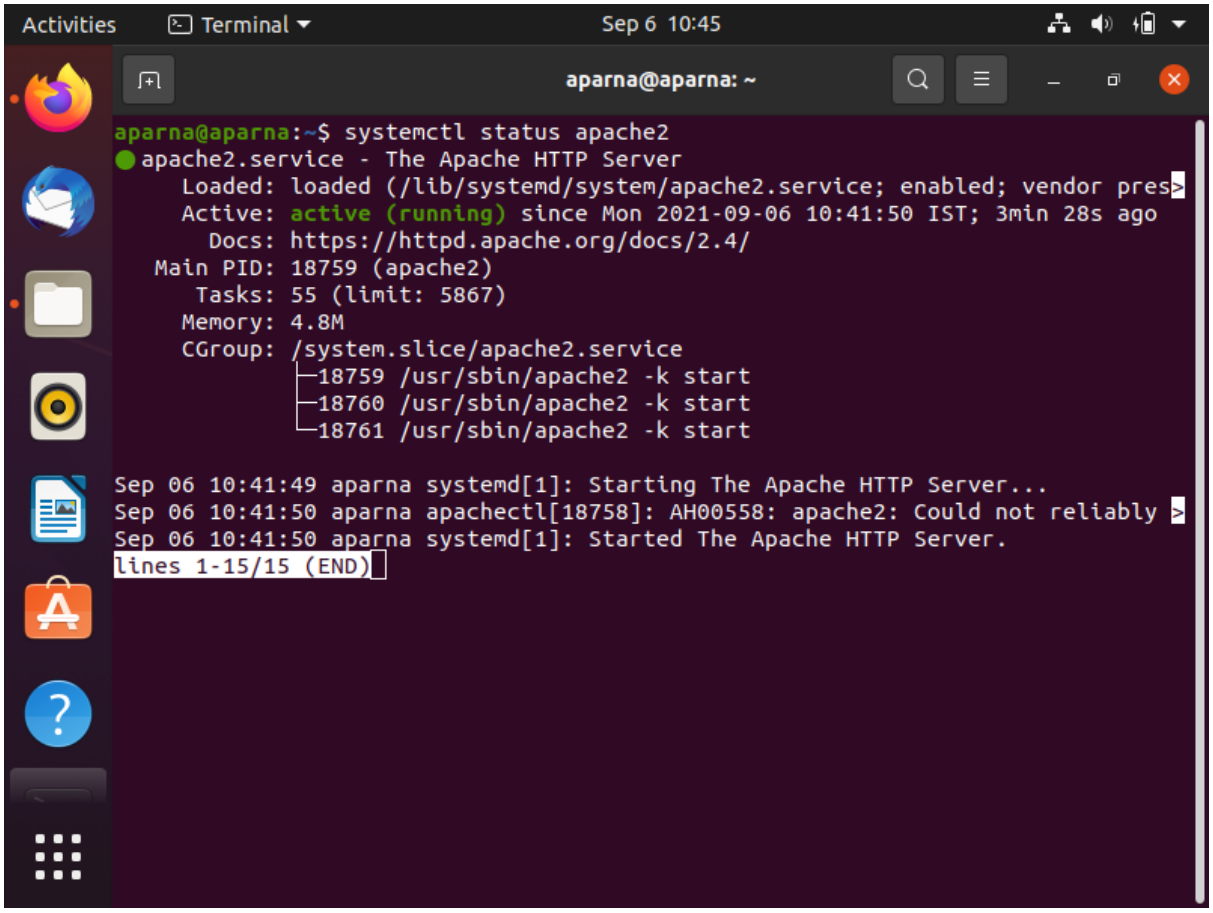
```
sudo apt update
```



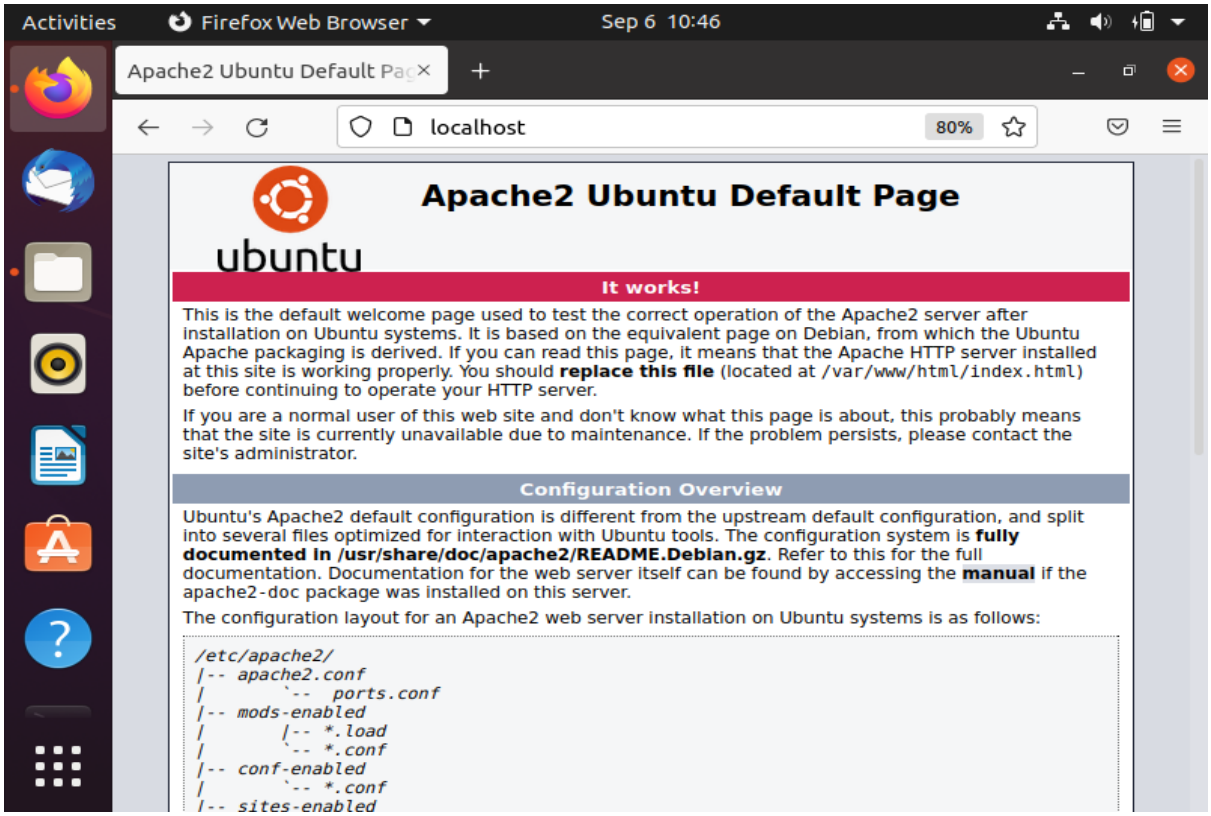
- After updating your package list install apache webserver. So, go ahead and type sudo apt install apache2 then hit the enter key. Press y key to proceed.

```
sudo apt install apache2
systemctl status apache2
```


Activities
Terminal
Sep 6 10:41
aparna@aparna: ~
aparna@aparna:~$ sudo apt install apache2


Activities
Terminal
Sep 6 10:45
aparna@aparna: ~
aparna@aparna:~$ systemctl status apache2
apache2.service - The Apache HTTP Server
Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor pres
Active: active (running) since Mon 2021-09-06 10:41:50 IST; 3min 28s ago
Docs: https://httpd.apache.org/docs/2.4/
Main PID: 18759 (apache2)
Tasks: 55 (limit: 5867)
Memory: 4.8M
CGroup: /system.slice/apache2.service
├─18759 /usr/sbin/apache2 -k start
├─18760 /usr/sbin/apache2 -k start
└─18761 /usr/sbin/apache2 -k start
Sep 06 10:41:49 aparna systemd[1]: Starting The Apache HTTP Server...
Sep 06 10:41:50 aparna apachectl[18758]: AH00558: apache2: Could not reliably
Sep 06 10:41:50 aparna systemd[1]: Started The Apache HTTP Server.
lines 1-15/15 (END)

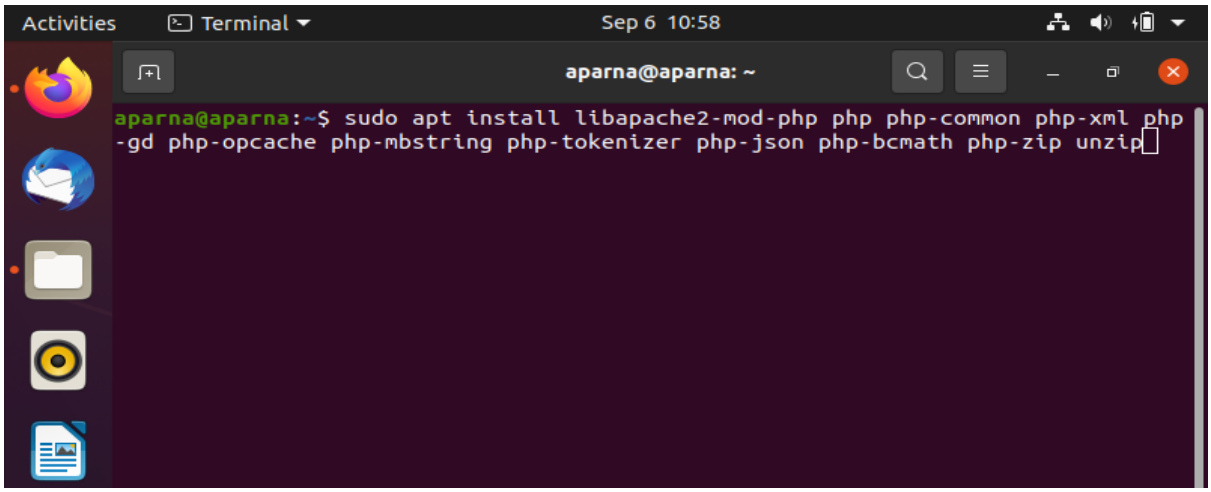
- Now open up the web browser and type localhost to see the default apache webpage is serving or not

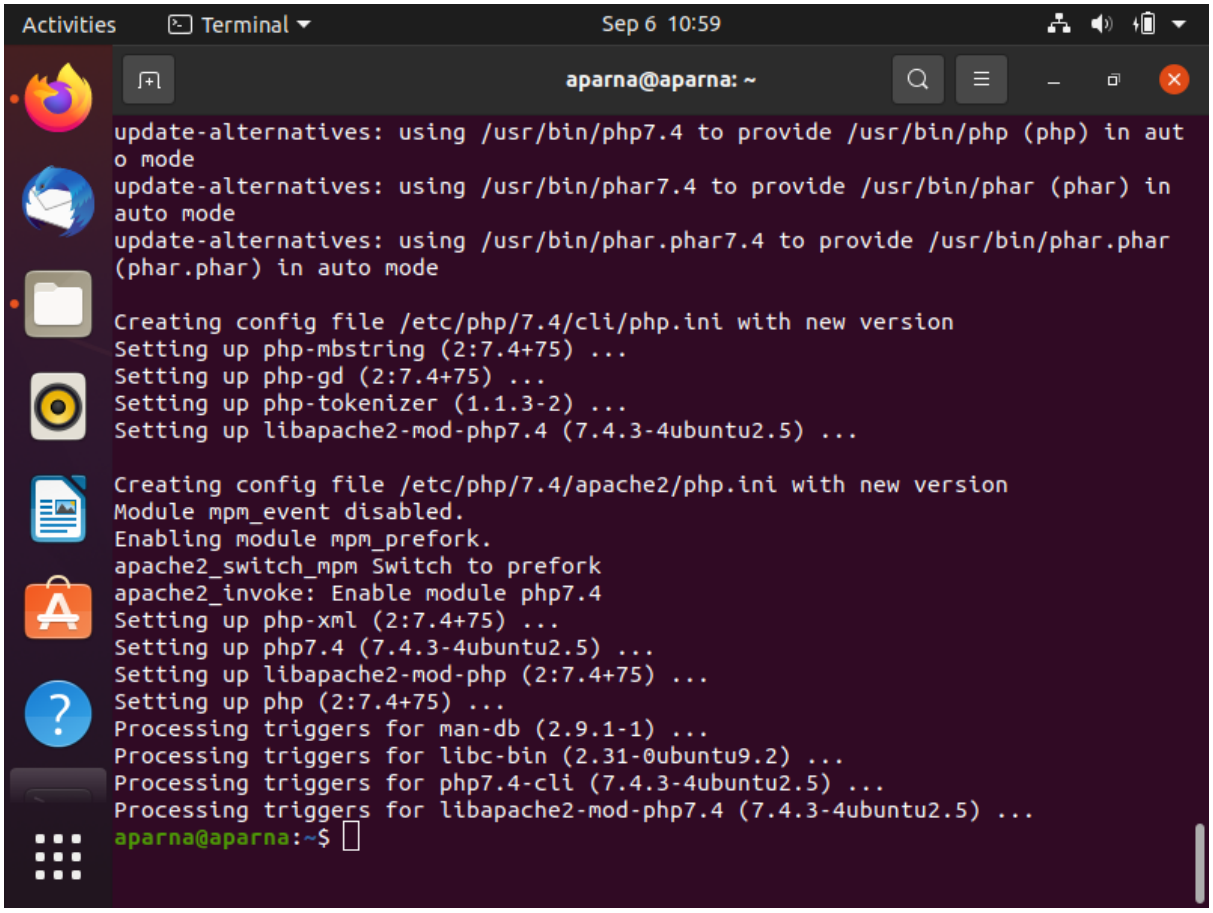


Step 2 – Install and Configure PHP 7.4

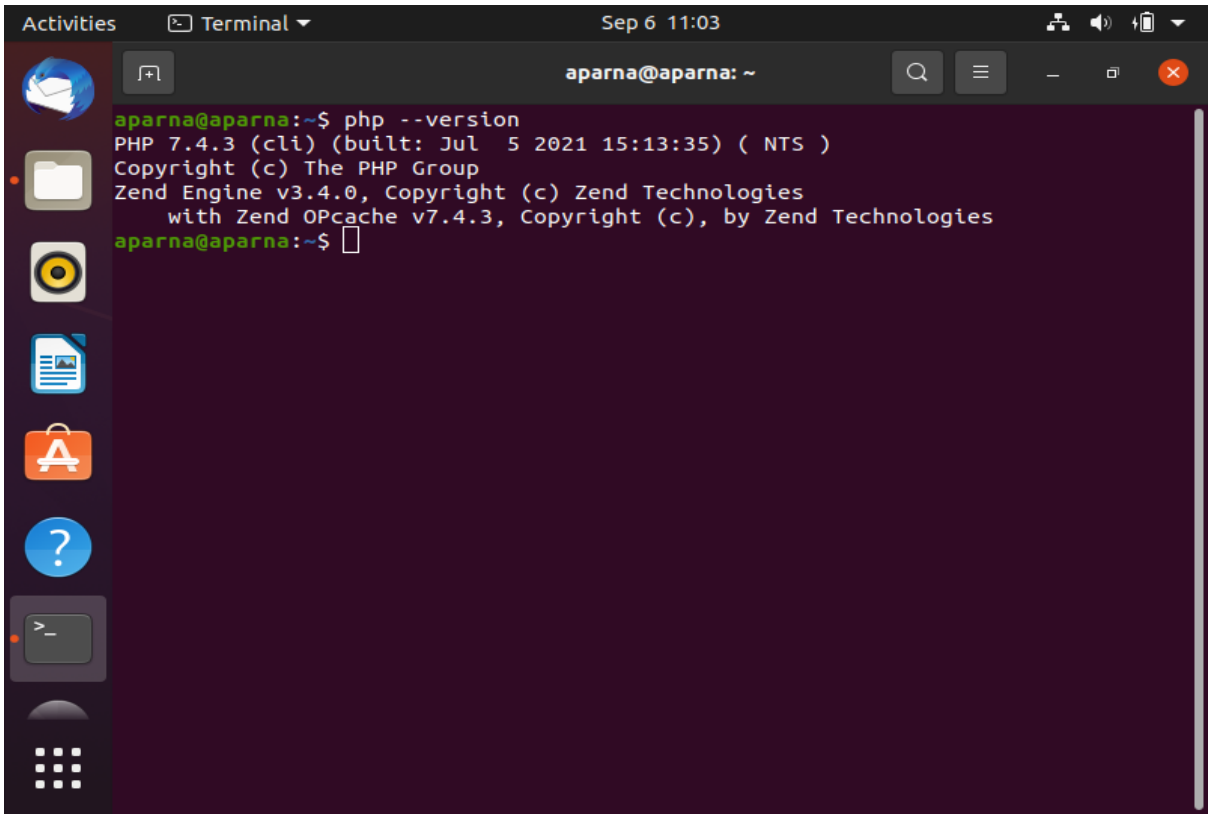
- Get back to the terminal and it's time to install PHP. To install Laravel 8.x, at least you must have PHP >= 7.3 on your system. And by default, the official Ubuntu 20.04 repository provides PHP 7.4 packages. Install PHP 7.4 packages using the apt command below

```
sudo apt install libapache2-mod-php php php-common php-xml php-gd php-opcache php-mbstring php-tokenizer php-json php-bcmath php-zip unzip
```





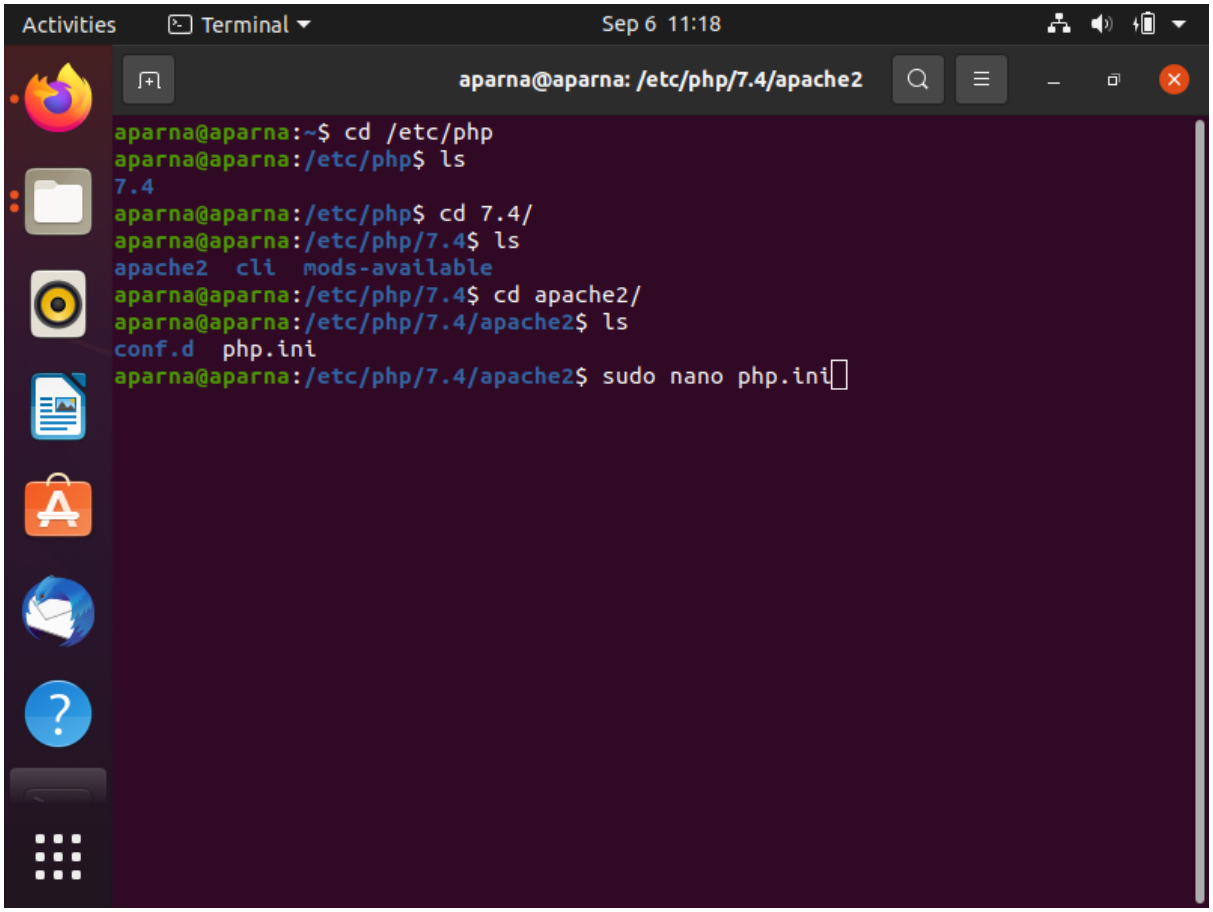
- You can check your PHP version using it.
  php – version



- Now go ahead and make tweak changes in PHP ini file and set cgi.fix_pathinfo set to be 0. If this number is kept as a 1, the php interpreter will do its best to process the file that is as near to the requested file as possible. This is a possible security risk. If this number is set to 0, conversely, the interpreter will only process the exact file path—a much safer alternative.

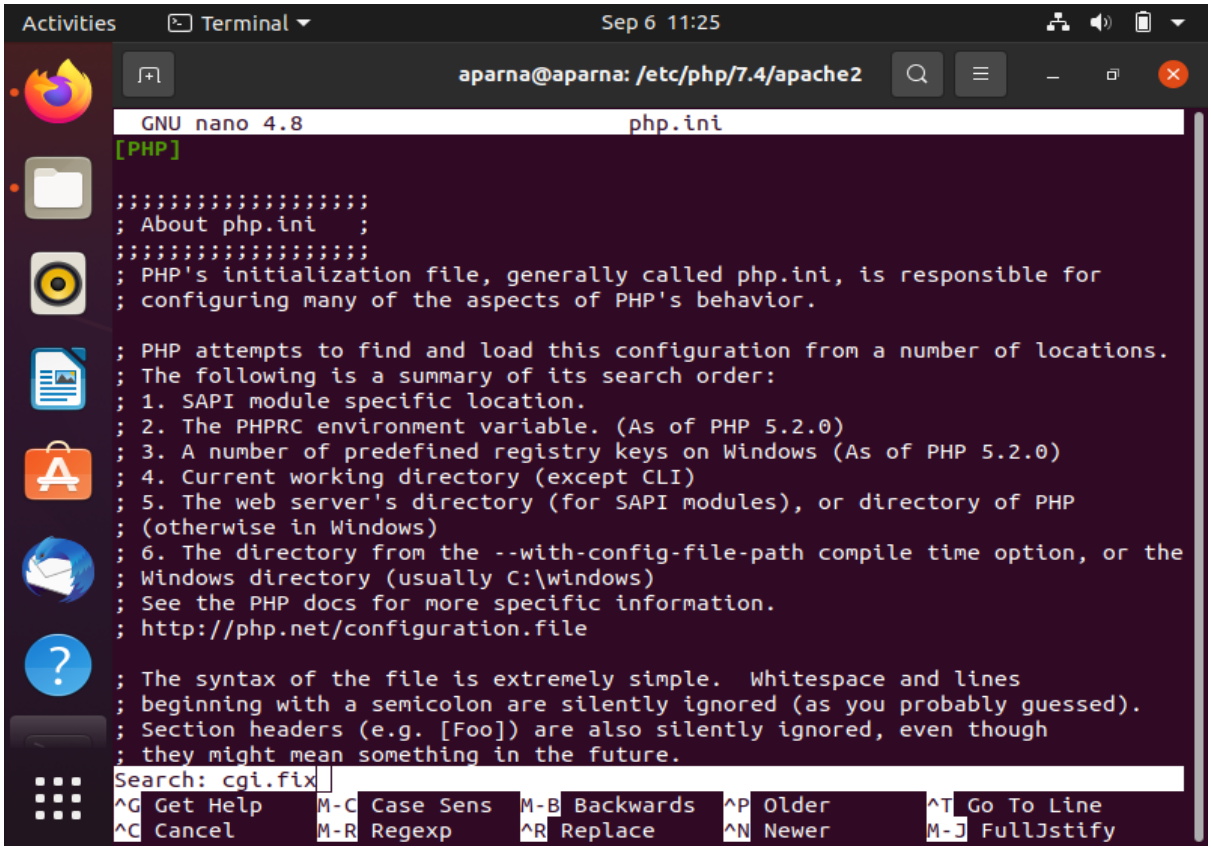
```
cd /etc/php/7.4/apache2
```

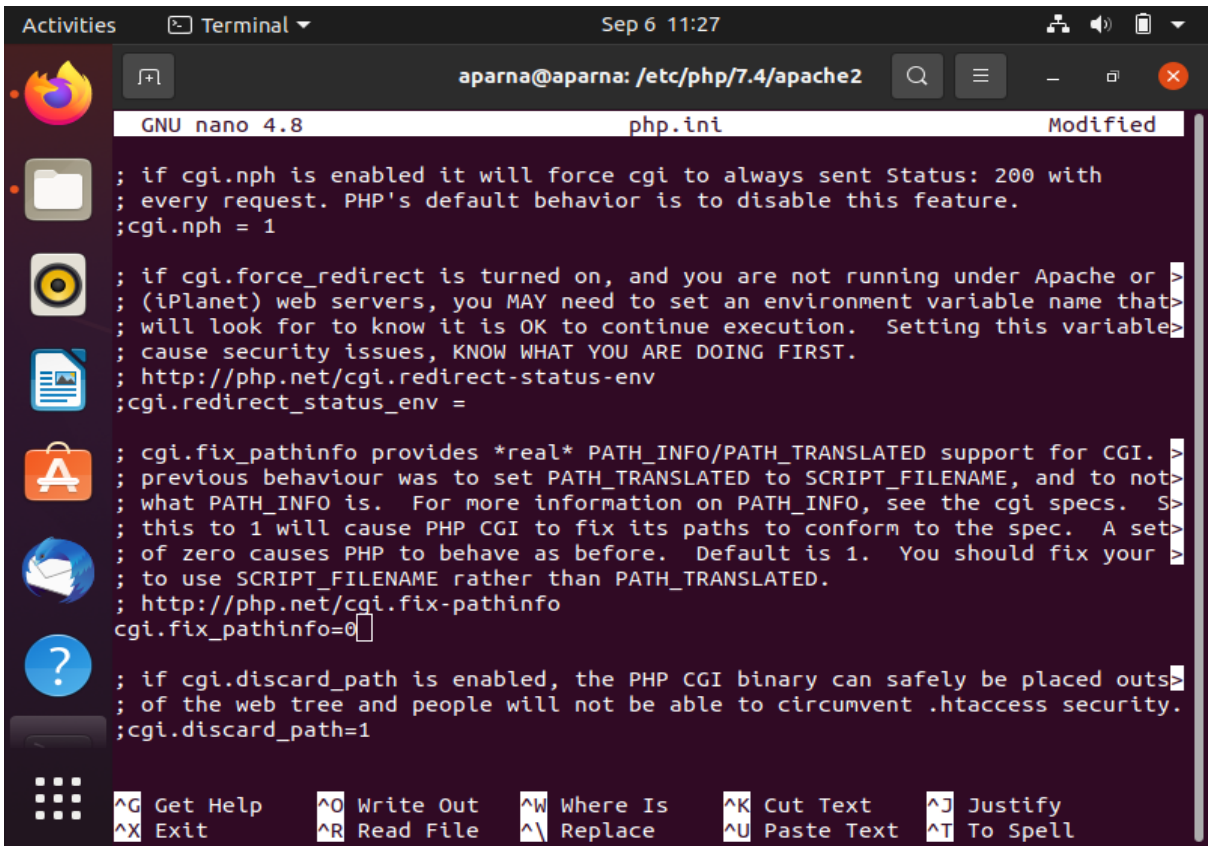
```
sudo nano php.ini
```



- Press ctrl+w and search for the word “cgi.fix” the uncomment the line and set it to 0.

```
...
cgi.fix_pathinfo=0
...
```

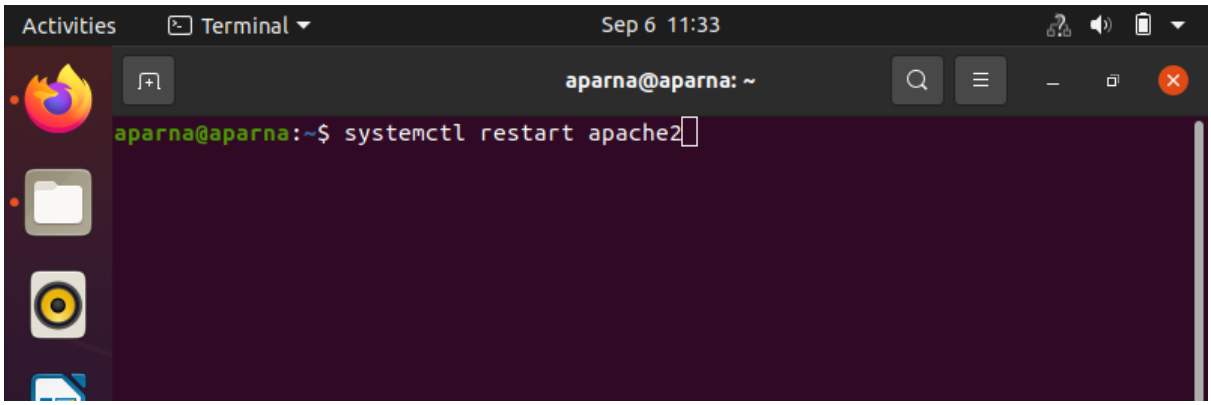

Activities
Terminal
Sep 6 11:25
aparna@aparna: /etc/php/7.4/apache2
GNU nano 4.8 php.ini
[PHP]
;;;;;;;;;;;;;;;;;;;
; About php.ini ;
;;;;;;;;;;;;;;;;;;;
; PHP's initialization file, generally called php.ini, is responsible for
; configuring many of the aspects of PHP's behavior.
; PHP attempts to find and load this configuration from a number of locations.
; The following is a summary of its search order:
; 1. SAPI module specific location.
; 2. The PHPRC environment variable. (As of PHP 5.2.0)
; 3. A number of predefined registry keys on Windows (As of PHP 5.2.0)
; 4. Current working directory (except CLI)
; 5. The web server's directory (for SAPI modules), or directory of PHP
; (otherwise in Windows)
; 6. The directory from the --with-config-file-path compile time option, or the
; Windows directory (usually C:\windows)
; See the PHP docs for more specific information.
; http://php.net/configuration.file
; The syntax of the file is extremely simple. Whitespace and lines
; beginning with a semicolon are silently ignored (as you probably guessed).
; Section headers (e.g. [Foo]) are also silently ignored, even though
; they might mean something in the future.
Search: cgi.fix
^G Get Help M-C Case Sens M-B Backwards ^P Older ^T Go To Line
^C Cancel M-R Regexp ^R Replace ^N Newer M-J FullJstify


Activities
Terminal
Sep 6 11:27
aparna@aparna: /etc/php/7.4/apache2
GNU nano 4.8 php.ini Modified
; if cgi.nph is enabled it will force cgi to always sent Status: 200 with
; every request. PHP's default behavior is to disable this feature.
;cgi.nph = 1
; if cgi.force_redirect is turned on, and you are not running under Apache or >
; (iPlanet) web servers, you MAY need to set an environment variable name that>
; will look for to know it is OK to continue execution. Setting this variable>
; cause security issues, KNOW WHAT YOU ARE DOING FIRST.
; http://php.net/cgi.redirect-status-env
;cgi.redirect_status_env =
; cgi.fix_pathinfo provides *real* PATH_INFO/PATH_TRANSLATED support for CGI. >
; previous behaviour was to set PATH_TRANSLATED to SCRIPT_FILENAME, and to not>
; what PATH_INFO is. For more information on PATH_INFO, see the cgi specs. S>
; this to 1 will cause PHP CGI to fix its paths to conform to the spec. A set>
; of zero causes PHP to behave as before. Default is 1. You should fix your >
; to use SCRIPT_FILENAME rather than PATH_TRANSLATED.
; http://php.net/cgi.fix-pathinfo
cgi.fix_pathinfo=0
; if cgi.discard_path is enabled, the PHP CGI binary can safely be placed outs>
; of the web tree and people will not be able to circumvent .htaccess security.
;cgi.discard_path=1
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell

Press Ctrl + x then y to Save and Exit.

- Now Restart The apache service.

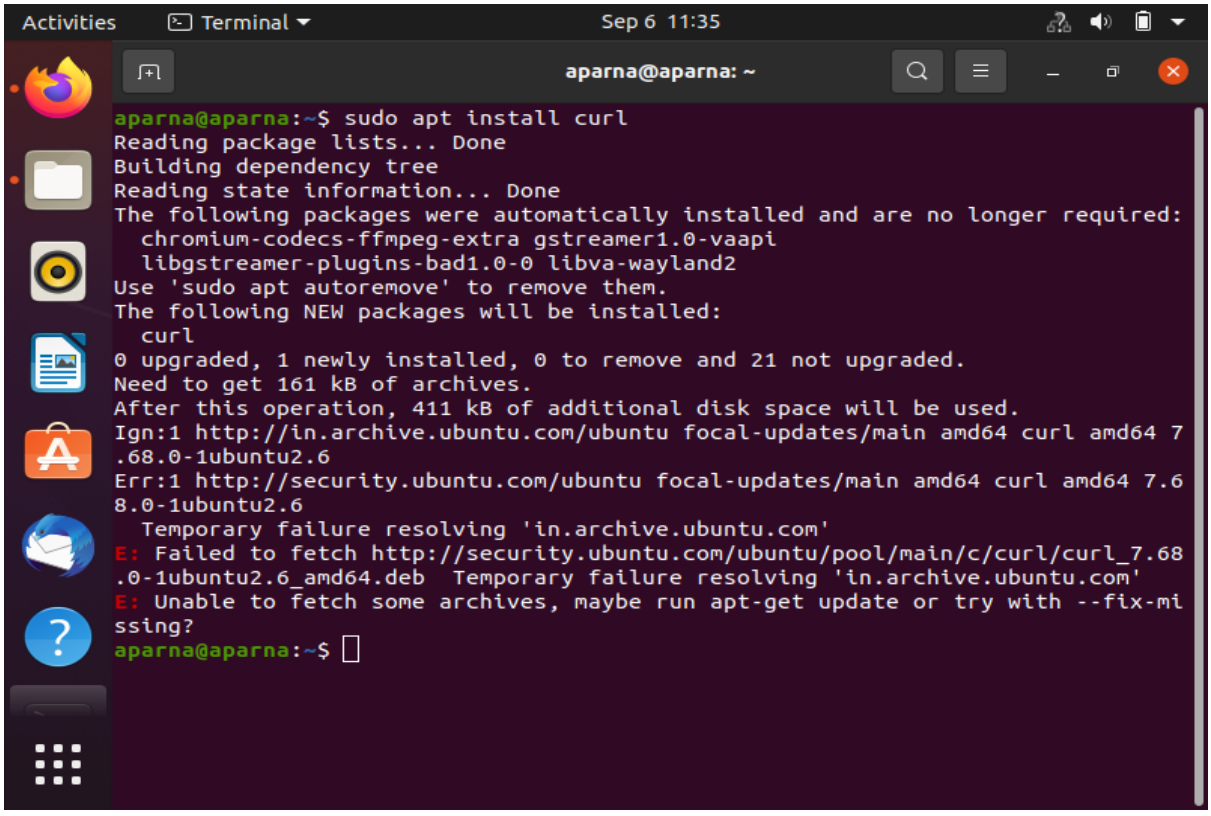
```
systemctl restart apache2
```

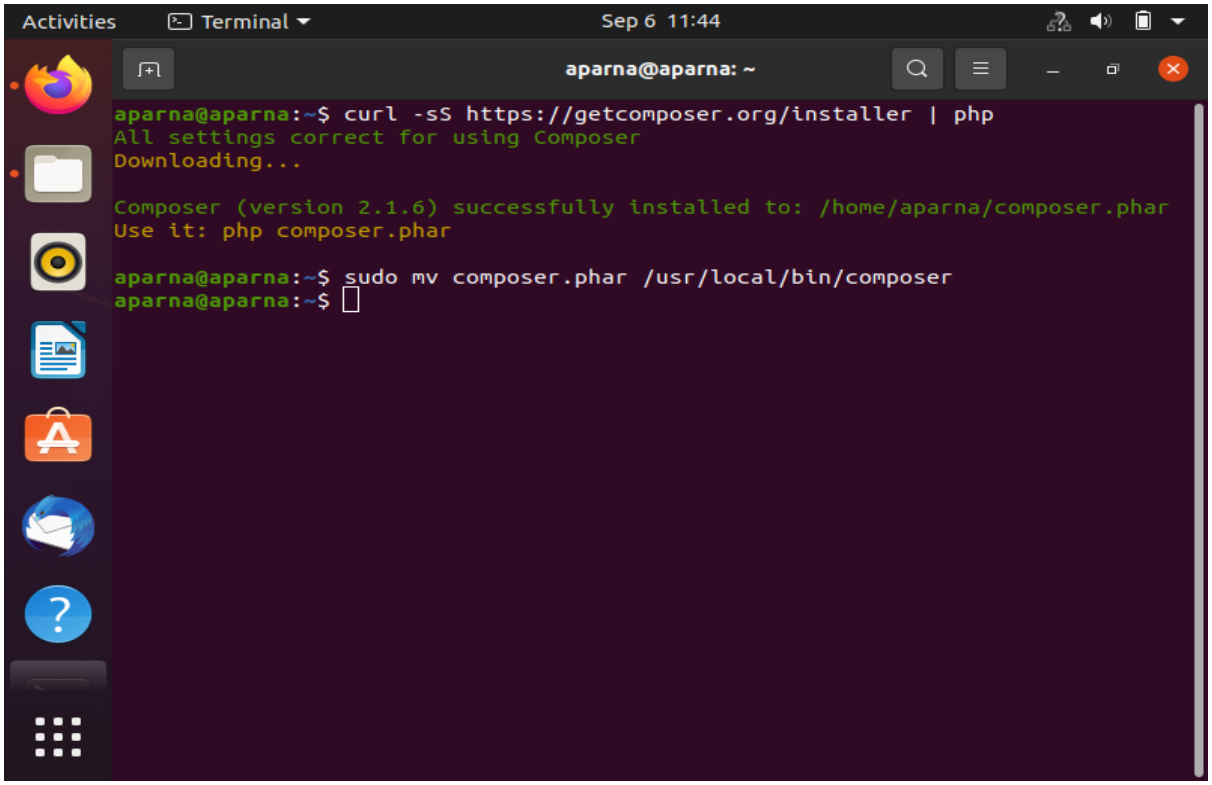


Step 3 – Install Composer PHP Packages Management

- Now it's time to install the composer package manager go ahead and download and install Composer. and move the composer .phar file to usr/local/bin/composer directory.

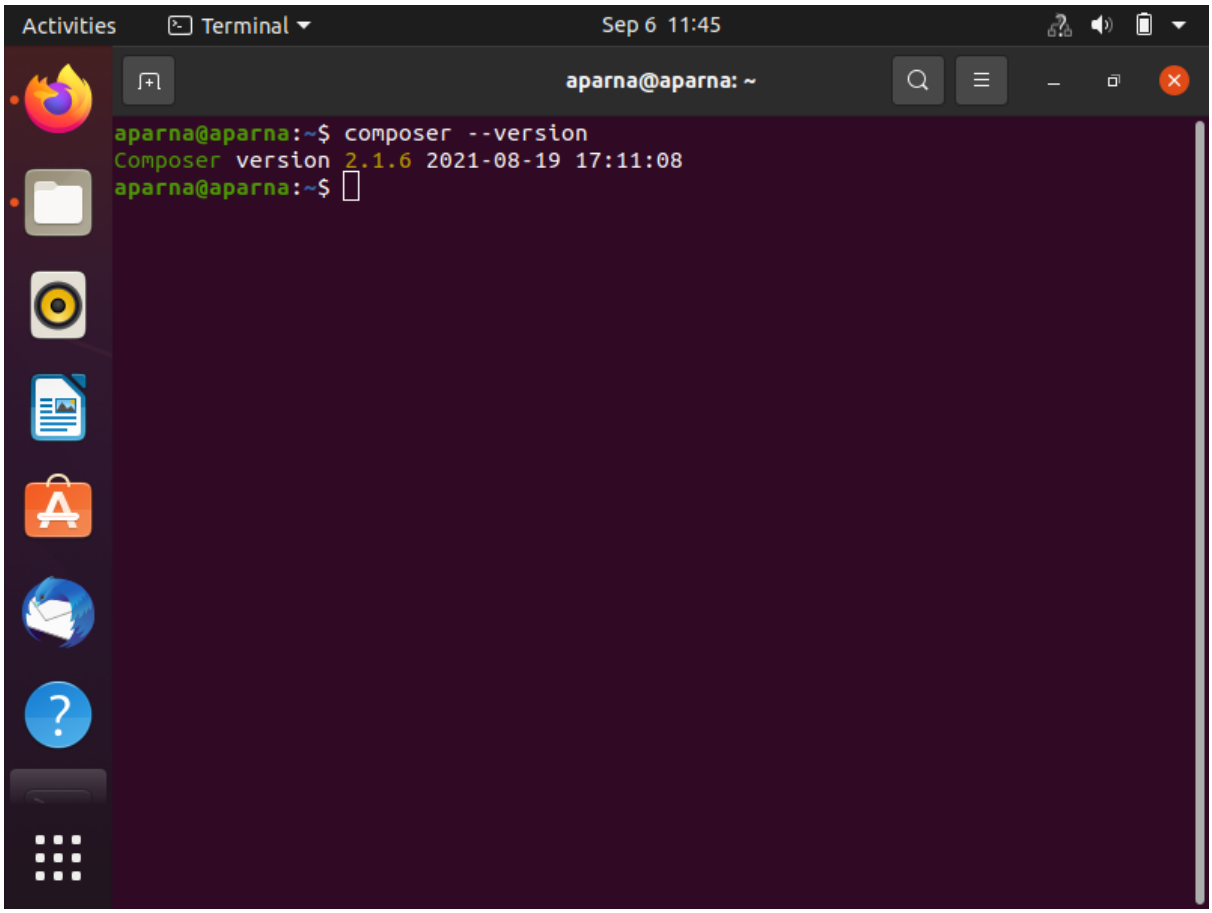
```
sudo apt install curl

curl -sS https://getcomposer.org/installer | php

sudo mv composer.phar /usr/local/bin/composer
```





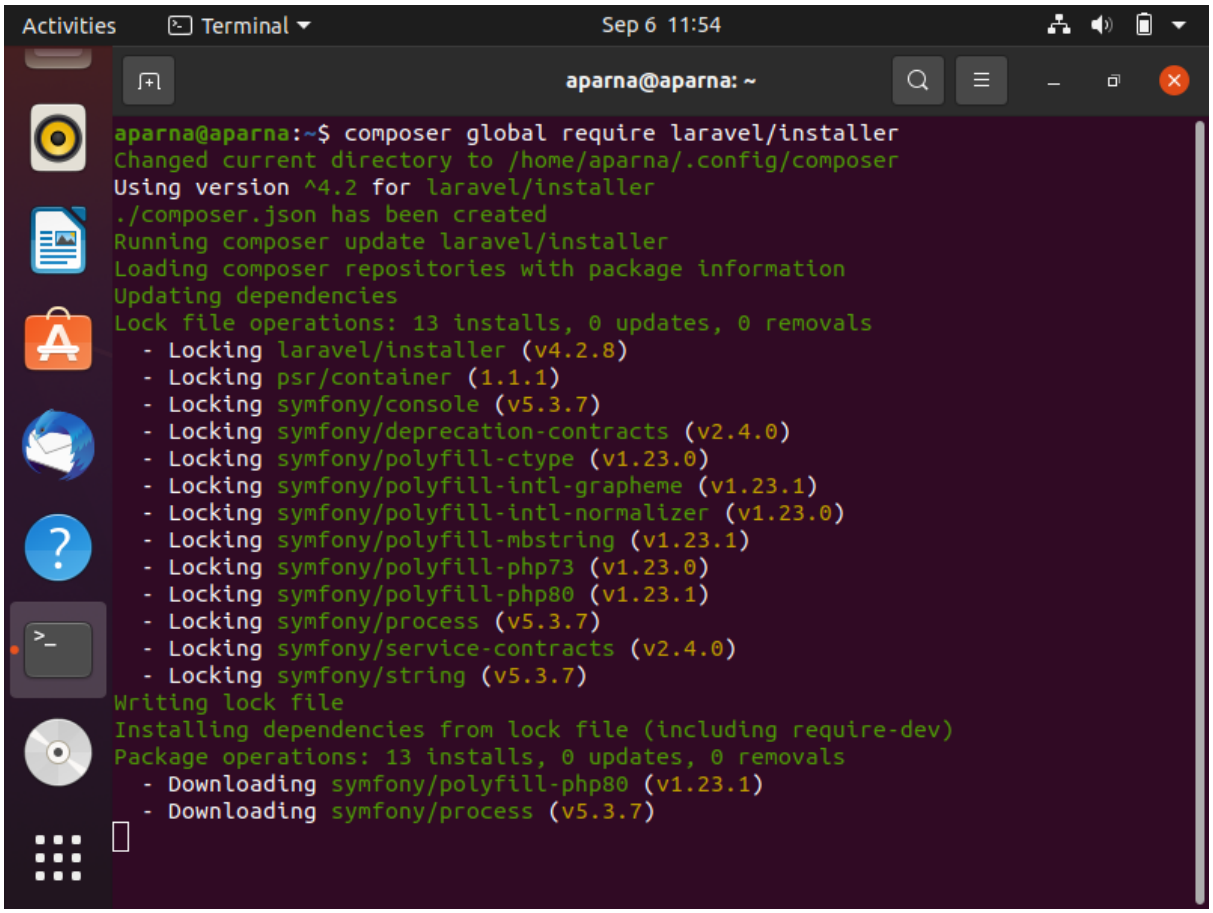
- You can check your installed composer version by typing the composer – version.

  composer –version

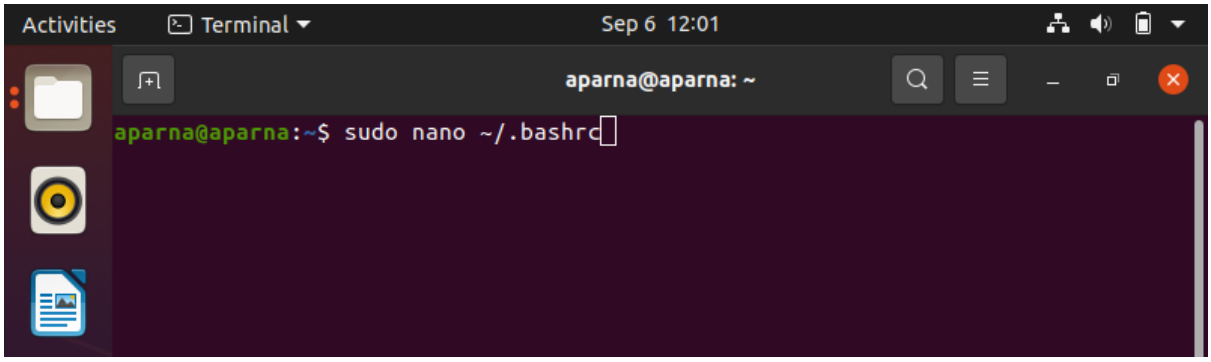


Step 4 – Install Laravel 8.x on Ubuntu 20.04

- Now install Laravel Framework using composer, just type composer global require Laravel/installer It will take a while to complete download its dependencies.
  composer global require laravel/installer

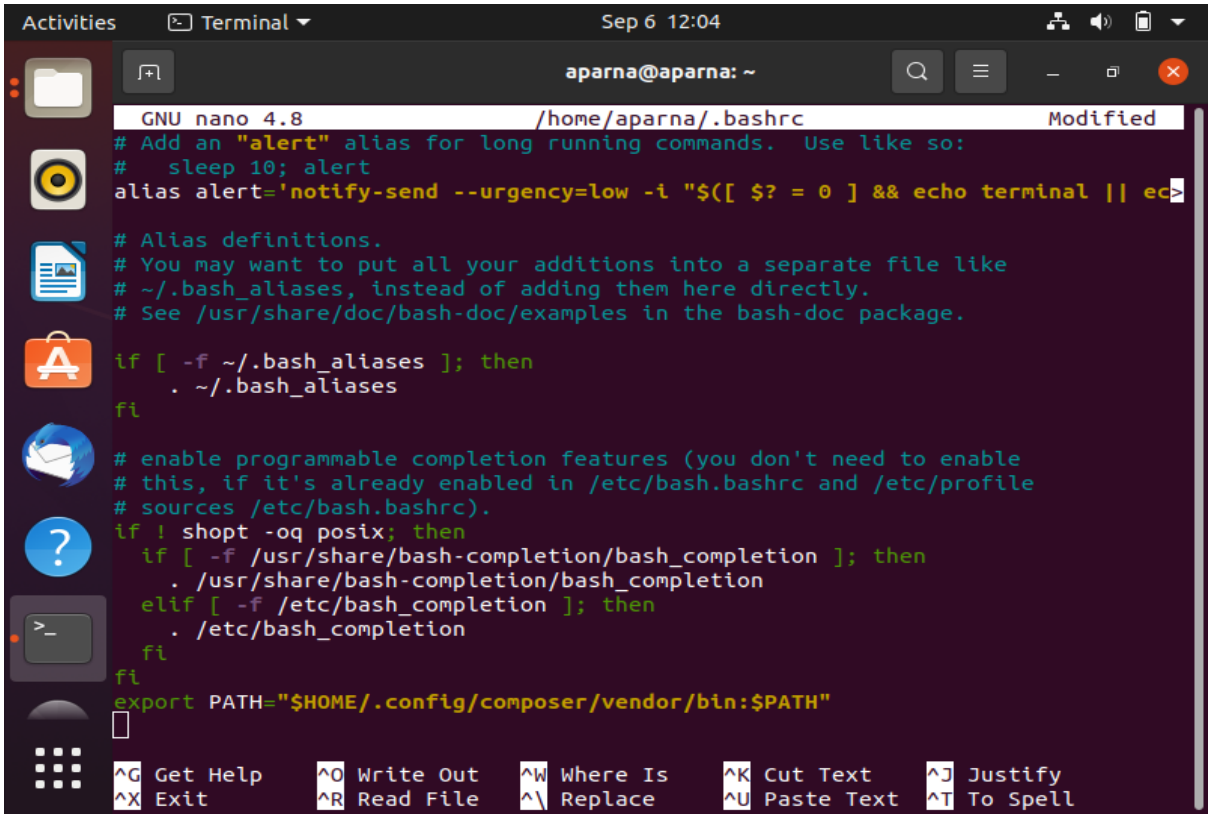


- As you had seen above image, all packages have been installed on the '~/.config/composer' directory. Next, we need to add the 'bin' directory to the PATH environment through the ~/.bashrc configuration. So Now Edit the ~/.bashrc configuration using nano command
  nano ~/.bashrc



And add the following line at the end of the file.

```
...
export PATH="$HOME/.config/composer/vendor/bin:$PATH"
...
```

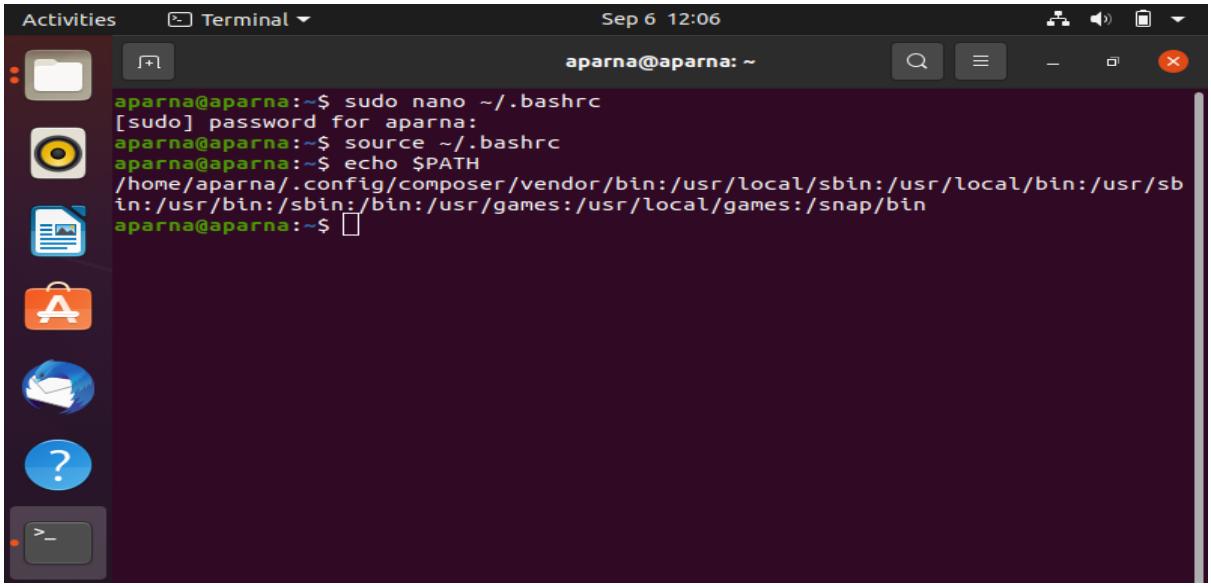


- Now reload your bashrc configuration using the source command.

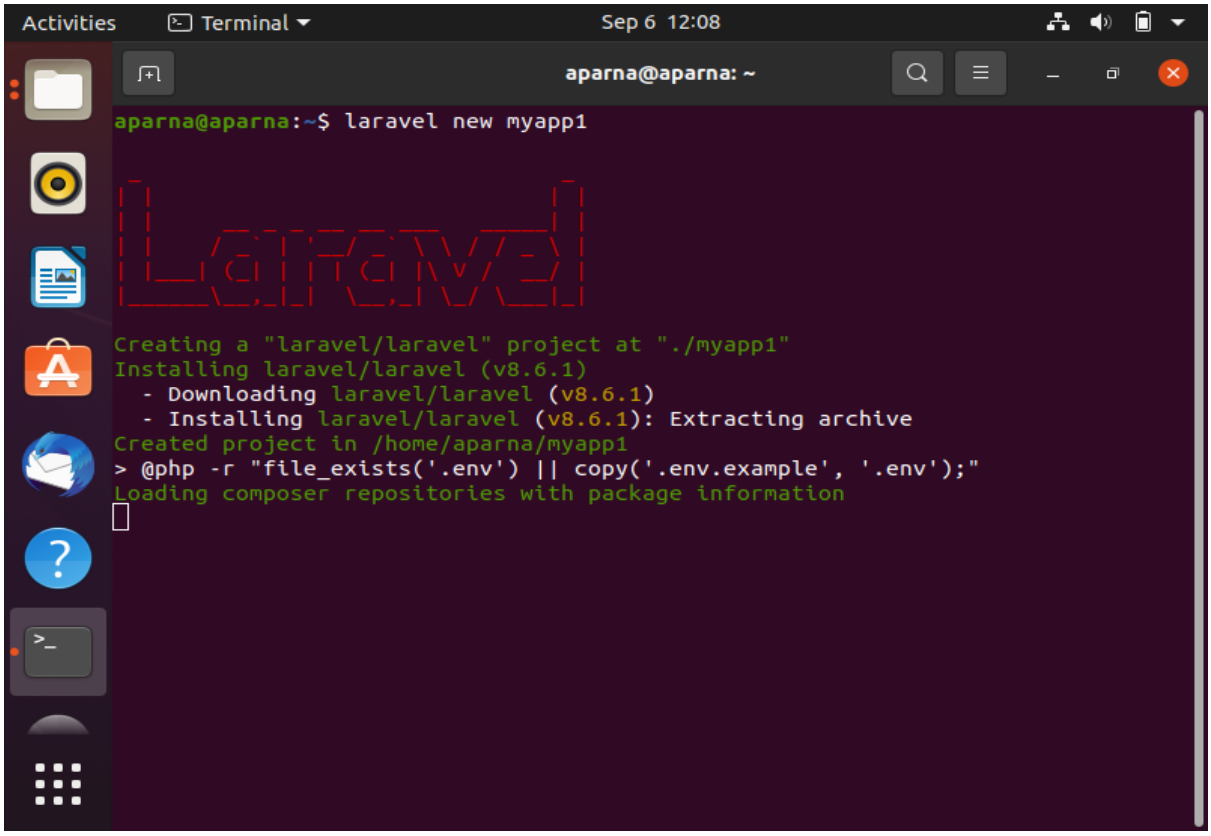
  ```
  source ~/.bashrc
  ```

- Now echo $PATH. It will return your “Bin” directory path for the Composer package.

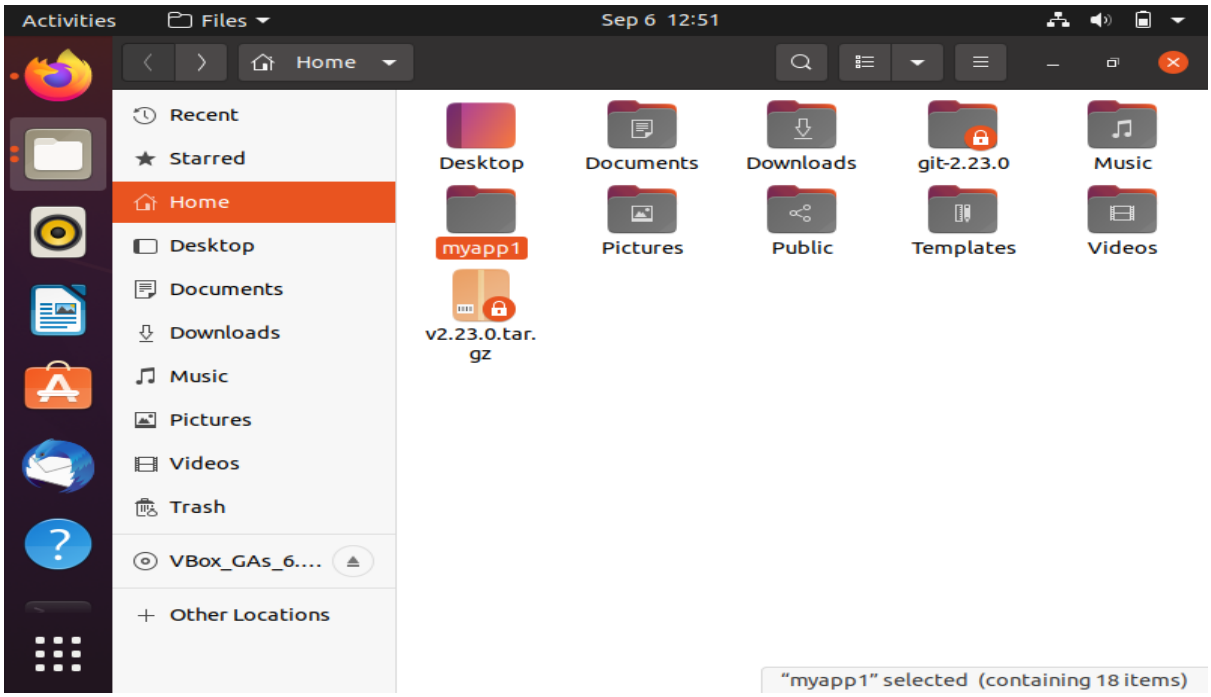
  ```
  echo $PATH
  ```



- The ‘bin’ directory for the composer packages has been added to the $PATH environment variable. And as a result, you can use the command ‘laravel’ to start and create a new project. Now go ahead and type Laravel new then your project name to start a new Laravel project
  laravel new myapp1



- Here you can see the installation of my new project myapp1 finished. You can also see inside my home directory a new directory has been created with my project name.

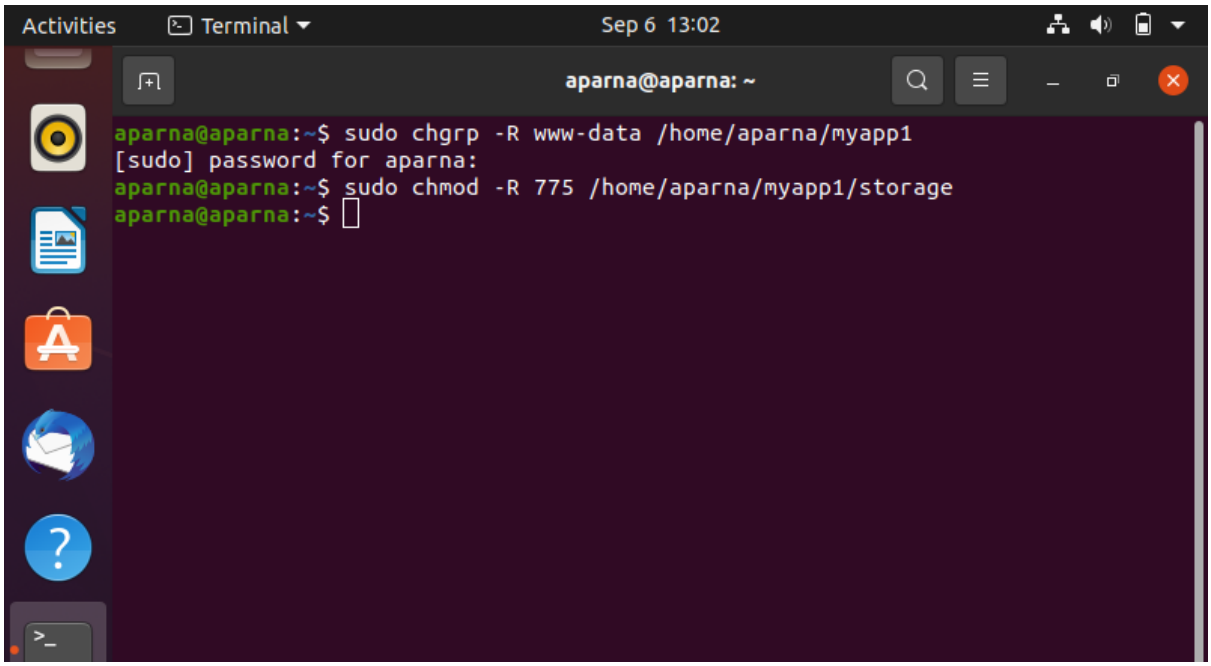


Step 5 – Finally Configure Apache for Laravel and test it.

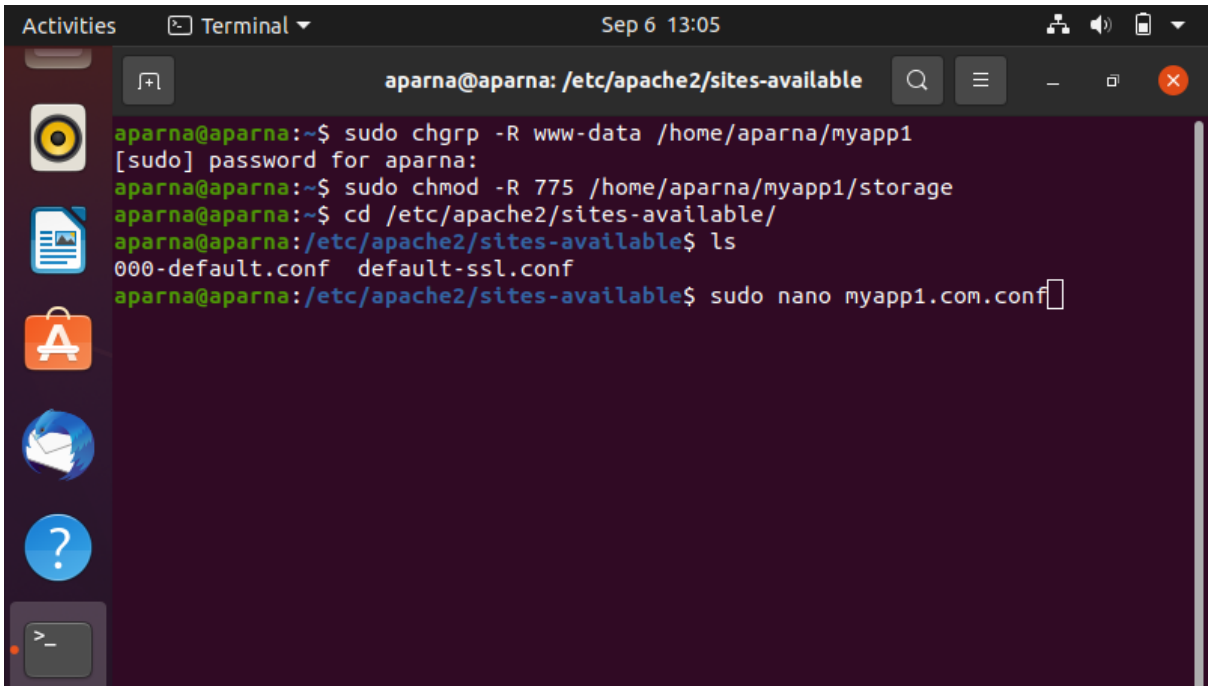
- First, add your project directory to www-data group use the following command

  sudo chgrp -R www-data /home/aparna/myapp1

- Also, you need to change access permission 775 of the storage directory under your project. So, go ahead and use the following command.
  sudo chmod -R 775 /home/aparna/myapp1/storage

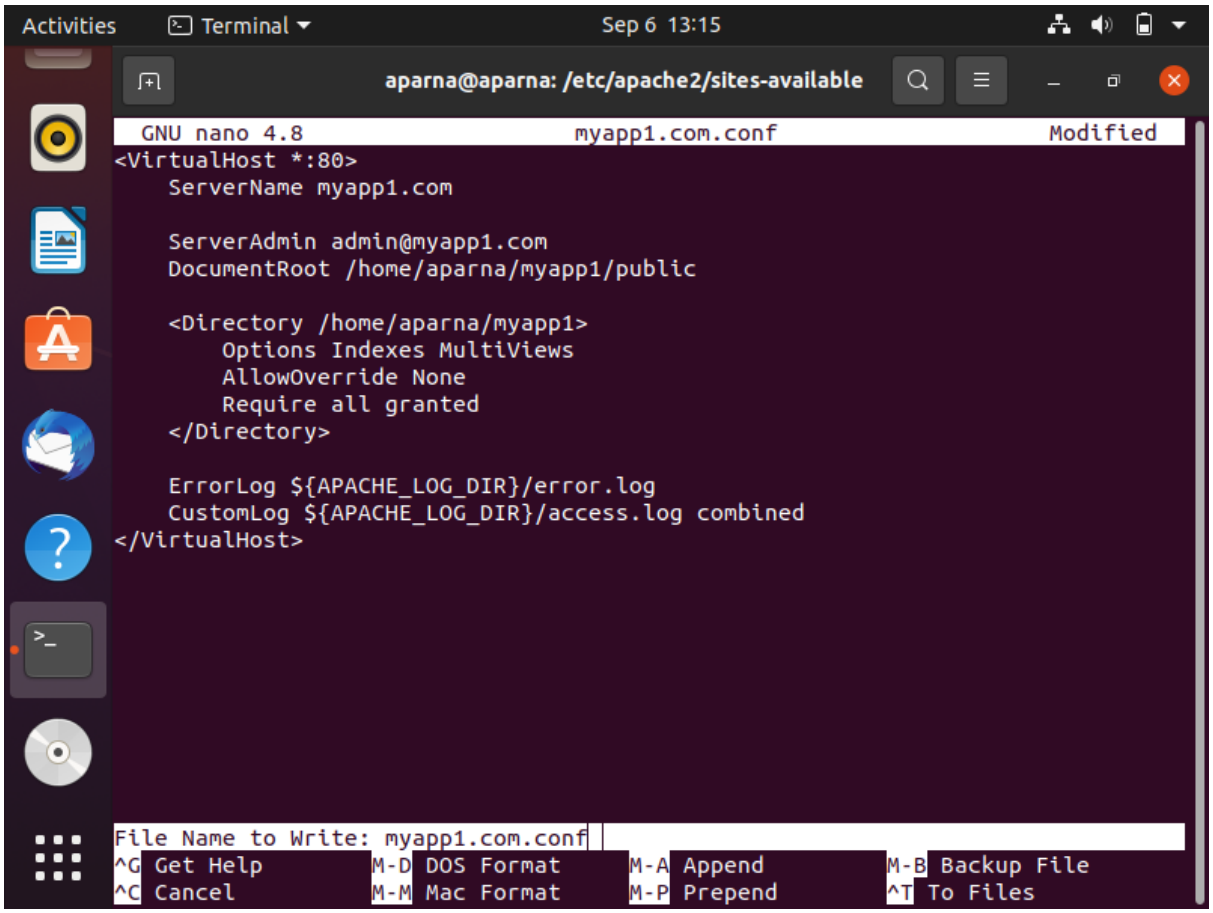


- Now create an apache vhost configuration go to the following directory and create a vhost config file using nano file editor.
  cd /etc/apache2/sites-available/
  sudo nano myapp1.com.conf



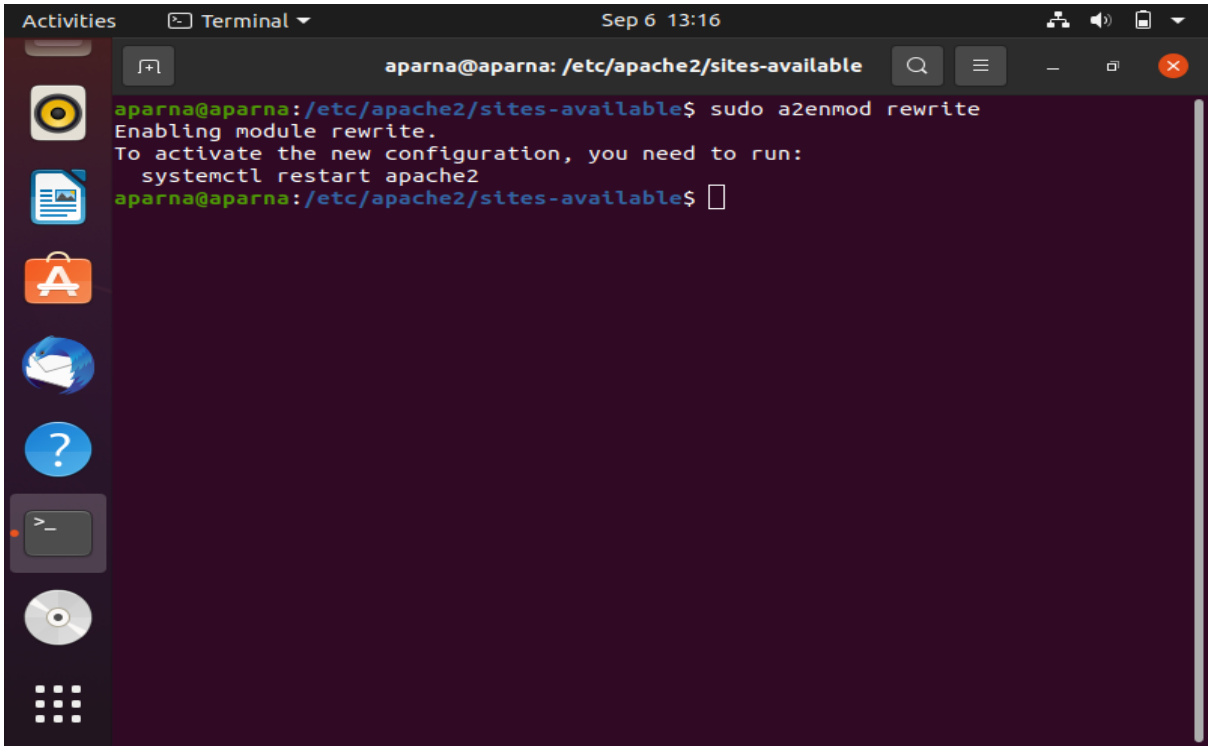
And paste the following line inside the file.

```
<VirtualHost *:80>
    ServerName myapp1.com

    ServerAdmin admin@myapp1.com
    DocumentRoot /home/aparna/myapp1/public

    <Directory /home/aparna/myapp1>
        Options Indexes MultiViews
        AllowOverride None
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

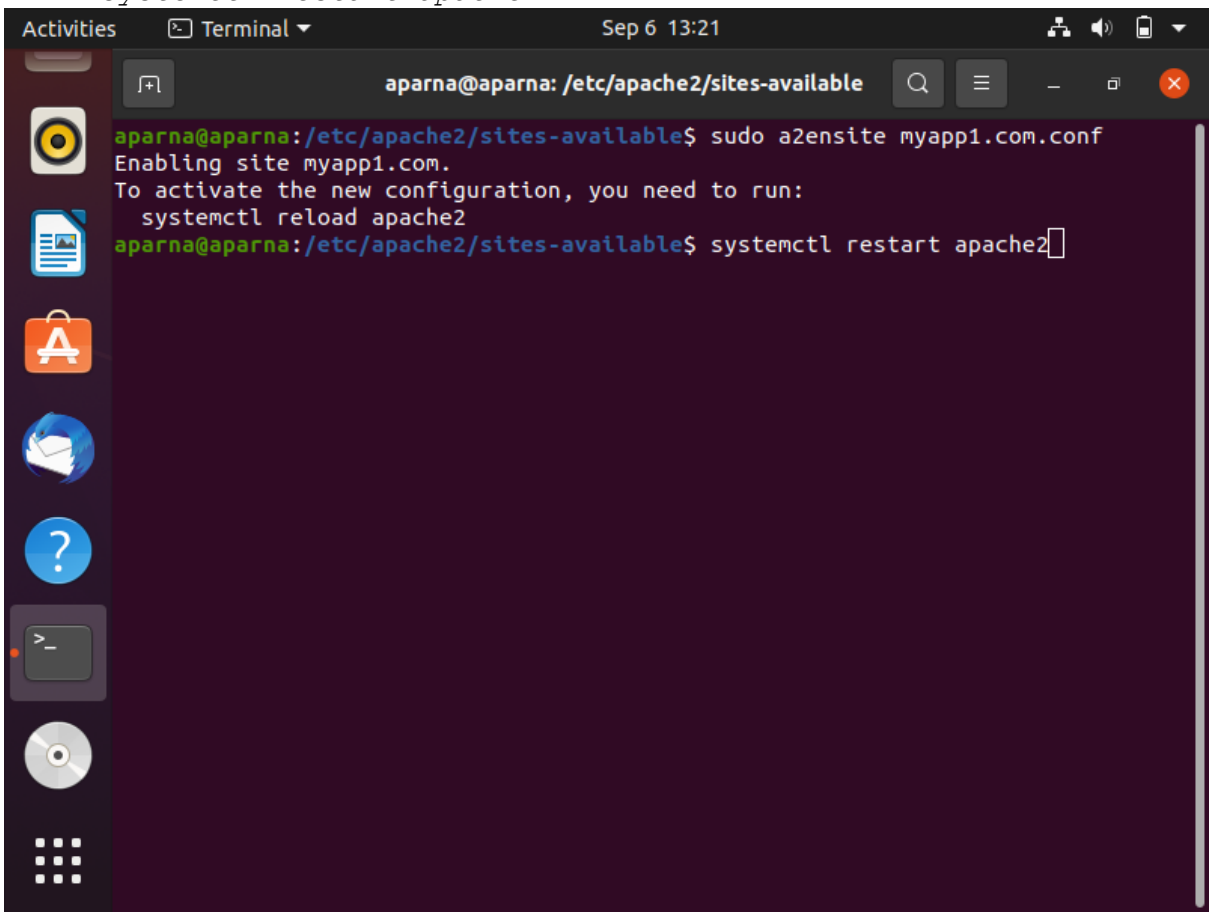


- Now enable mod rewrite for apache2 just type

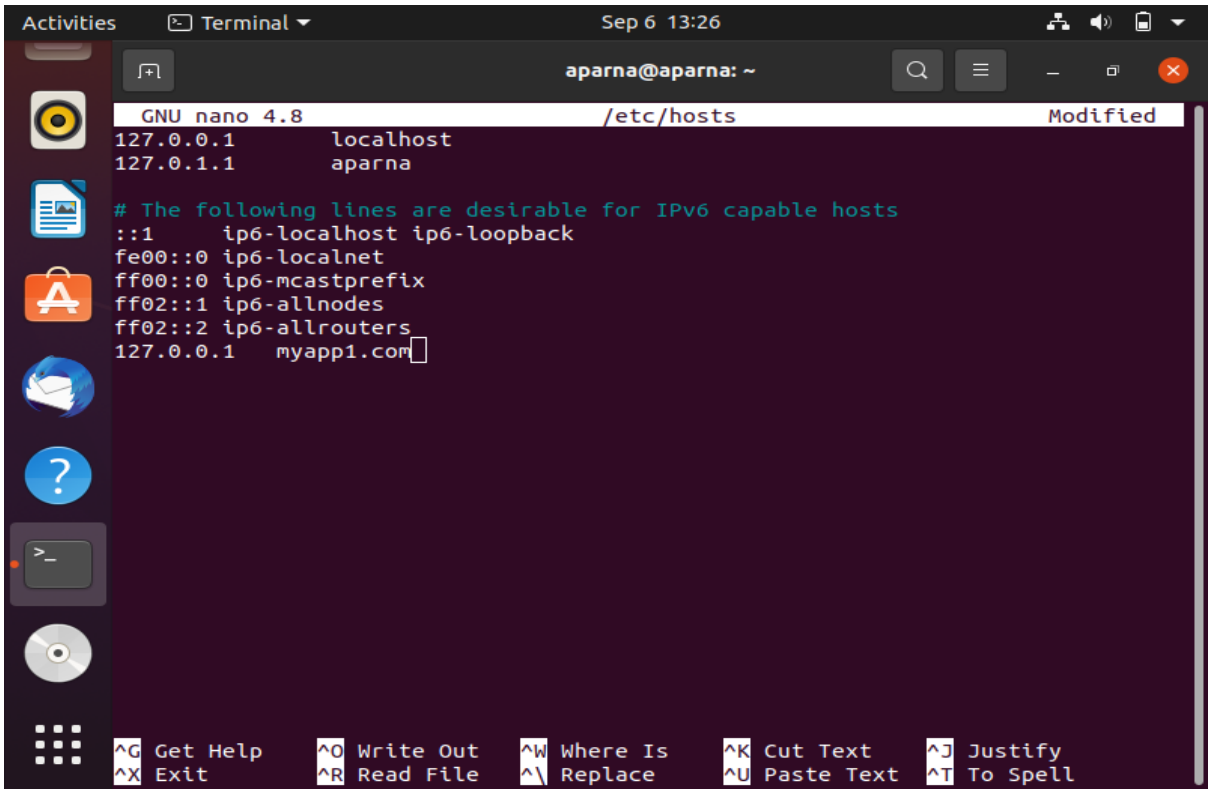
  ```
  sudo a2enmod rewrite
  ```



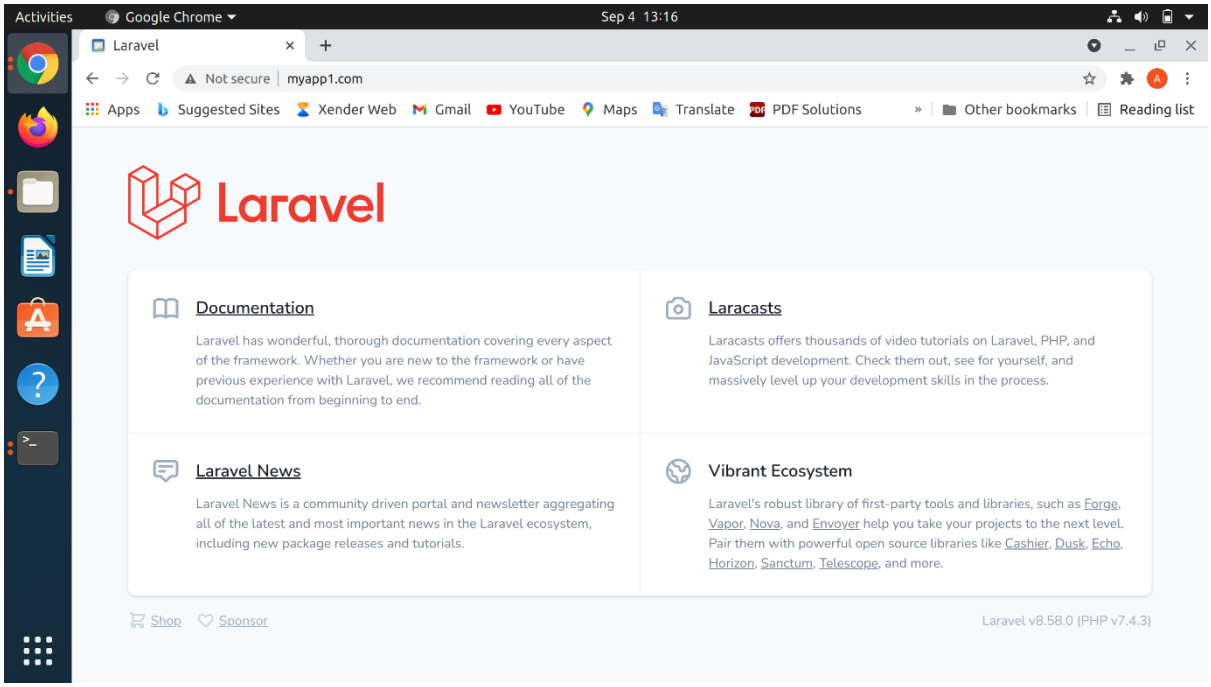
- Now enable your site, just type
  sudo a2ensite myapp1.com.conf
- Finally, Restart the apache service, type
  systemctl restart apache2



- As you are in a local environment you need a local dns resolver for your site. Go ahead and edit /etc/hosts file, add a dns record for your site then save the file.
  sudo nano /etc/hosts
  ...
  127.0.0.1 myapp1.com
  ...



- Now get back to the web browser and open a tab then type your project hostname.



Experiment No. 7

Aim

Build and install software from source code, familiarity with make and cmake utilities expected.

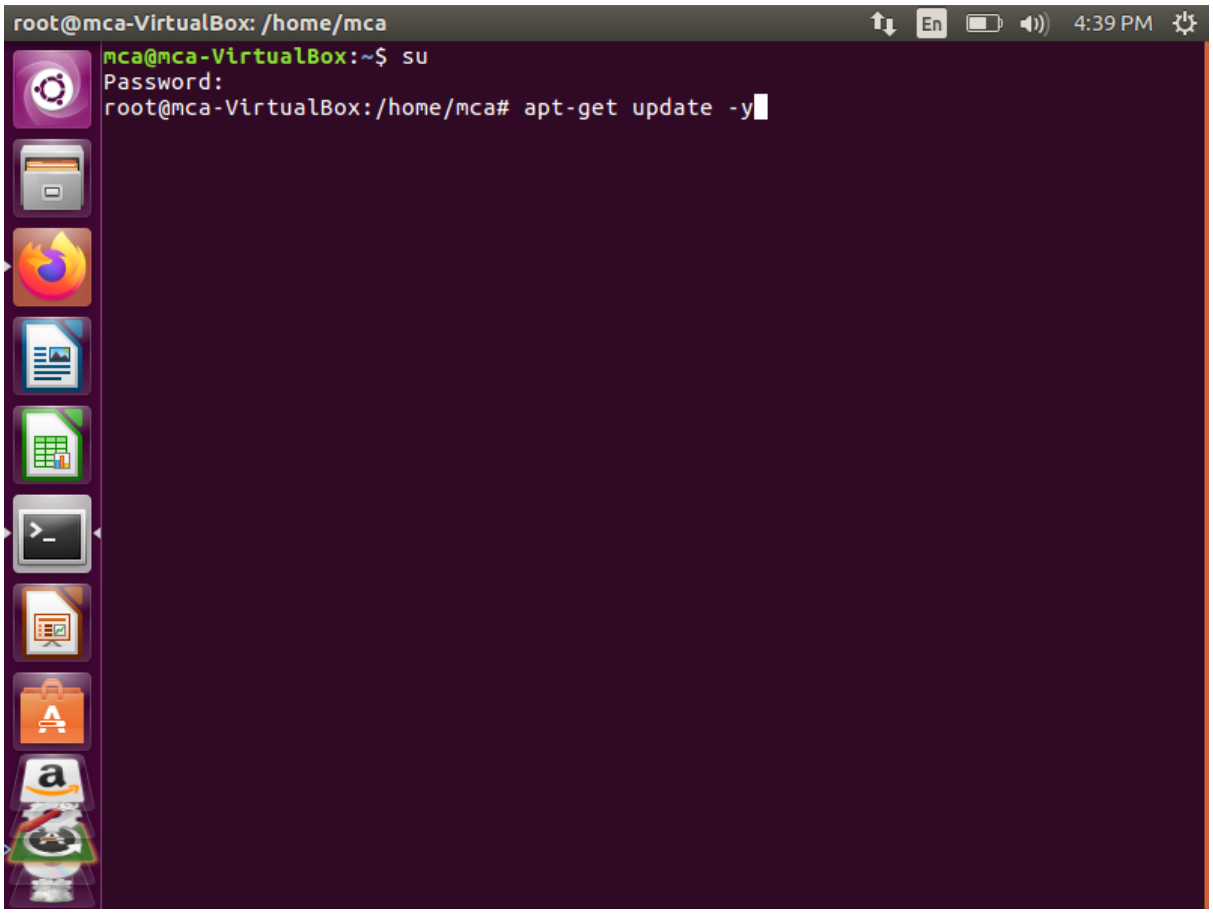
Procedure

Here you will be installing the git versioning software system. Make sure you are logged in as the root user.

- Step 1: Get The Server Ready

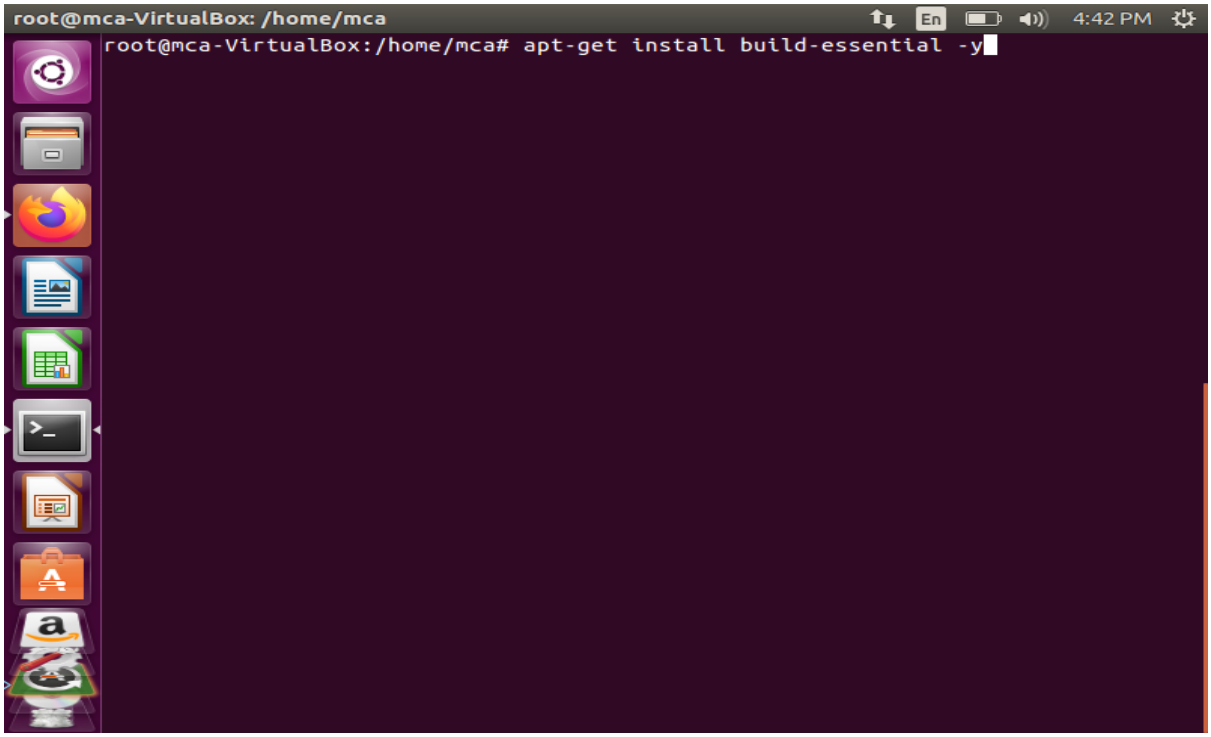
Make sure your packages are up to date:

```
apt-get update -y
```



- Next, you'll need to make sure you have a compiler available. Run this command to install build-essential:

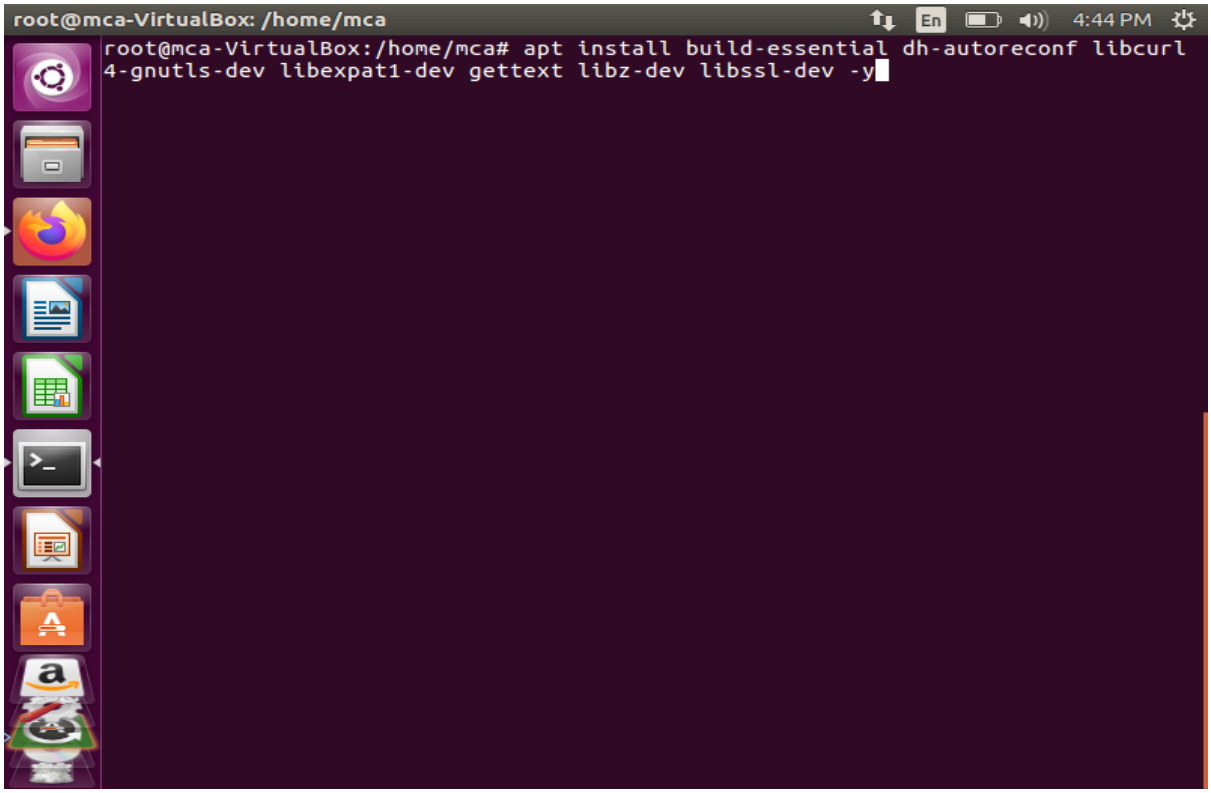
```
apt-get install build-essential -y
```



➢ Step 2: Download Dependencies

When installing a package from source code, you'll need to manage the installation of the package dependencies. We'll use apt-get to install git's dependencies:

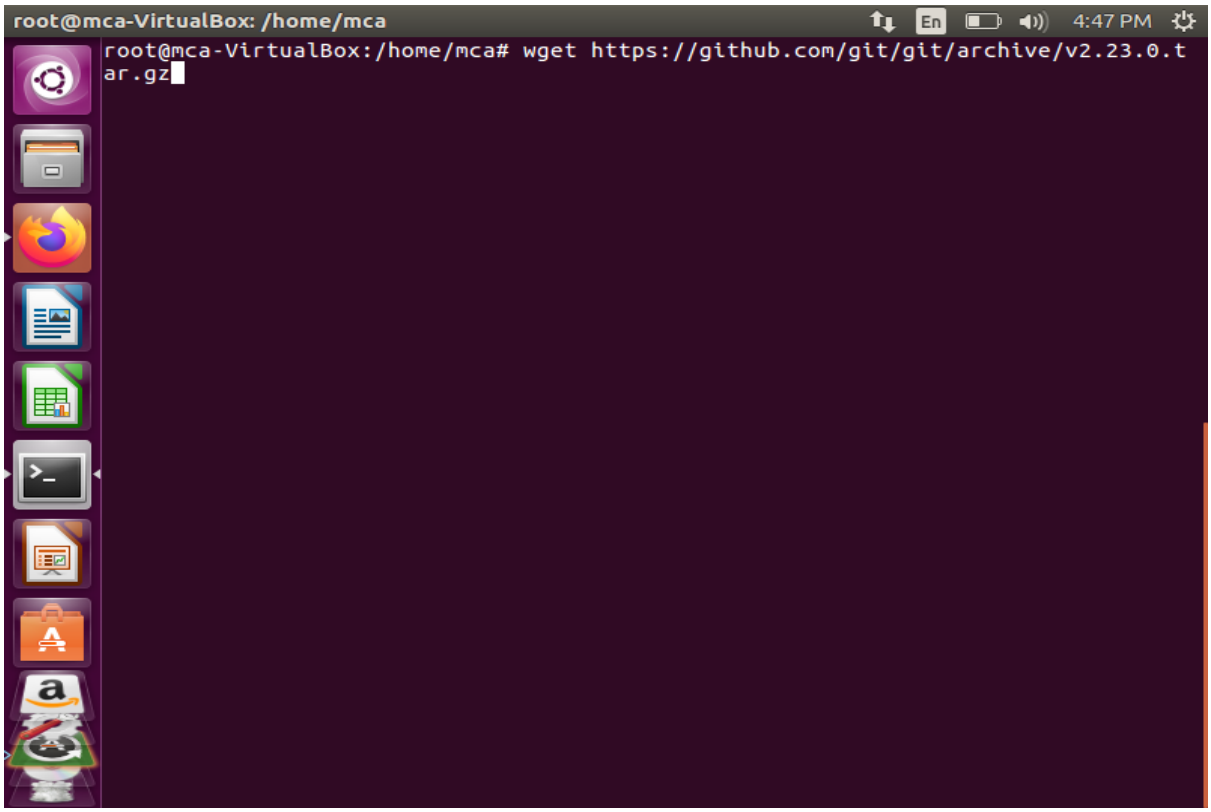
```
apt install build-essential dh-autoreconf libcurl4-gnutls-dev libexpat1-dev gettext libz-dev libssl-dev -y
```

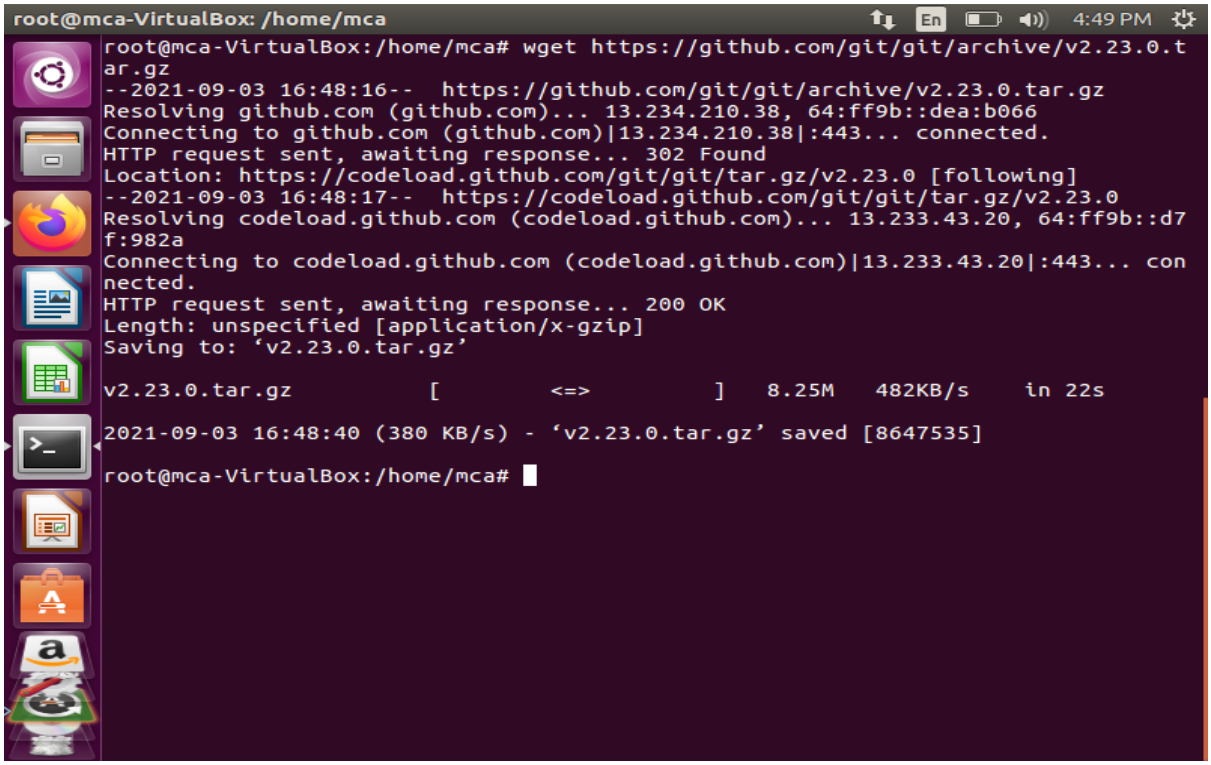


➢ Step 3: Download The Source Package

Once the package dependencies are in place, download the package with wget:

```
wget https://github.com/git/git/archive/v2.23.0.tar.gz
```

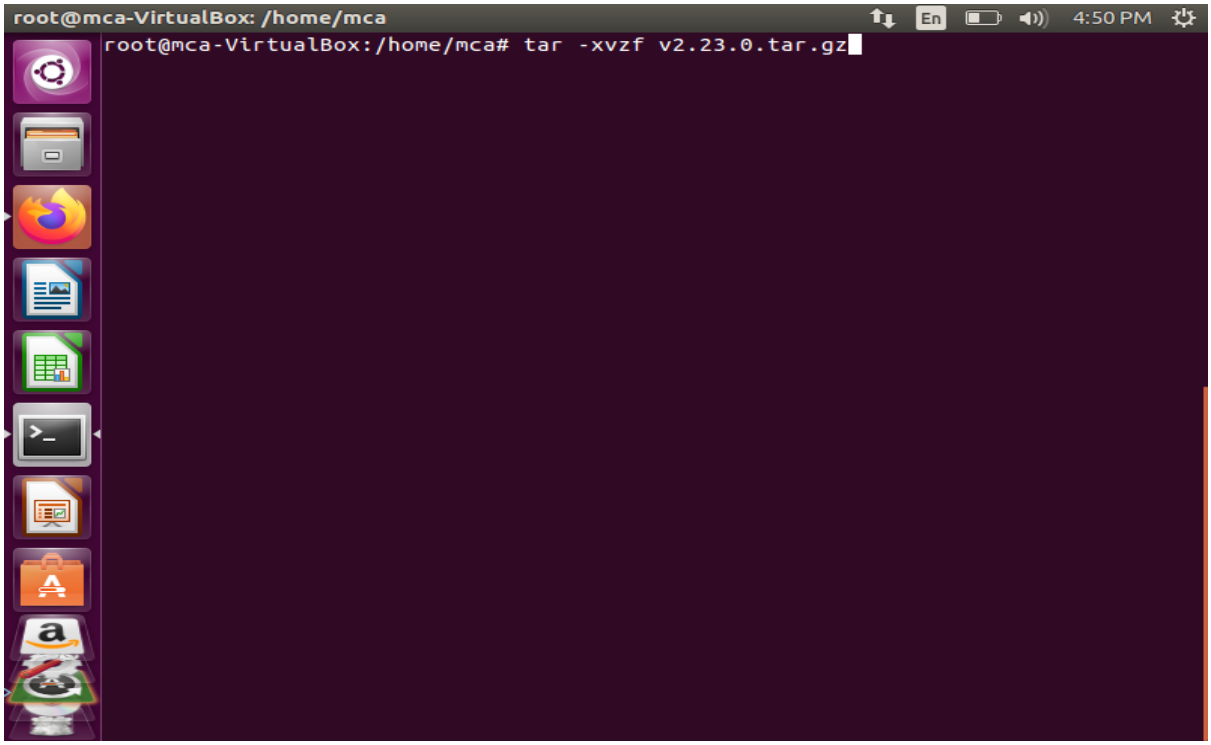


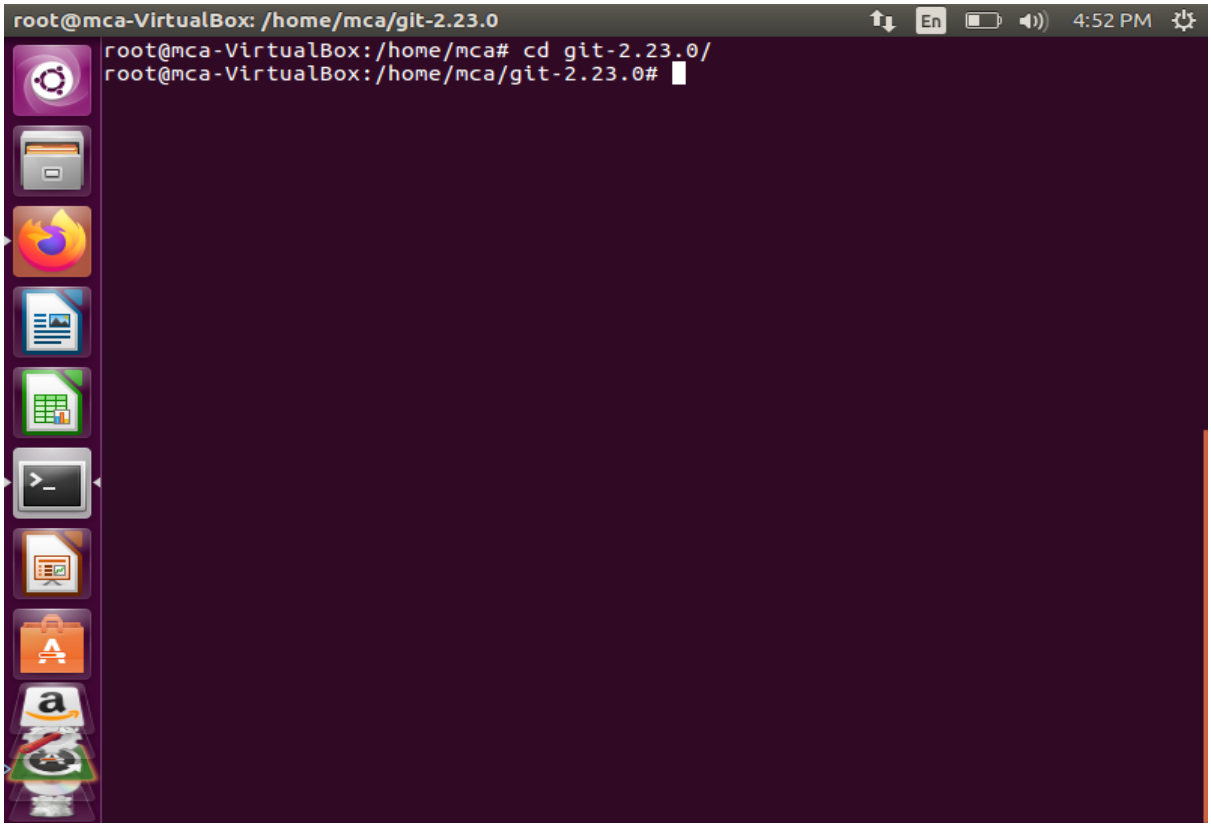


- Next, we need to extract the archive and cd (change directories) into the new git directory:

`tar -xvzf v2.23.0.tar.gz`

`cd git-2.23.0/`





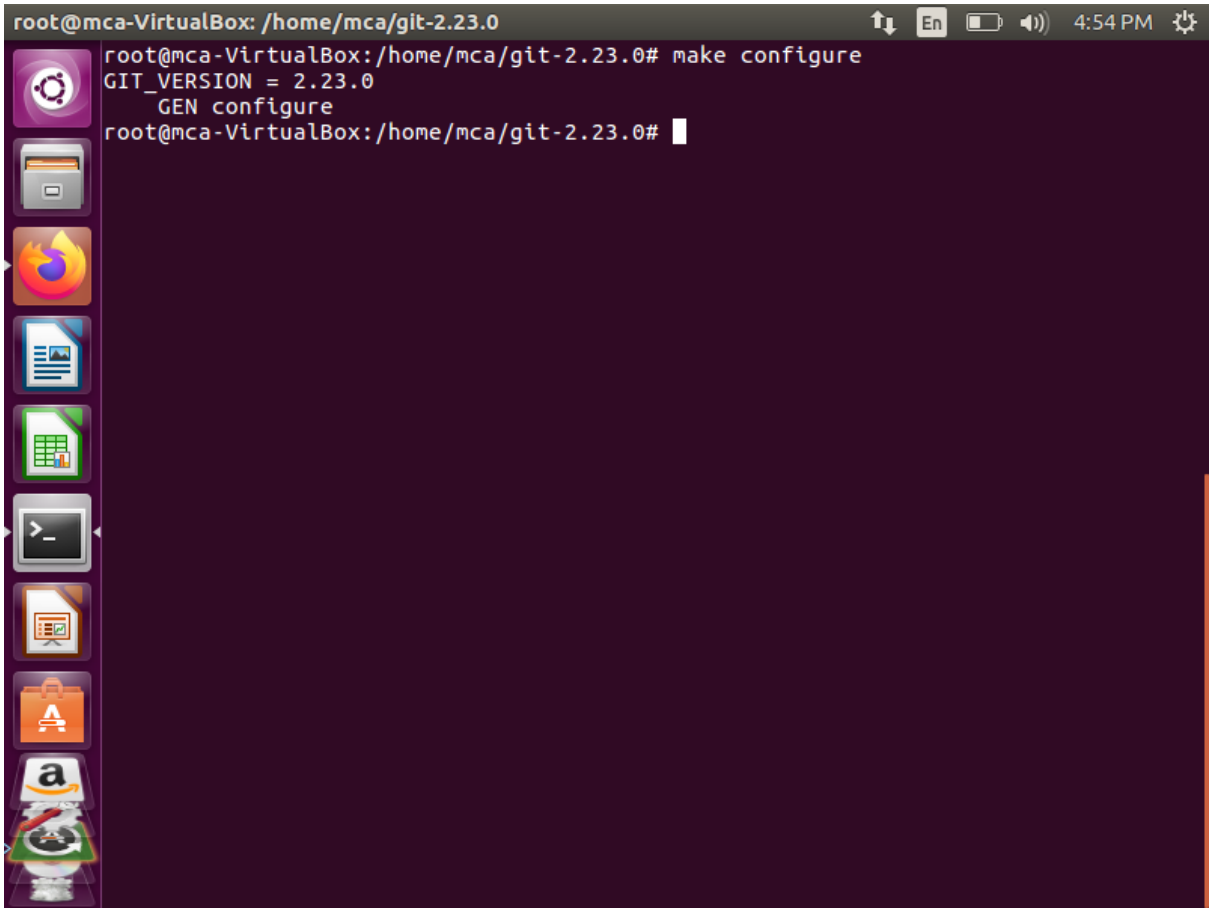
➢ Step 4: Install Git

Now that we have our package extracted and ready to go, we need to configure it:

```
make configure
```

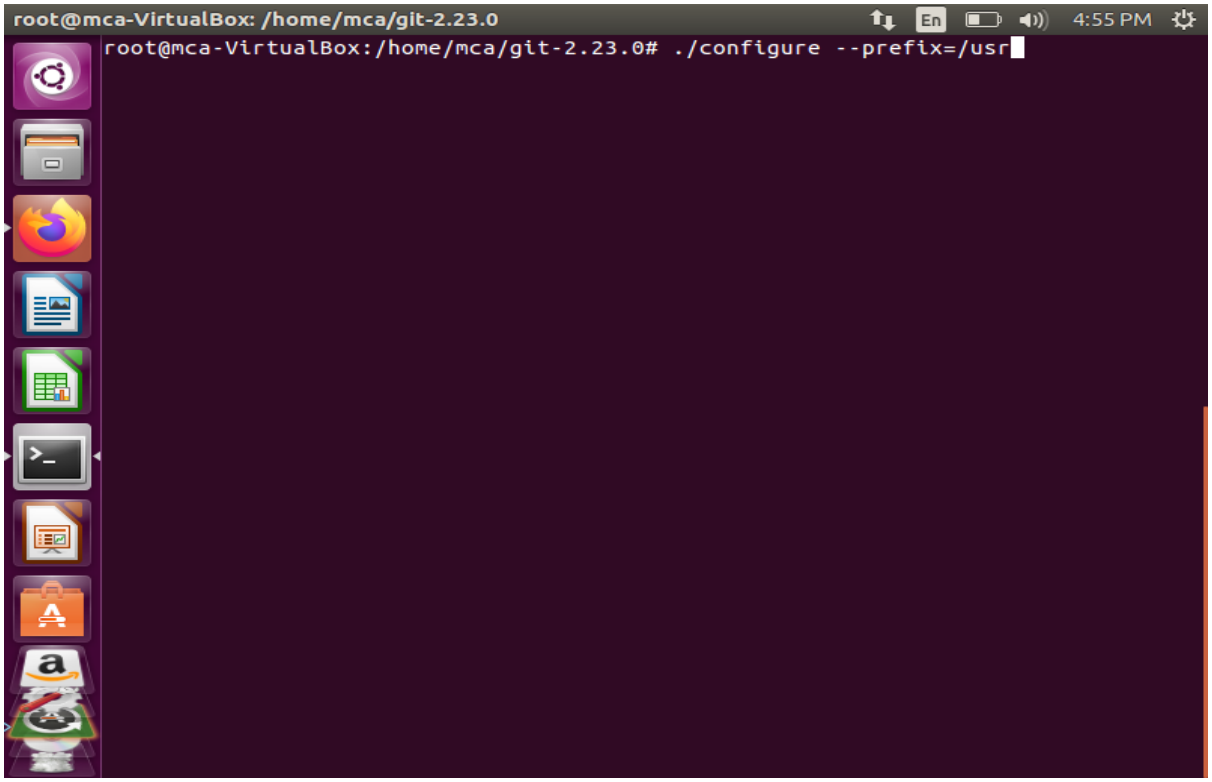
You should see an output similar to this:

```
GIT_VERSION = 2.23.0
GEN configure
```



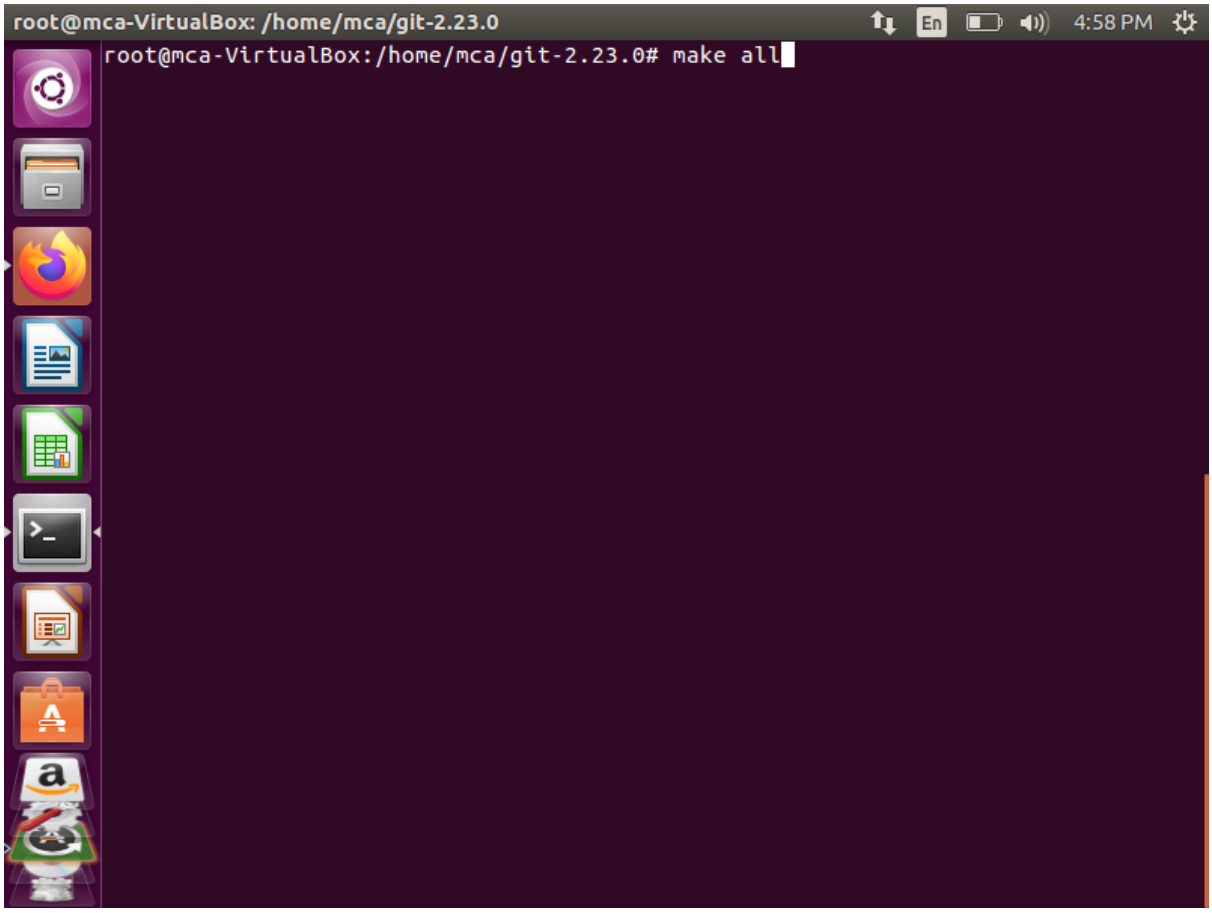
- Next, let's verify that all of the dependencies necessary to build the package are available by running this command:

```
./configure --prefix=/usr
```



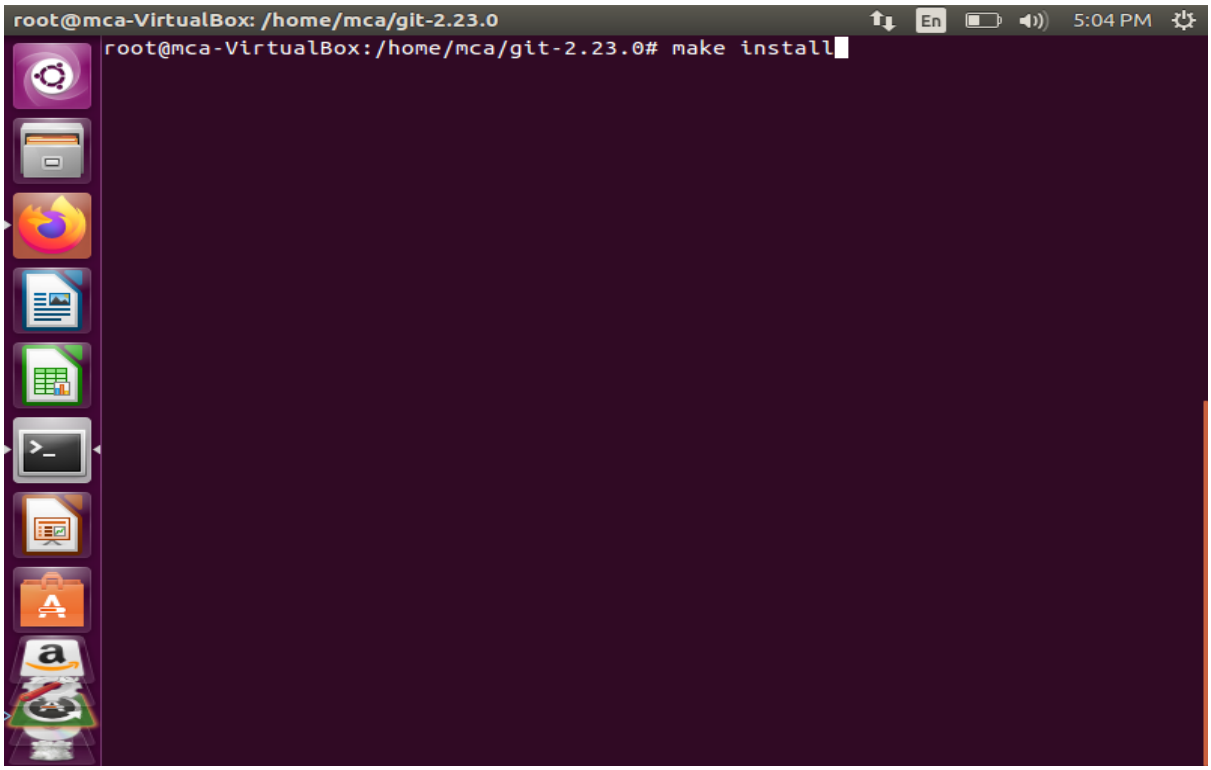
- After that, we'll build the source code:

```
make all
```



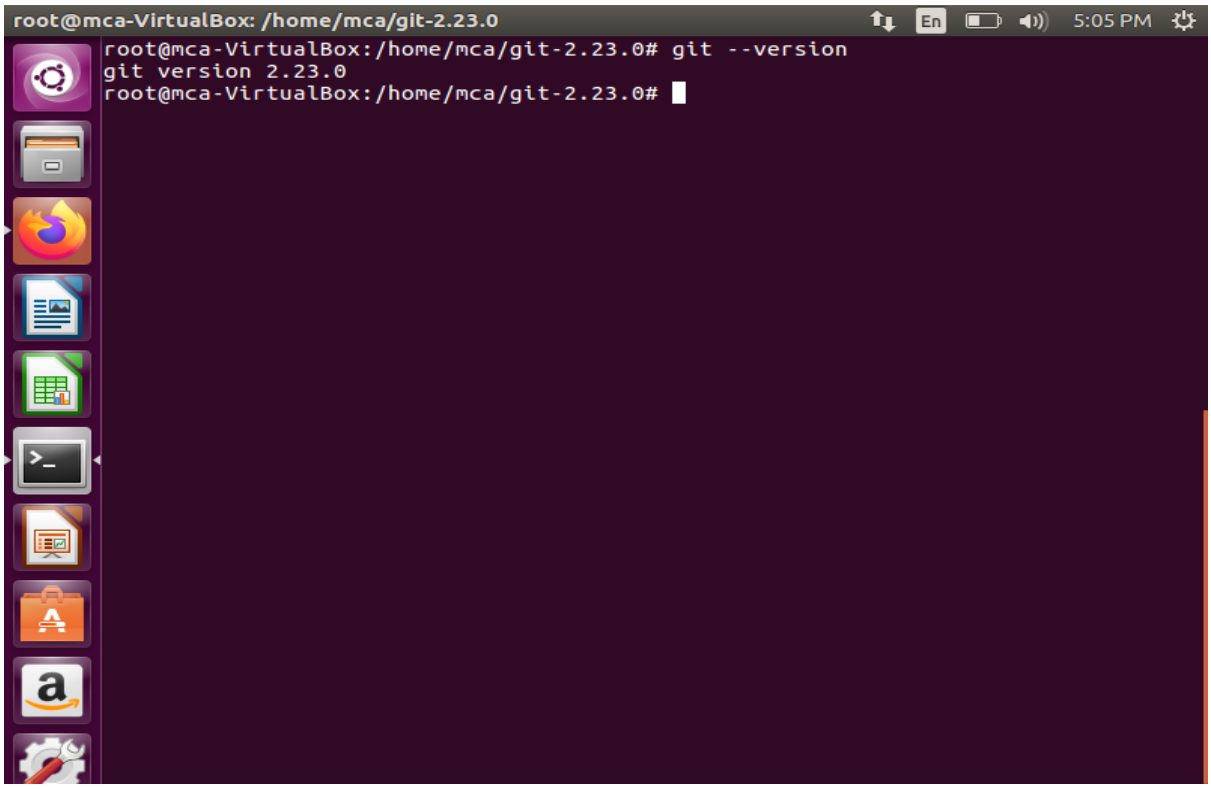
- Now afterthe binaries are all built, install git:

```
make install
```



- Verify that git is working with version 2.23.0:

```
git -version
```



Experiment No. 8

Aim

Introduction to command line tools for networking IPv4 networking, network commands: ping route traceroute, nslookup, ip. Setting up static and dynamic IP addresses. Concept of Subnets, CIDR address schemes, Subnet masks, iptables, setting up a firewall for LAN, Application layer (L7) proxies.

Result

The network infrastructure is a very complex structure of cables, routers, access points, data packets and a million other small components that together make the entire network work seamlessly. Any issue in any of these smaller components may lead to an overall collapse of the network infrastructure. This may lead to disruption of WiFi, cellular and wired(ethernet) infrastructure. This is the reason why it is very important to have an access to how the network is performing and know troubleshooting techniques.

The operating system acts as an intermediate platform between the user and the underlying network infrastructure. To use the below commands in Windows operating system, one needs to click on Start, go to Run and type cmd. This will open up the command prompt. In Mac OS, you can use the terminal application.

Ipv4 Networking

The operating system consists of various built-in, command-line networking utilities that are used for network troubleshooting. IP is part of an internet protocol suite, which also includes the transmission control protocol. Together, these two are known as TCP/IP. The internet protocol suite governs rules for packetizing, addressing, transmitting, routing, and receiving data over networks.

IP addressing is a logical means of assigning addresses to devices on a network. Each device connected to the internet requires a unique IP address. Most networks that handle internet traffic are packet-switched. Small units of data, called packets, are routed through a network. A source host, like your computer, delivers these IP packets to a destination host, such as a server, based on IP addresses in packet headers. Packet-switching allows many users on a network to share the same data path.

An IP address has two parts—-one part identifies the host, such as a computer or other device. And the other part identifies the network it belongs to. TCP/IP uses a subnet mask to separate them.

*IP (version 4) addresses are 32-bit integers that can be expressed in hexadecimal notation. The more common format, known as dotted quad or dotted decimal, is x.x.x.x, where each x can be any value between 0 and 255. For example, 192.0.2.146 is a valid IPv4 address.

IPv4 still routes most of today's internet traffic. A 32-bit address space limits the number of unique hosts to 232, which is nearly 4.3 billion IPv4 addresses for the world to use (4,294,967,296, to be exact).

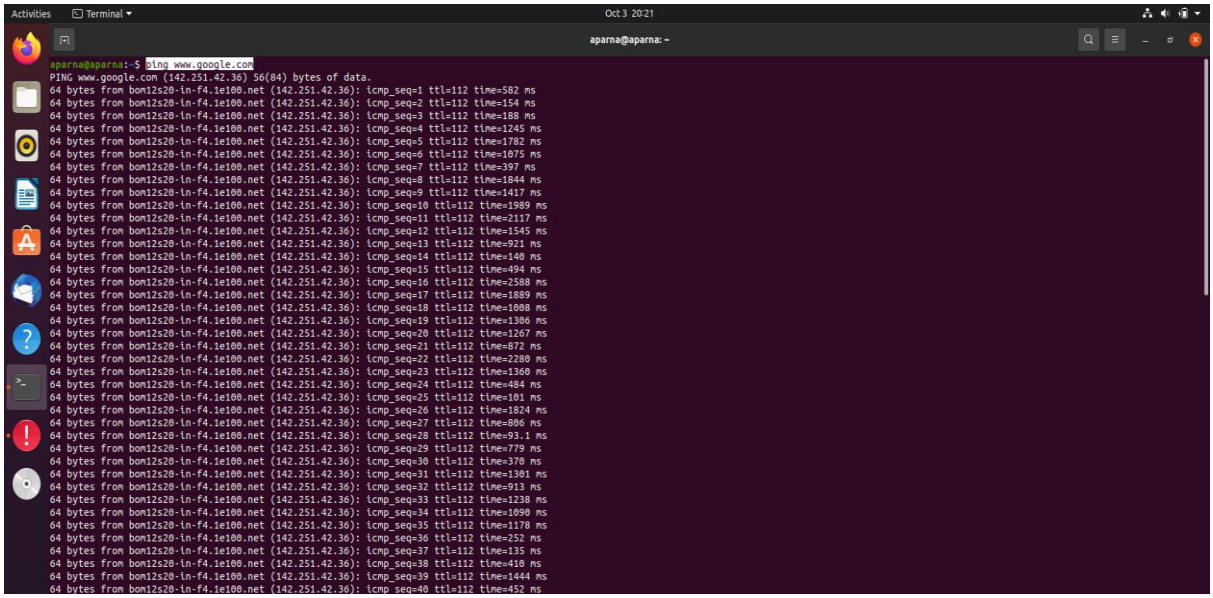
Network Commands

ping: Ping command is typically used for checking the network connectivity from your system to an end device like a server or a printer and also of a website. This command is used while troubleshooting the entire network. So, when you enter a URL in your web browser, what you are actually doing is instructing your machine to connect to the website name. The website name is actually an alias for the IP address.

So this command can be used in two ways:

1. It can be used to ping a network IP address.

2. It can used to ping a website or hostname directly.



Route: Using the route command displays or modifies the computer's routing table.For a typical computer that has a single network interface and is connected to a local area network (LAN) that has a router, the routing table is pretty simple and isn't often the source of network problems. Still, if you're having trouble accessing other computers or other networks, you can use the route command to make sure that a bad entry in the computer's routing table isn't the culprit.

For a computer with more than one interface and that's configured to work as a router, the routing table is often a major source of trouble. Setting up the routing table properly is a key part of configuring a router to work.

Syntax:

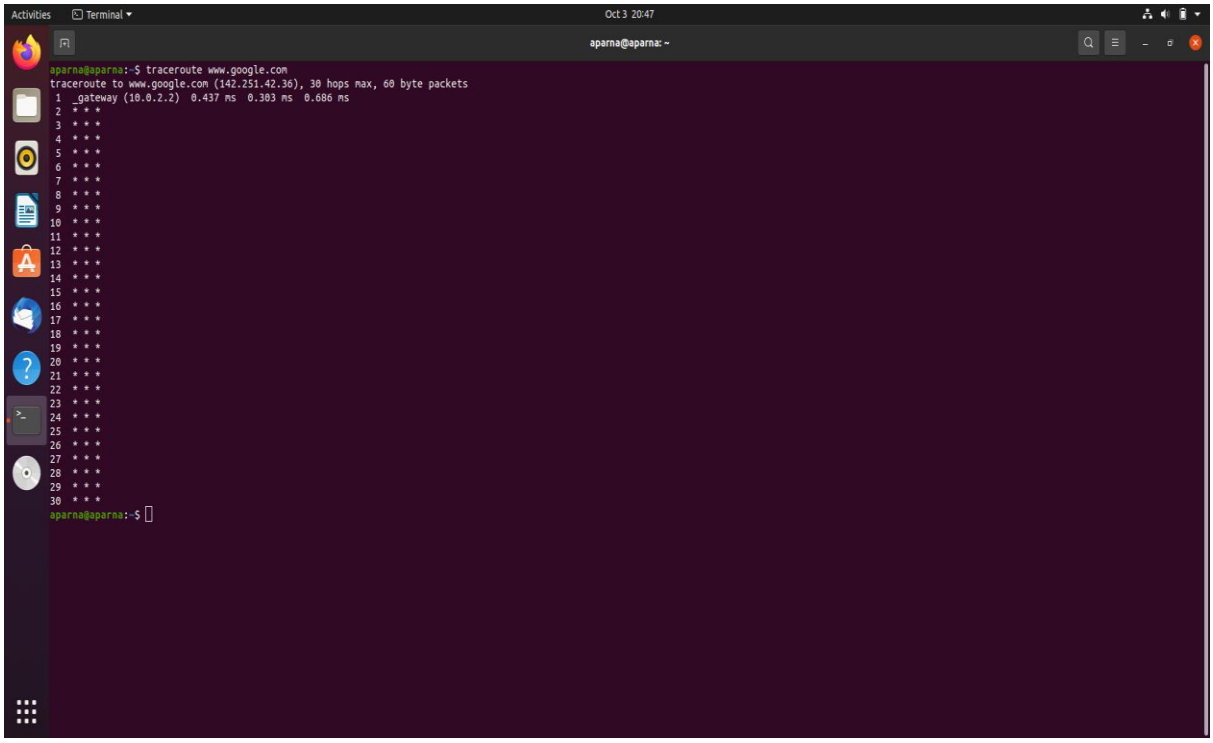
```
route [-f] [-p] [command [destination] [mask subnetmask]
[gateway] [metric costmetric]]
```

This section explains each of the options that you can use with the route command.

 The -f option clears the routing tables of all gateway entries. If you use the - f option in conjunction with one of the commands, the tables are cleared before you run the command.

 By default, routes are not preserved when you restart the system. Use the - p option with the add command to make a route persistent. Use the -p option with the print command to view the list of registered persistent routes.

Traceroute: Traceroute command in Linux prints the route that a packet takes to reach the host. This command is useful when you want to know about the route and about all the hops that a packet takes.



The first column corresponds to the hop count. The second column represents the address of that hop and after that, you see three space-separated time in milliseconds. *traceroute*

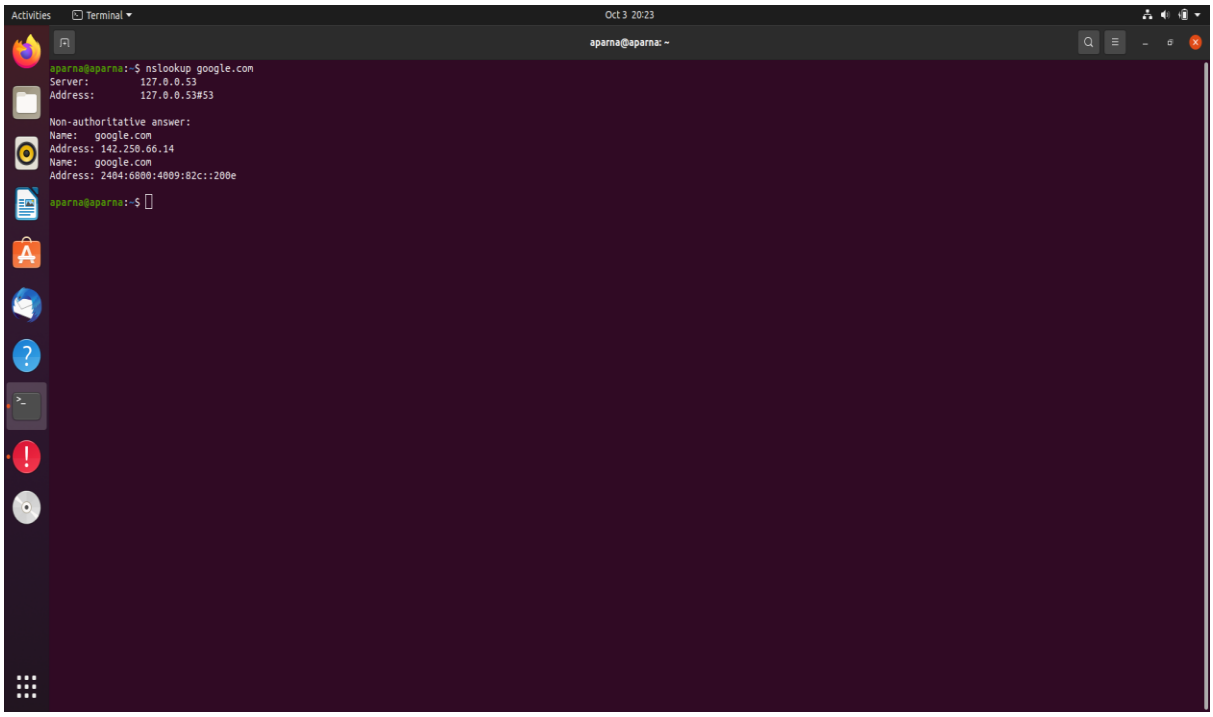
command sends three packets to the hop and each of the time refers to the time taken by the packet to reach the hop.

Syntax:
traceroute [options] host_Address [pathlength]

Options:
• -4 Option: Use ip version 4 i.e. use Ipv4
• -6 Option: Use ip version 6 i.e. use Ipv6
• -F Option: Do not fragment packet.

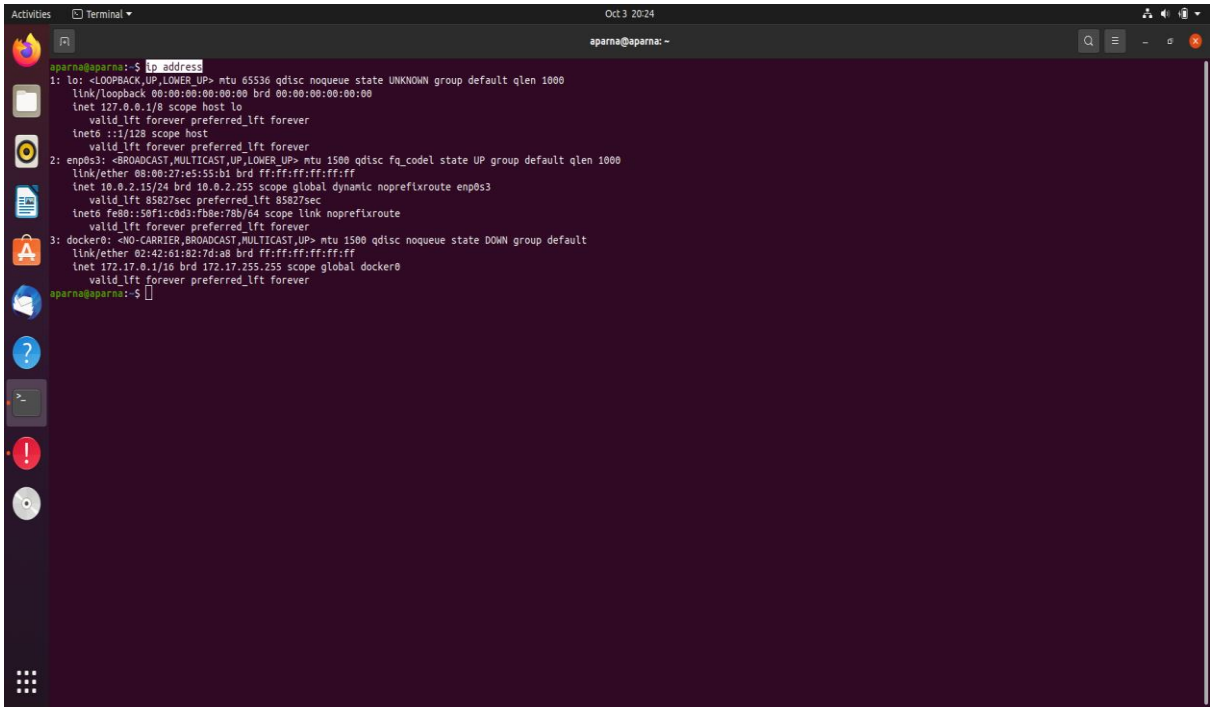
Nslookup: nslookup (stands for "Name Server Lookup") is a useful command for getting information from DNS server. It is a network administration tool for querying the Domain Name System (DNS) to obtain domain name or IP address mapping or any other specific DNS record. It is also used to troubleshoot DNS related problems.

Syntax: *nslookup [option]*



ip: ip command in Linux is present in the net-tools which is used for performing several network administration tasks. IP stands for Internet Protocol. This command is used to show or manipulate routing, devices, and tunnels. It is similar to *ifconfig* command but it is much more powerful with more functions and facilities attached to it. *ifconfig* is one of the deprecated commands in the net-tools of Linux that has not been maintained for many years. ip command is used to perform several tasks like assigning an address to a network interface or configuring network interface parameters.
It can perform several other tasks like configuring and modifying the default and static routing, setting up tunnel over IP, listing IP addresses and property information, modifying the status of the interface, assigning, deleting and setting up IP addresses and routes.

Syntax:
ip [OPTIONS] OBJECT { COMMAND | help }

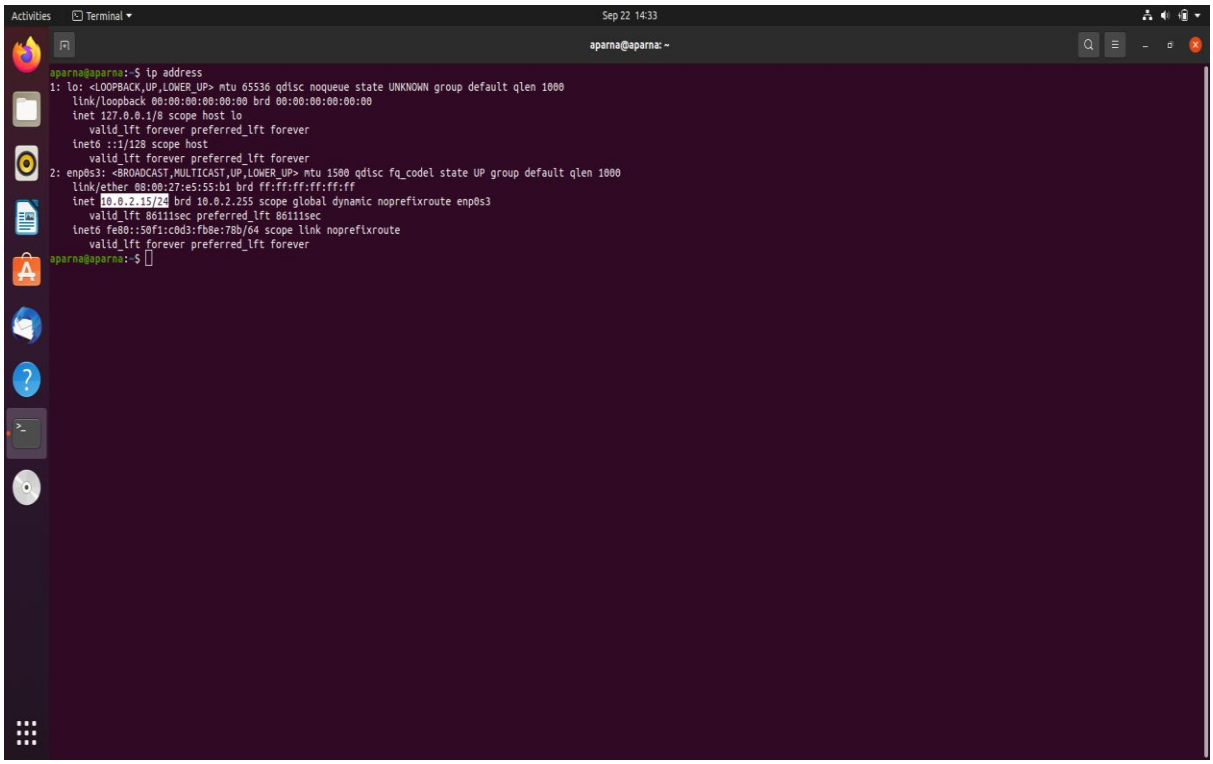


This will show the information related to all interfaces available on our system, but if we want to view the information of any particular interface, add the options show followed by the name of the particular network interface.

Options:
• -address: This option is used to show all IP addresses associated on all network devices.
• -link: It is used to display link layer information, it will fetch characteristics of the link layer devices currently available. Any networking device which has a driver loaded can be classified as an available device.

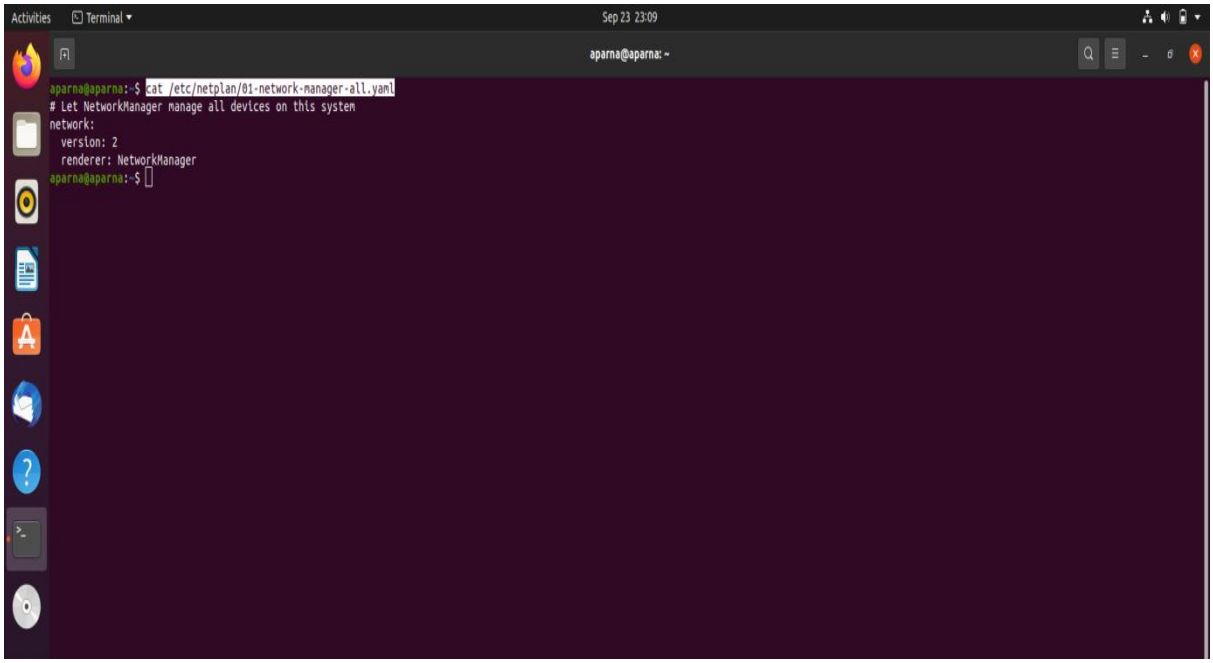
Setting up static IP addresses

- Step 1 : List all the interfaces in the system.Use the *ip address* command to define a static IP address on an interface.

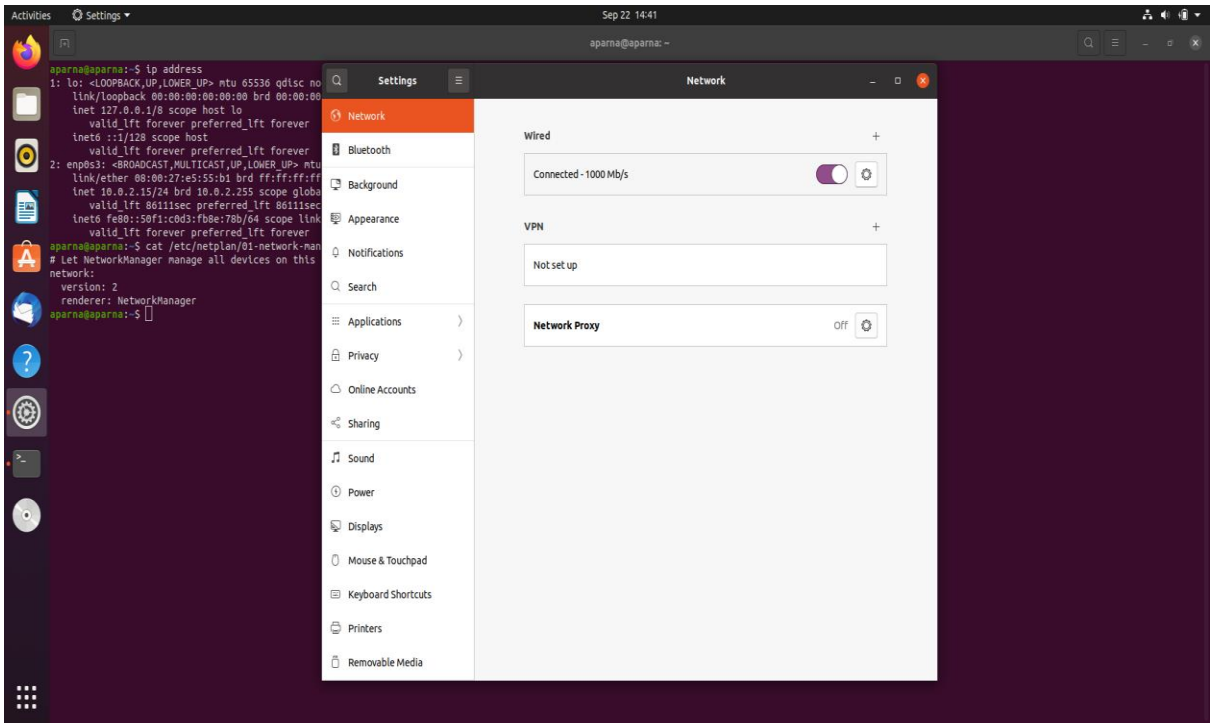


- Step 2 : To view the content of Netplan network configuration file, run the following command:

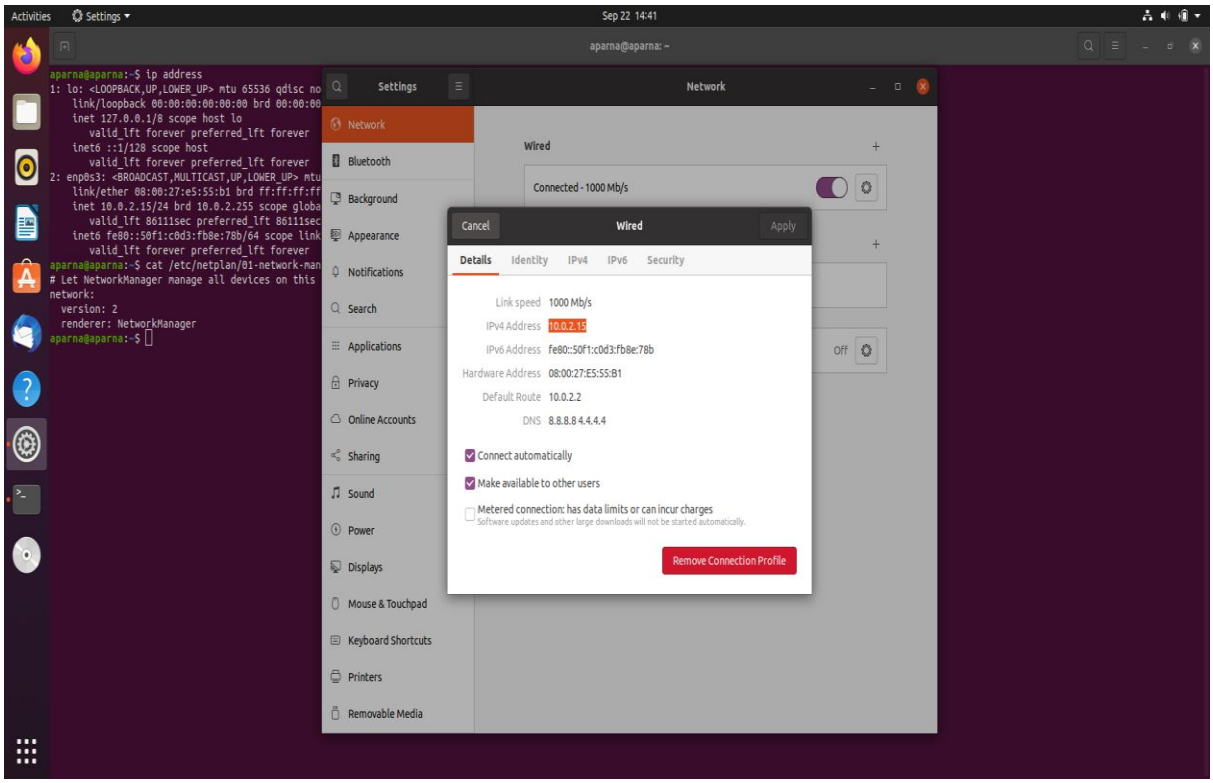
  cat /etc/netplan/01-network-manager-all.yaml



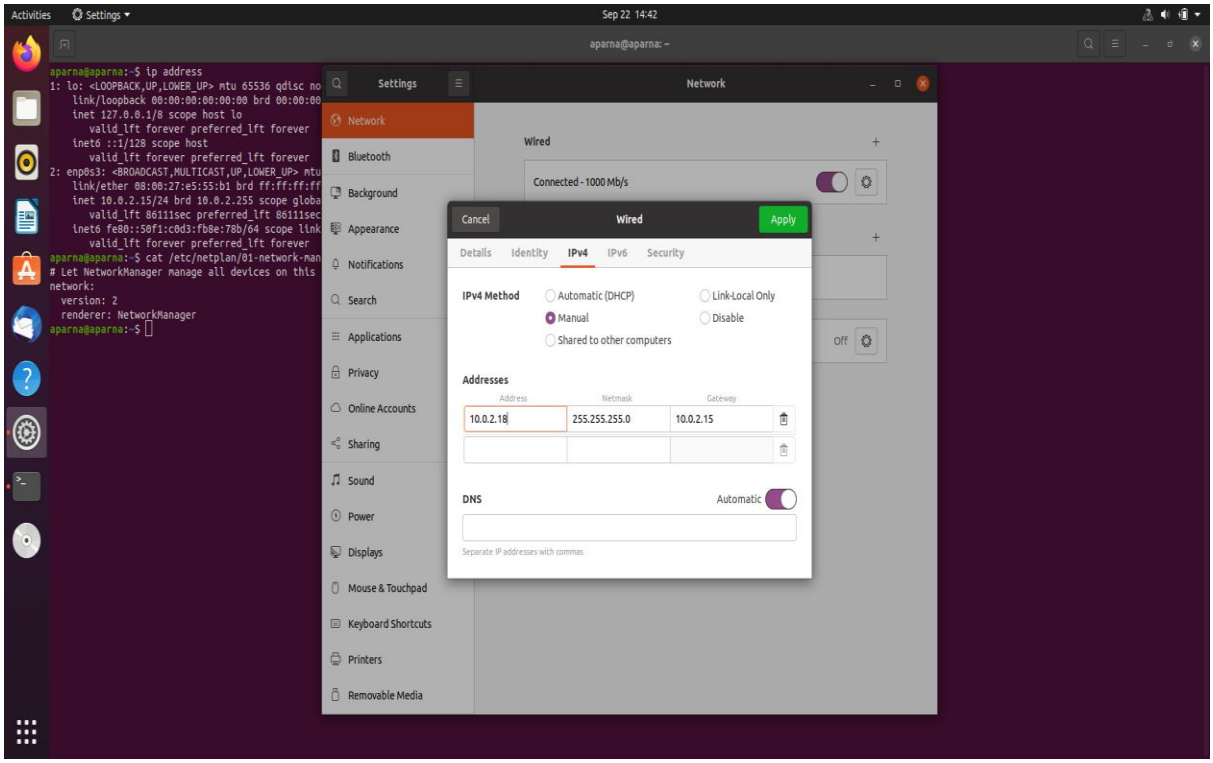
- Step 3 : Click on the top right network icon and select settings of the network interface you wish to configure to use a static IP address on Ubuntu.



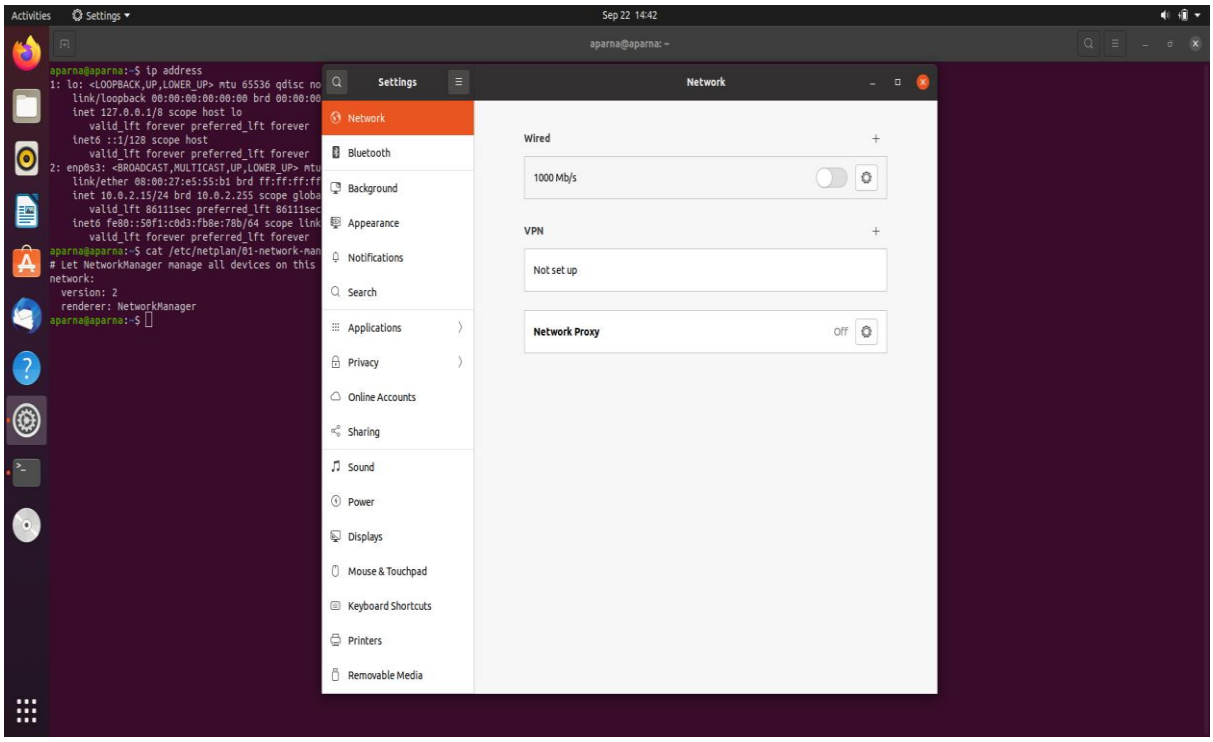
Click on the settings icon to start IP address configuration.



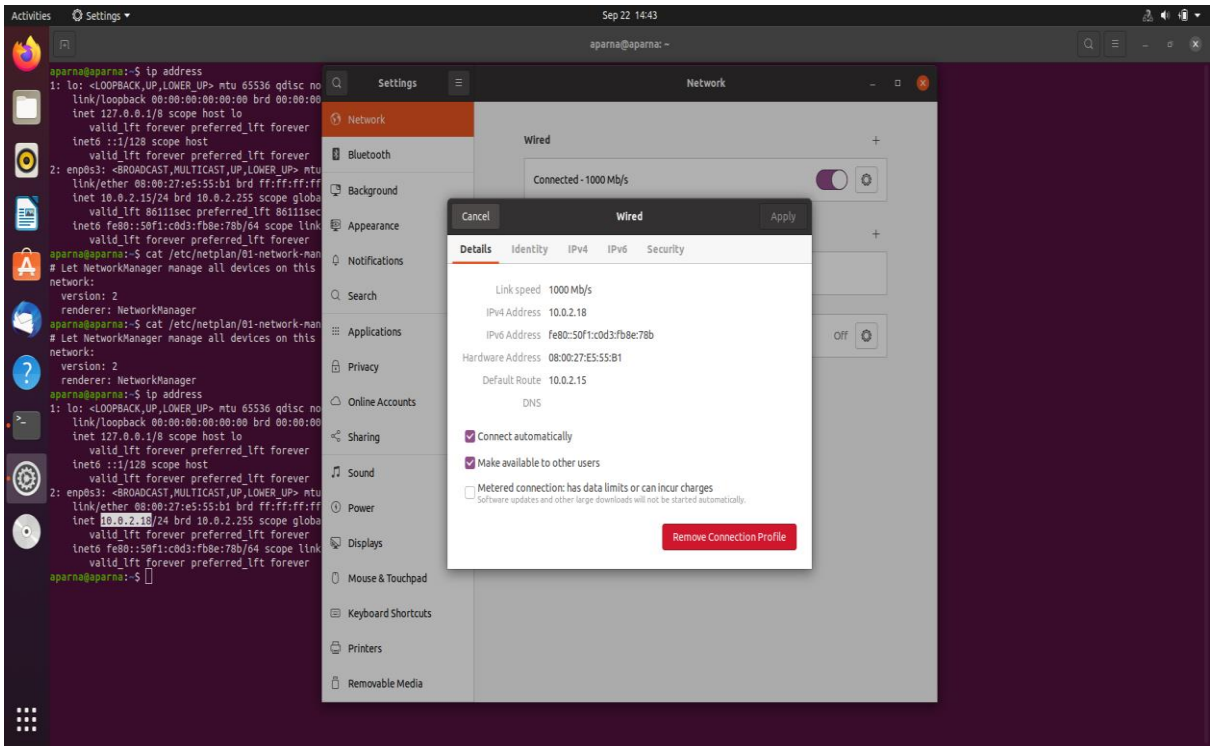
- Step 4 : Select IPv4 tab.Select manual and enter your desired IP address, netmask, gateway and DNS settings. Once ready click Apply button.



Turn OFF and ON switch to apply your new network static IP configuration settings.



- Step 5 : Run the command `ip address` and click on the network settings icon once again to confirm your new static IP address settings.



Subnet:

A subnet, or subnetwork, is a network inside a network. Subnets make networks more efficient. Through subnetting, network traffic can travel a shorter distance without passing through unnecessary routers to reach its destination.Organizations will use a subnet to subdivide large networks into smaller, more efficient subnetworks. One goal of a subnet is to split a large network into a grouping of smaller, interconnected networks to help minimize traffic. This way, traffic doesn't have to flow through unnecessary routs, increasing network speeds.

CIDR address scheme:

CIDR, which stands for Classless Inter-Domain Routing, is an IP addressing scheme that improves the allocation of IP addresses. It replaces the old system based on classes A, B, and C. This scheme also helped greatly extend the life of IPv4 as well as slow the growth of routing tables.

CIDR is based on variable-length subnet masking (VLSM). This allows it to define prefixes of arbitrary lengths making it much more efficient than the old system. CIDR IP addresses are composed of two sets of numbers. The network address is written as a prefix, like you would see a normal IP address (e.g. 192.255.255.255). The second part is the suffix which indicates how many bits are in the entire address (e.g. /12). Putting it together, a CIDR IP address would look like the following:

192.255.255.255/12

The network prefix is also specified as part of the IP address. This varies depending upon the number of bits required. Therefore, taking the example above, we can say that the first 12 bits are the network part of the address while the last 20 bits are for host addresses.

Subnet mask:

A subnet mask is like an IP address, but for only internal usage within a network. Routers use subnet masks to route data packets to the right place. Subnet masks are not indicated within data packets traversing the Internet — those packets only indicate the destination IP address, which a router will match with a subnet.

Iptables:

iptables is a user-space utility program that allows a system administrator to configure the IP packet filter rules of the Linux kernel firewall, implemented as different Netfilter modules. The filters are organized in different tables, which contain chains of rules for how to treat network traffic packets. Different kernel modules and programs are currently used for different protocols; *iptables* applies to

IPv4, *ip6tables* to IPv6, *arptables* to ARP, and *ebtables* to Ethernet frames.

•Tables is the name for a set of chains.

•Chain is a collection of rules.

•Rule is condition used to match packet.

•Target is action taken when a possible rule matches. Examples of the target areACCEPT, DROP, QUEUE.

•Policy is the default action taken in case of no match with the inbuilt chains and can be ACCEPT or DROP.

Syntax:

```
iptables --table TABLE -A/-C/-D... CHAIN rule --jump Target
```

TABLE

There are five possible tables:

•**filter:** Default used table for packet filtering. It includes chains like INPUT, OUTPUT and FORWARD.

•**nat :** Related to Network Address Translation. It includes PREROUTING and POSTROUTING chains.

•**mangle :** For specialised packet alteration. Inbuilt chains include PREROUTING and OUTPUT.

•**raw :** Configures exemptions from connection tracking. Built-in chains are PREROUTING and OUTPUT.

•**security :** Used for Mandatory Access Control CHAINS

There are few built-in chains that are included in tables. They are:

•**INPUT :**set of rules for packets destined to localhost sockets.

•**FORWARD :**for packets routed through the device.

•**OUTPUT :**for locally generated packets, meant to be transmitted outside.

•**PREROUTING :**for modifying packets as they arrive.

•**POSTROUTING :**for modifying packets as they are leaving.

Configure and Set Up a Firewall on Ubuntu

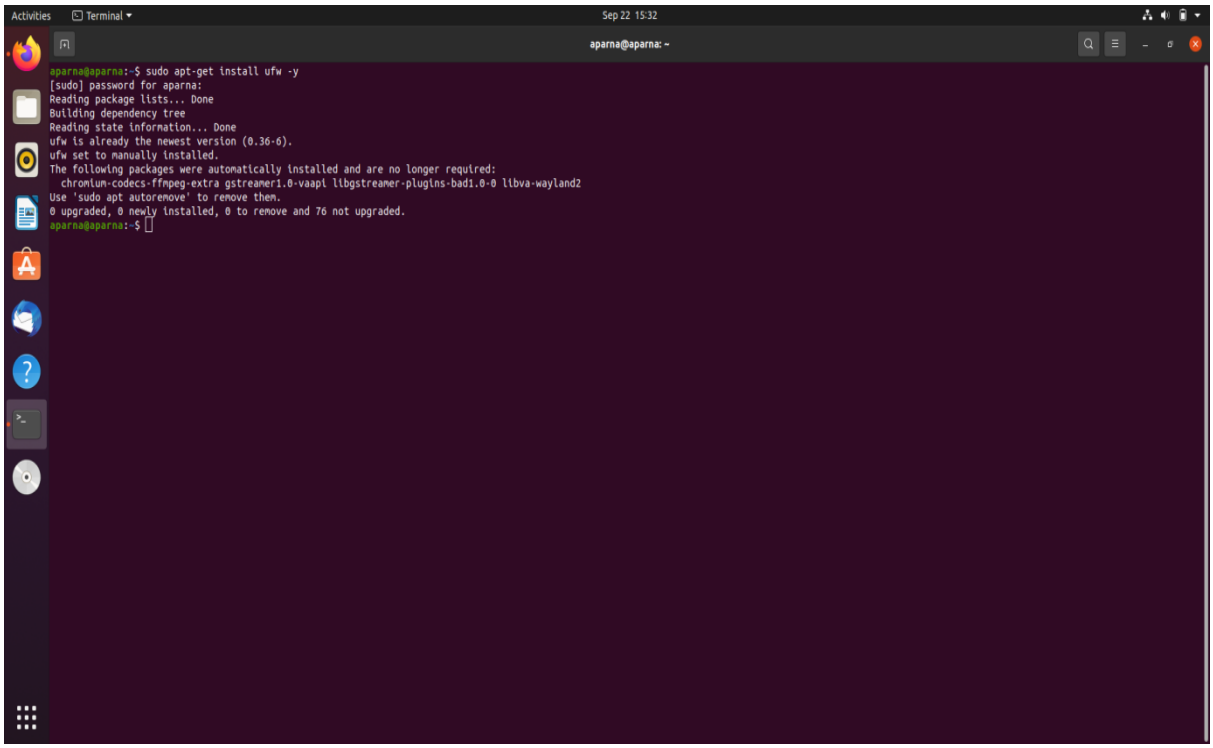
UFW stands for Uncomplicated Firewall which acts as an interface to IPTABLES that simplifies the process of the configuration of firewalls it will be a very hard for a beginners to learns and configure the firewall rules where we will secure the network from unknown users are machines. UFW works on the policies we configure as rules.

- For this, we needed a non-root user with root permission on the machine.

Installing the UFW (Firewall)

UFW is installed by default with Ubuntu, if not installed then we will install them using the below command –

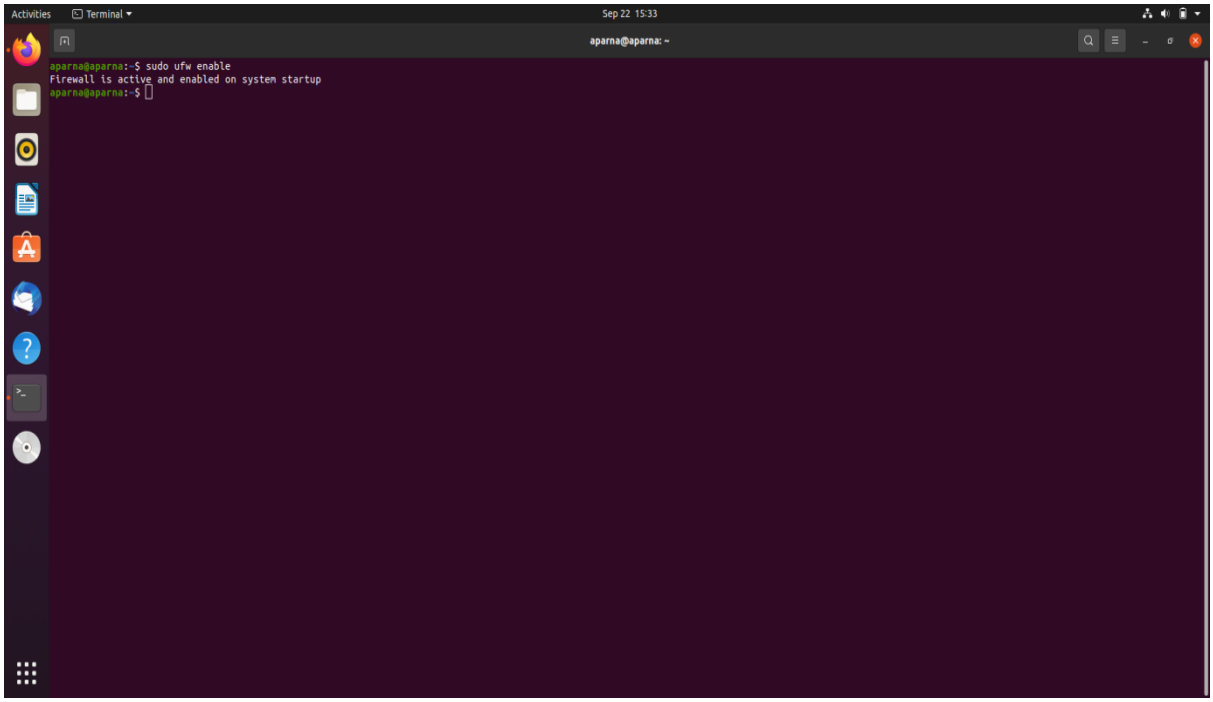
```
sudo apt-get install ufw -y
```



Enabling the UFW (Firewall)

Below is the command to enable the UFW –

```
sudo ufw enable
```

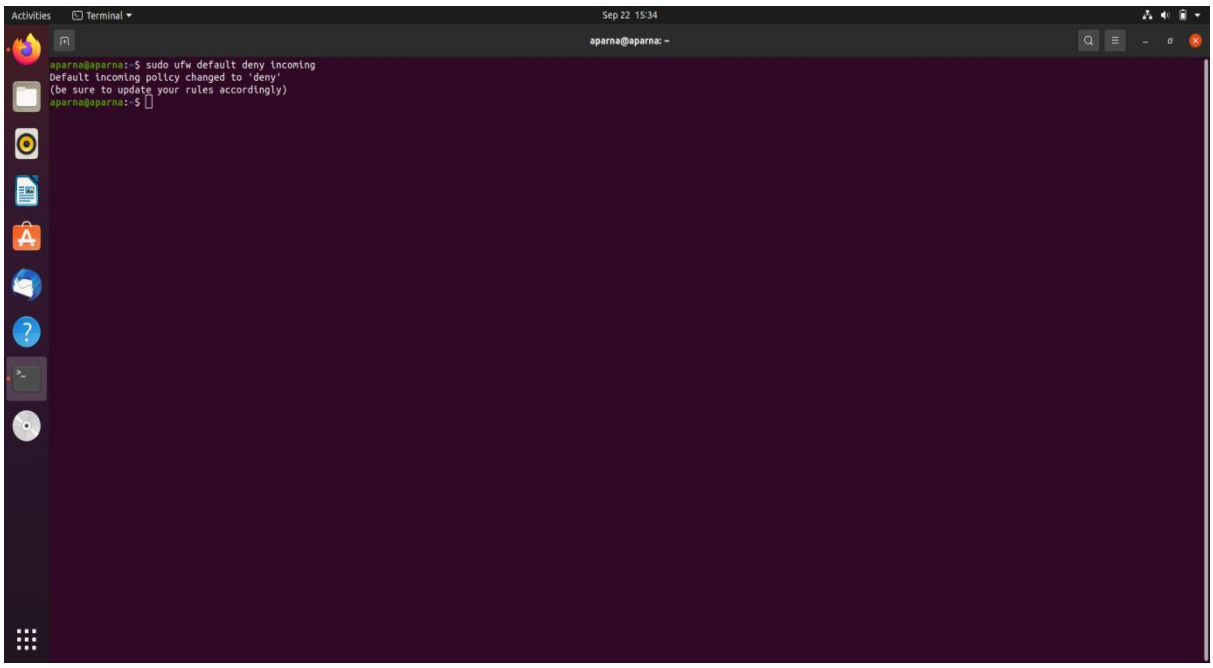


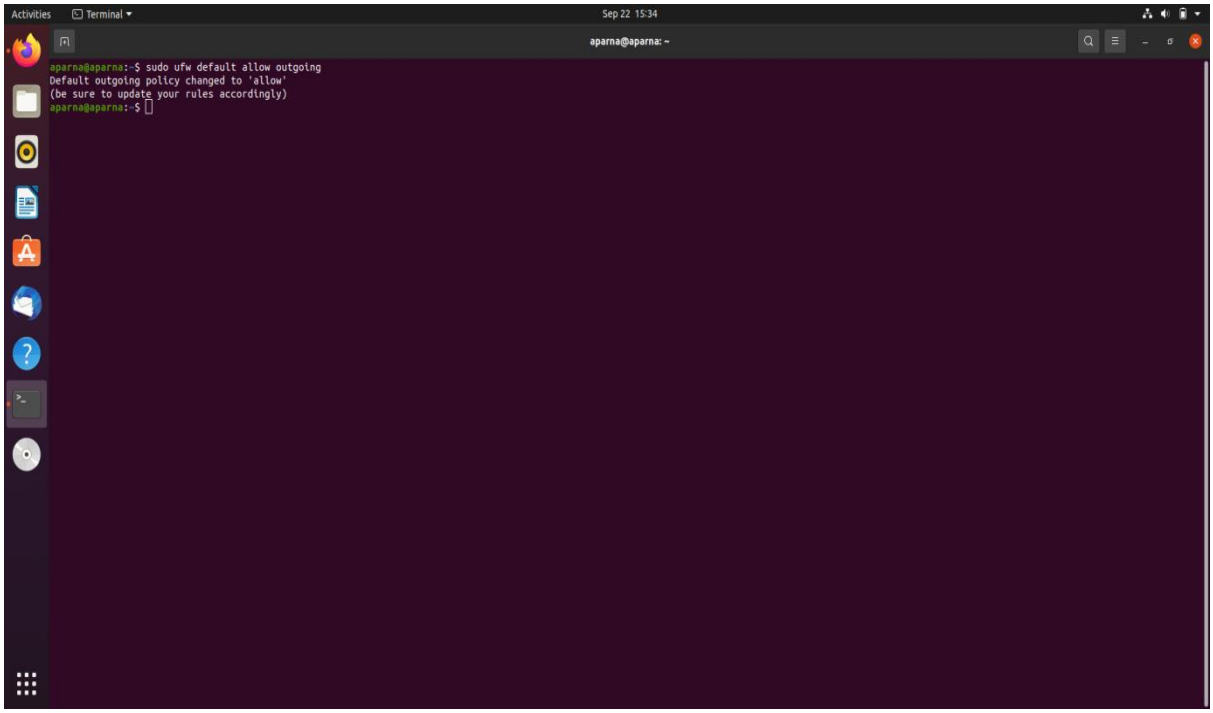
Enabling the Default Policies

As the beginner, we will first configure default policies, which control and handles the traffic which will not match the other rules. By default, the rules will deny all incoming connections and allow all outgoing connections will be allowed which stops someone trying to reach the machine from the internet world.

```
sudo ufw default deny incoming
```

```
sudo ufw default allow outgoing
```

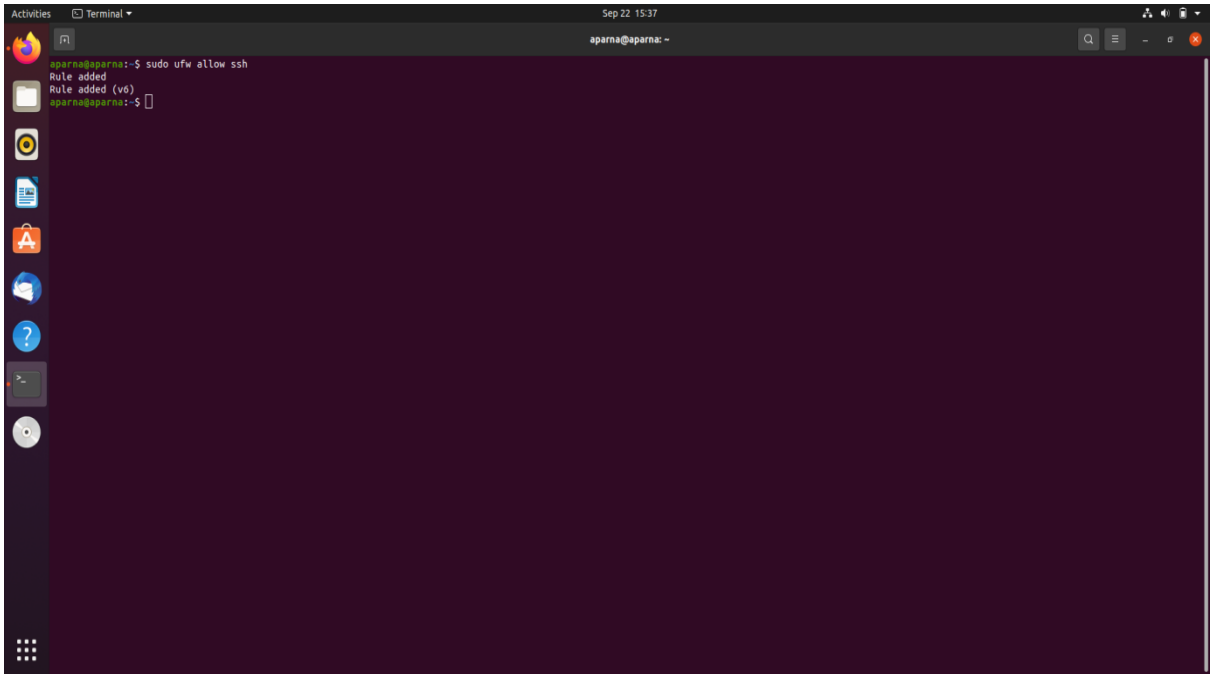




Enabling SSH Connections

Using the above commands, we have disabled all the incoming connections, it will deny all the incoming connections, we needed to create a rule which will explicitly allow the SSH incoming connection.Below is the command to enable the incoming connection for SSH.

```
sudo ufw allow ssh
```

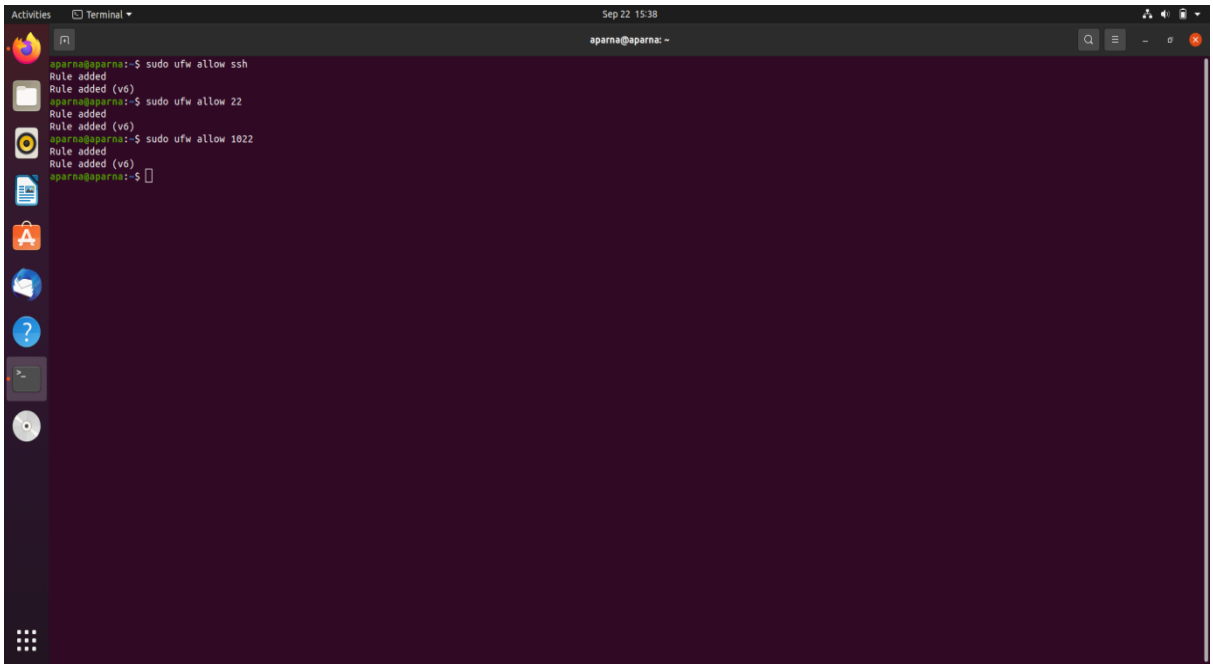


With the above command, the port 22 will be allowed for incoming connections. We can use the below command directly using the port no 22 to allow the SSH connections.

```
sudo ufw allow 22
```

However, if we have configured the SSH daemon to use a different port like 2022 or 1022, then we can use the below command –

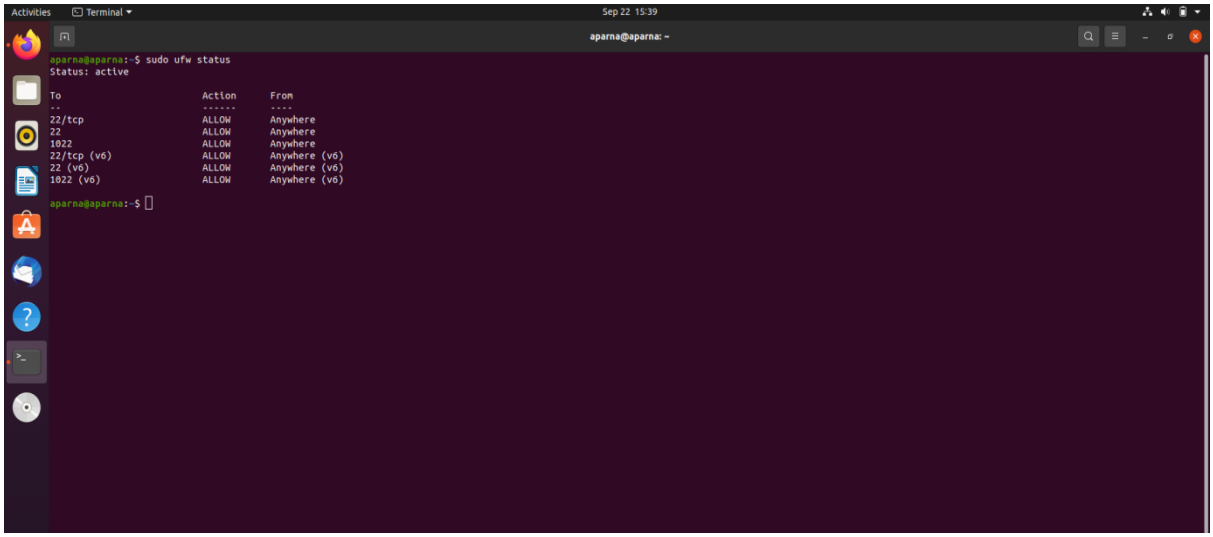
```
sudo ufw allow 1022
```



Checking the UFW (Firewall) Status

Below is the command to check the current status of the firewall rules.

```
sudo ufw status
```



Enabling the UFW for regular port like (HTTP, HTTPS & FTP)

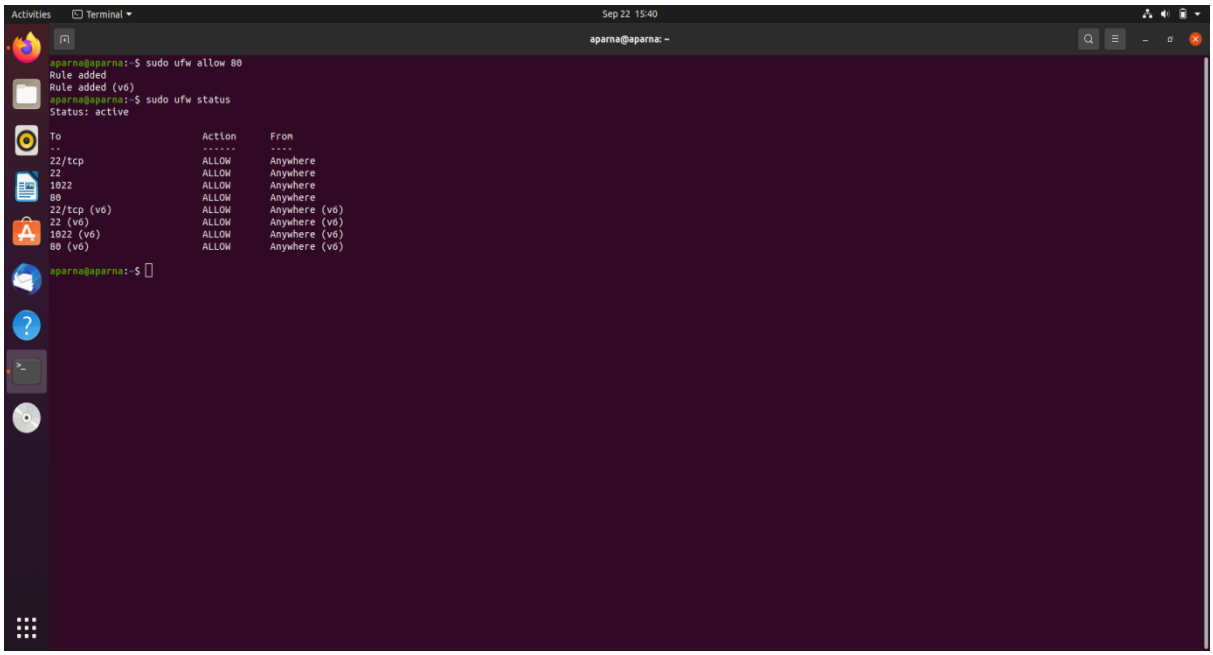
At this point, we will allow others to connect to the server for the regular ports like HTPP, HTTPS, and FTP ports respectively.

HTTP port 80

```
sudo ufw allow 80
```

We can check the UFW (Firewall) status using the below command

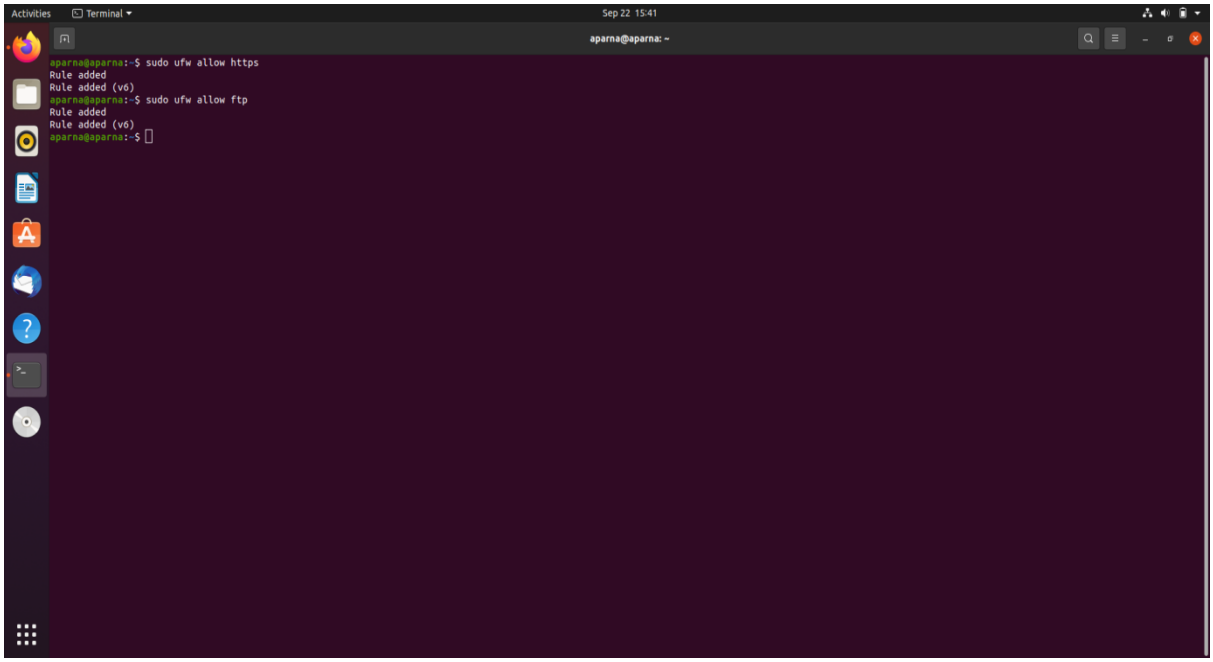
```
sudo ufw status
```



Like that will use the below command to enable HTTPs and FTP ports (443 and 21) respectively.

```
sudo ufw allow https
```

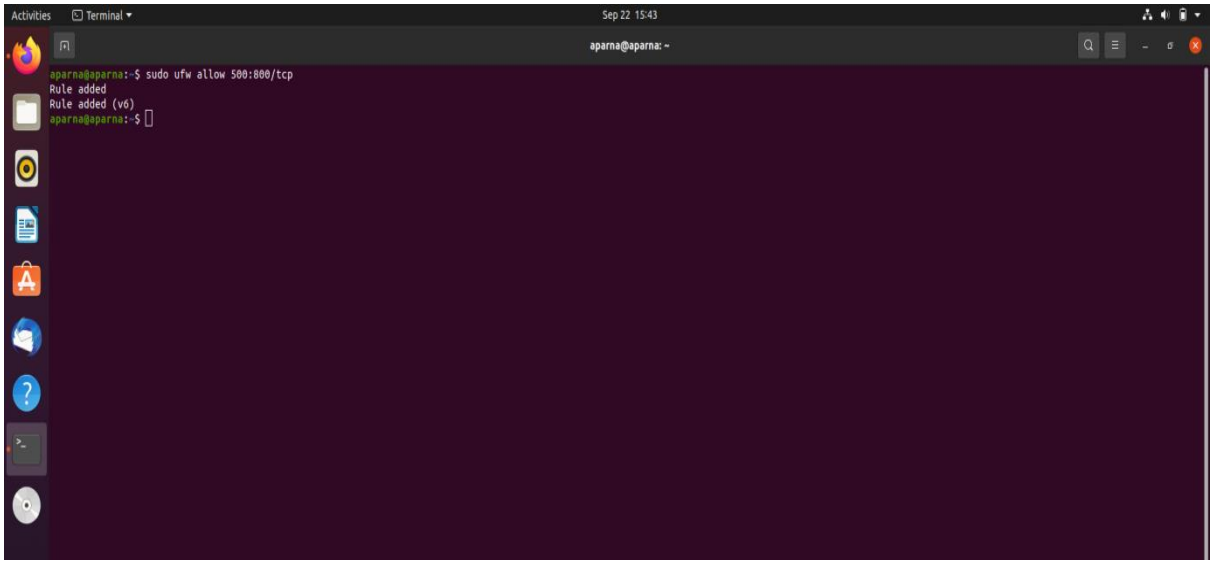
```
sudo ufw allow ftp
```



Enabling to Allow Specific Range of Ports

We can also allow or deny particular ranges of ports with UFW to allow the multiple ports instead of allowing single ports.Below is the command to enable a specific range of ports.

```
sudo ufw allow 500:800/tcp
```



Enable to Allow specific IP Addresses

If we want to allow a particular machine to allow for all the ports. We can use the below command.

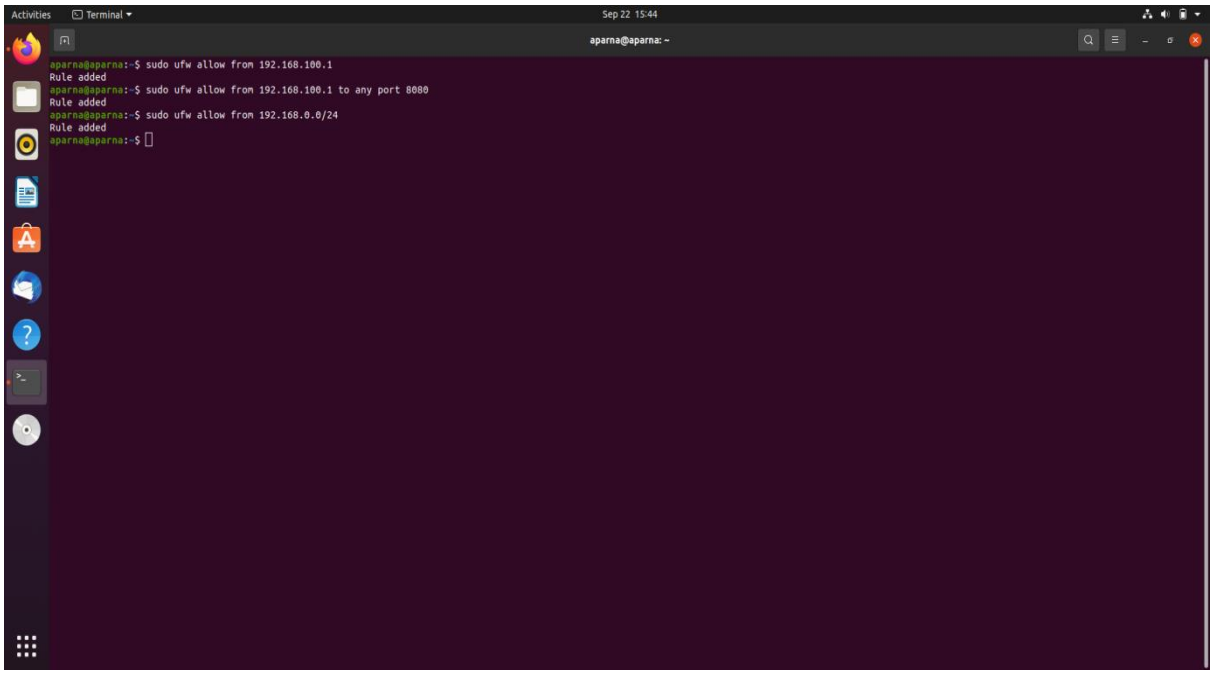
```
sudo ufw allow from 192.168.100.1
```

If we want to allow for only specific port we can use the below command.

sudo ufw allow from 192.168.100.1 to any port 8080

If we want to enable the specific subnets like we want to enable for office networks we can use the below command.

sudo ufw allow from 192.168.0.0/24



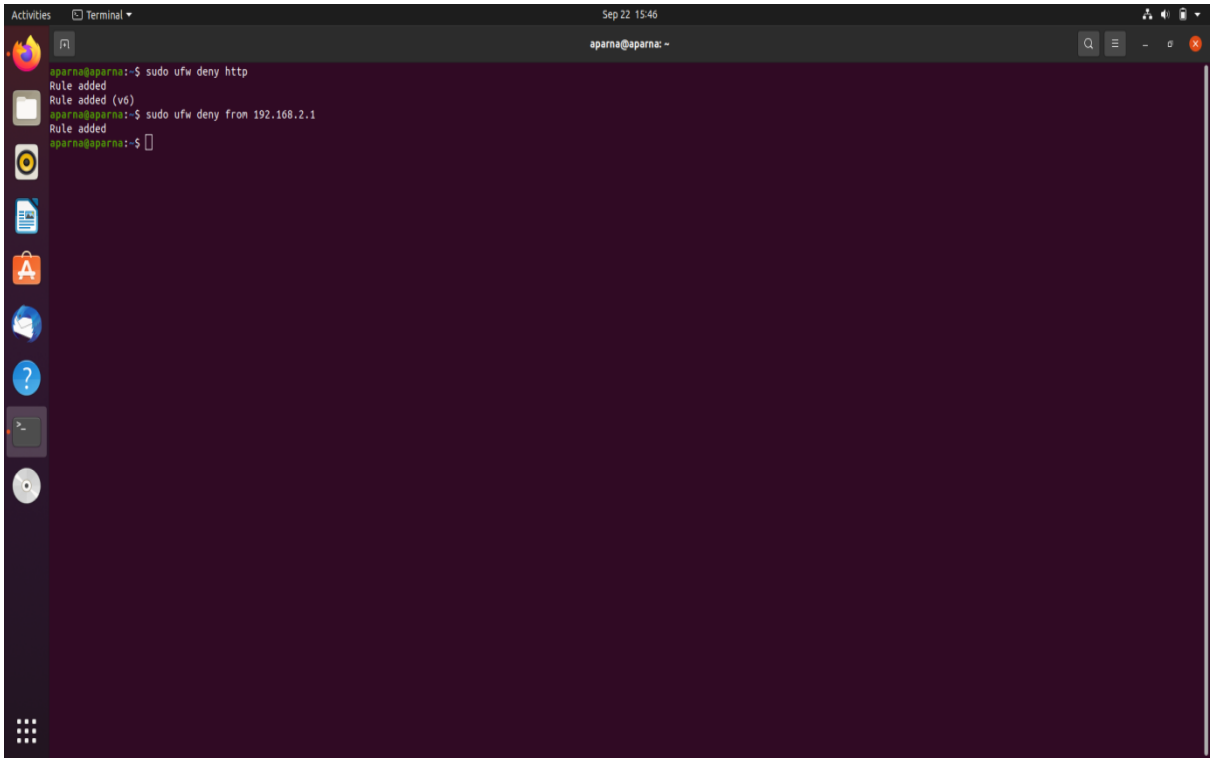
Deny the Connections or Rules

If we want to deny any ports or network we can use the below commands to deny the connections.

sudo ufw deny http

If we want to deny all the connects from a specific network we can use the below command.

sudo ufw deny from 192.168.2.1



Deleting the Rules

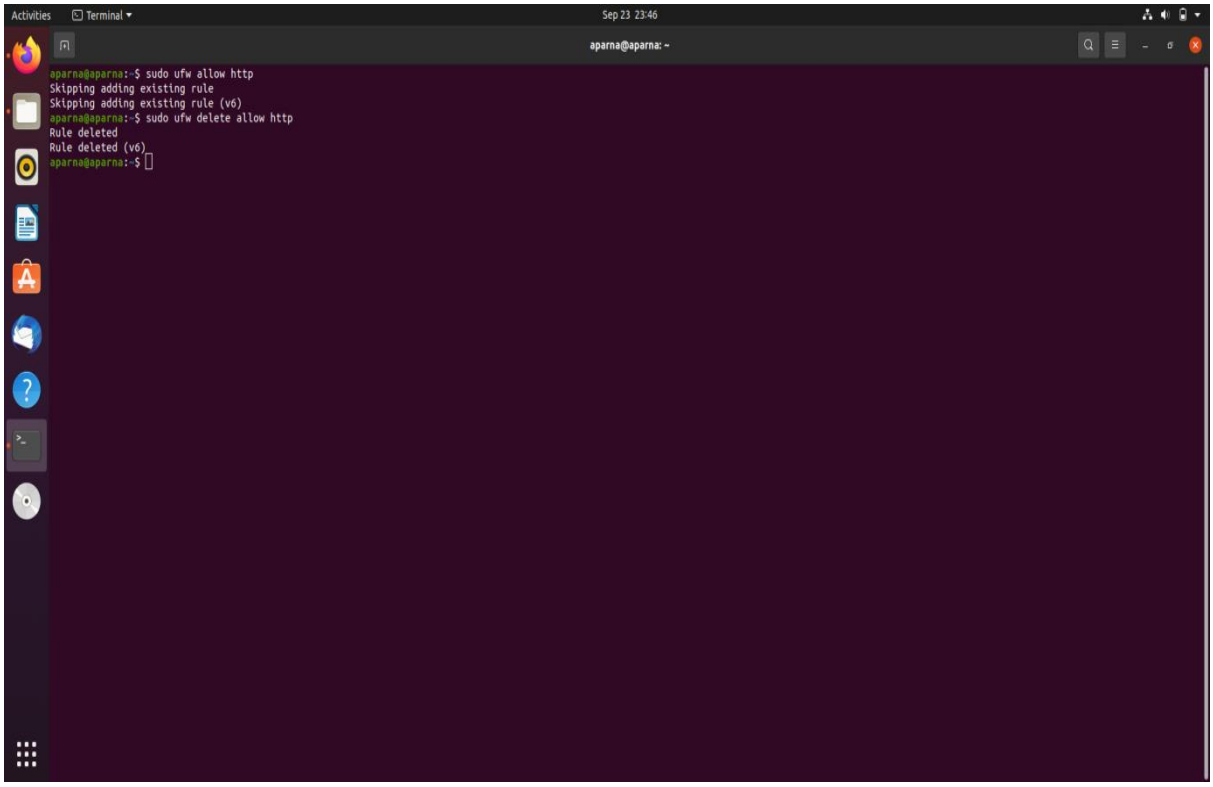
We can delete the rules in two ways one with the actual rules and other with the rules numbers.

Actual Rules

The rules can be deleted using the actual rule which we allowed using the allow command. Below is the command to delete the HTTP rules from UFW.

```
sudo ufw allow http
```

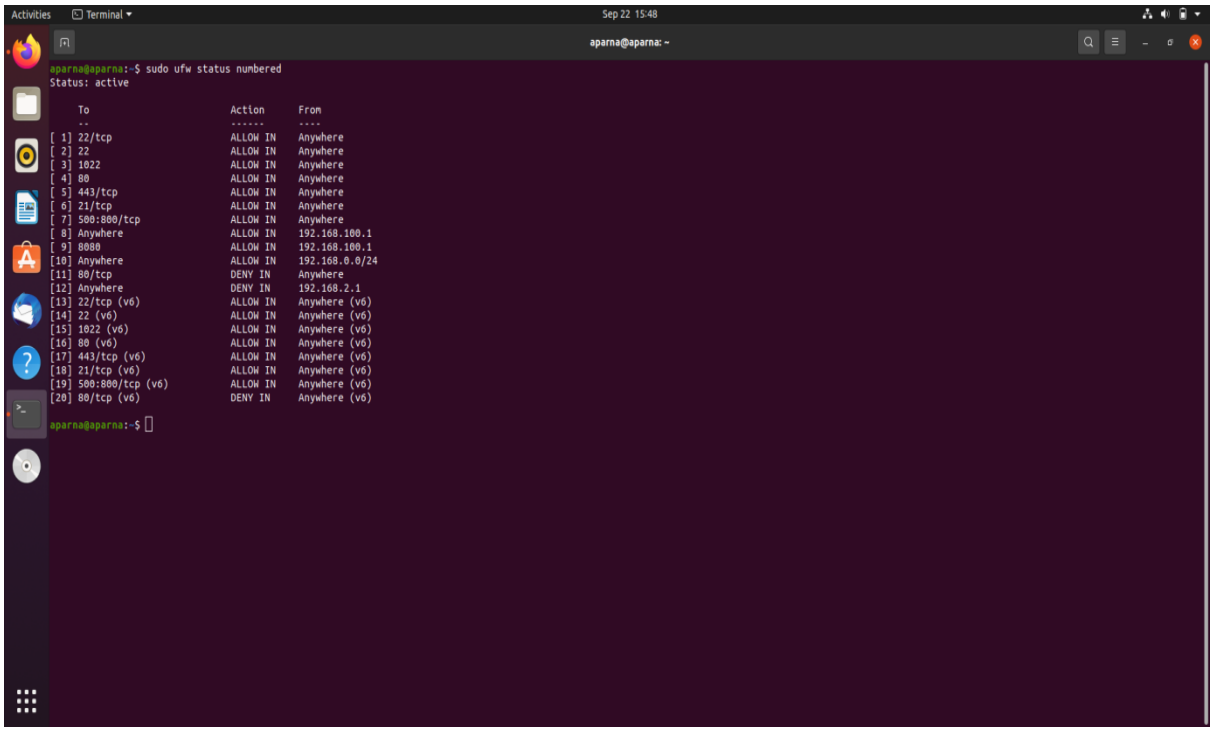
```
sudo ufw delete allow http
```



Rules Number

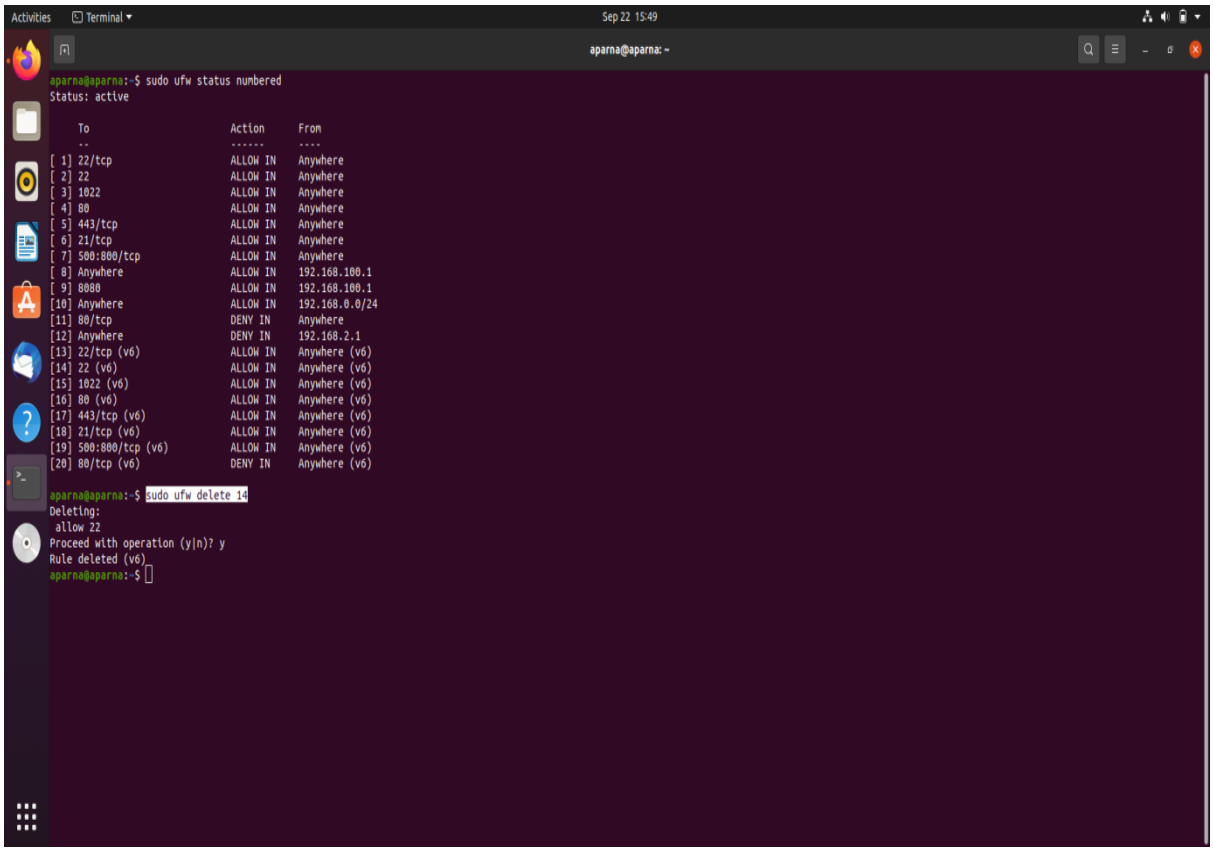
We can use the Rules numbers to delete the firewall rules, we can get the list of firewall rules with the below command.

sudo ufw status numbered



If we want to delete the rule 14, then we can use the below command to delete the rules with the below command.

```
sudo ufw delete 14
```

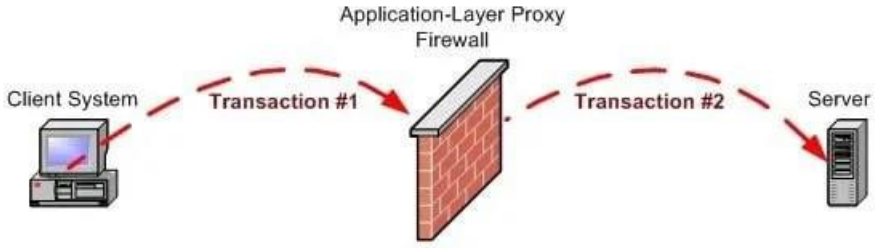


Application layer Proxies

Application Layer Proxy is any service or server that acts as a proxy for client computer requests at the application's protocols. For example, in Microsoft Proxy Server, the Web Proxy Service is an application layer proxy for the Hypertext Transfer Protocol (HTTP), Secure Hypertext Transfer Protocol (S-HTTP), File Transfer Protocol (FTP), and Gopher protocols. Application layer proxies provide security by hiding internal network addresses from the outside world.

Application layer proxies provide more support for the additional capabilities of each protocol than do circuit layer proxies. For example, application layer proxies can support virus scanning. Application layer proxies are also client-neutral and require no special software components or operating system on the client computer to enable the client to communicate with servers on the Internet using the proxy server.

Microsoft Proxy Server can grant users access to selected application layer protocols and can restrict access to remote Web sites by domain name, IP address, and subnet mask.



Transaction is split in two: to client, firewall appears to be the server (transaction #1); to server, firewall appears to be the client (transaction #2)

Experiment No. 9

Aim

Analyzing network packet stream using tcpdump and wireshark. Perform basic service tests using nc.

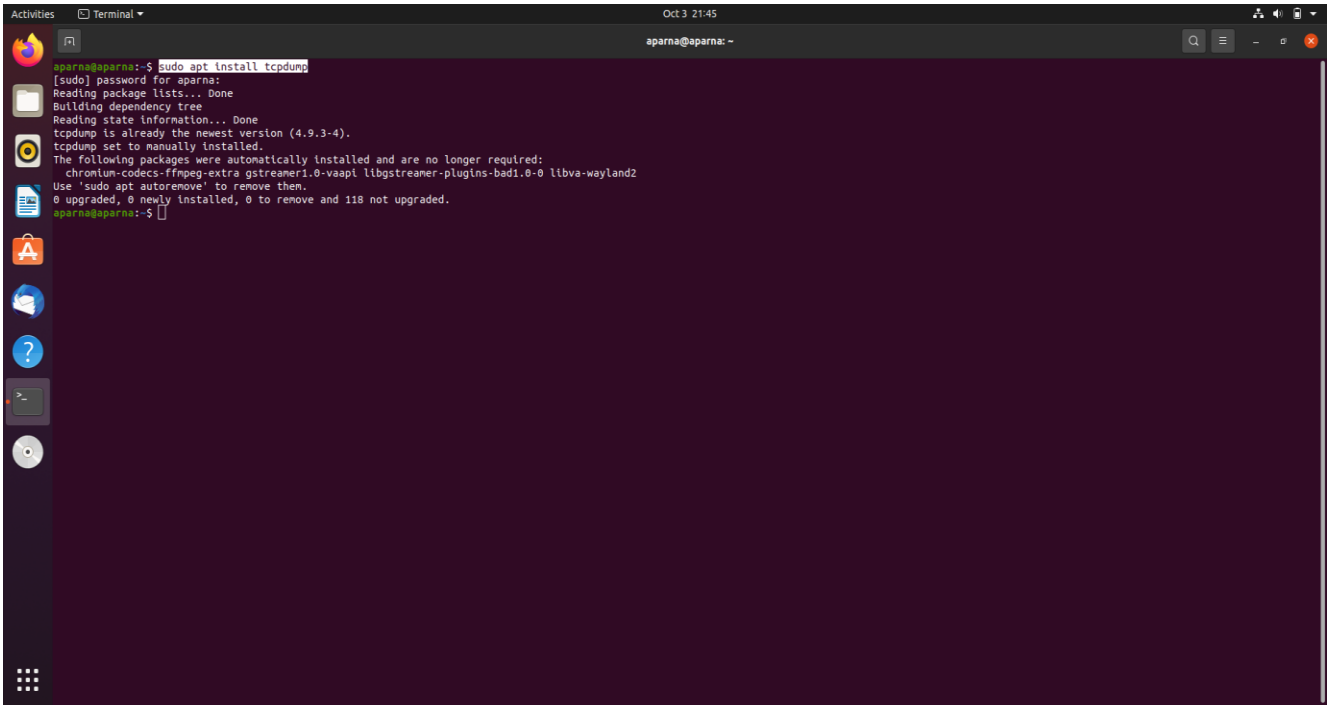
Result

Tcpdump

tcpdump is a packet sniffing and packet analyzing tool for a System Administrator to troubleshoot connectivity issues in Linux. It is used to capture, filter, and analyze network traffic such as TCP/IP packets going through your system. It is many times used as a security tool as well. It saves the captured information in a pcap file, these pcap files can then be opened through Wireshark or through the command tool itself.

Installing tcpdump tool in Linux

```
Sudo apt install tcpdump
```

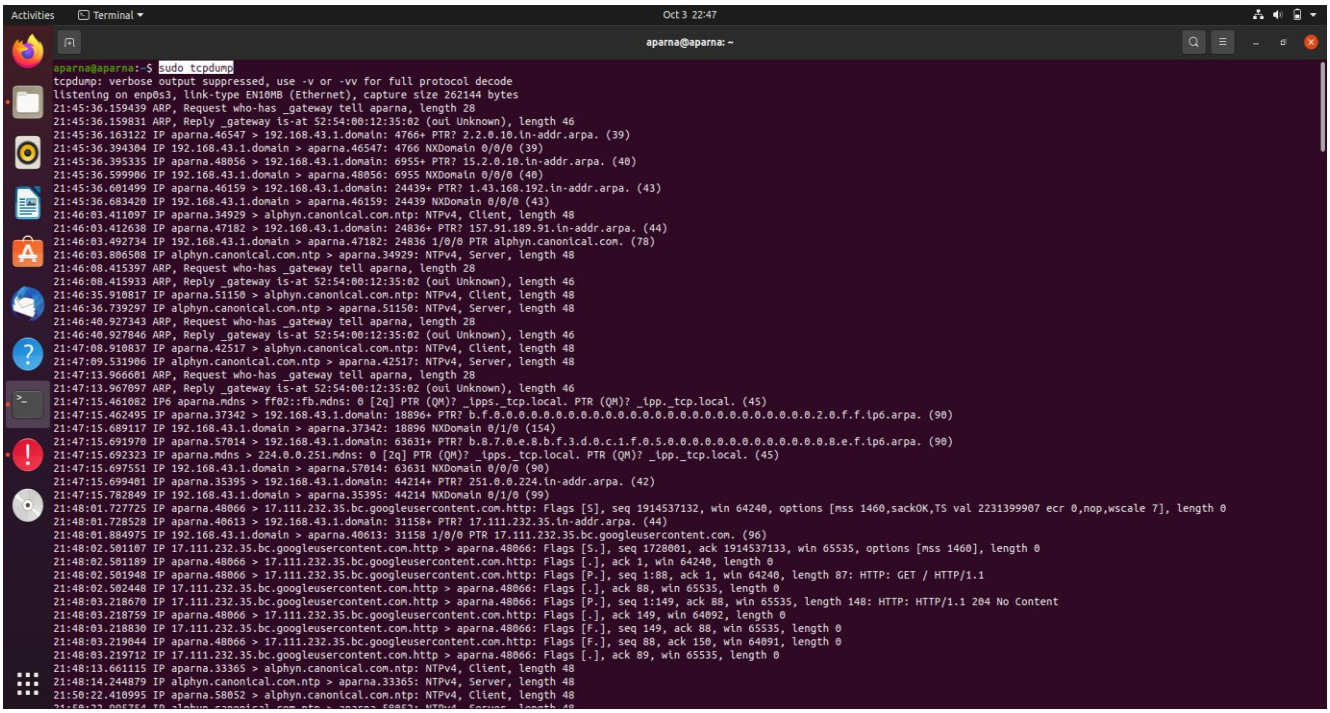


Working with tcpdump command

1. To capture the packets of current network interface

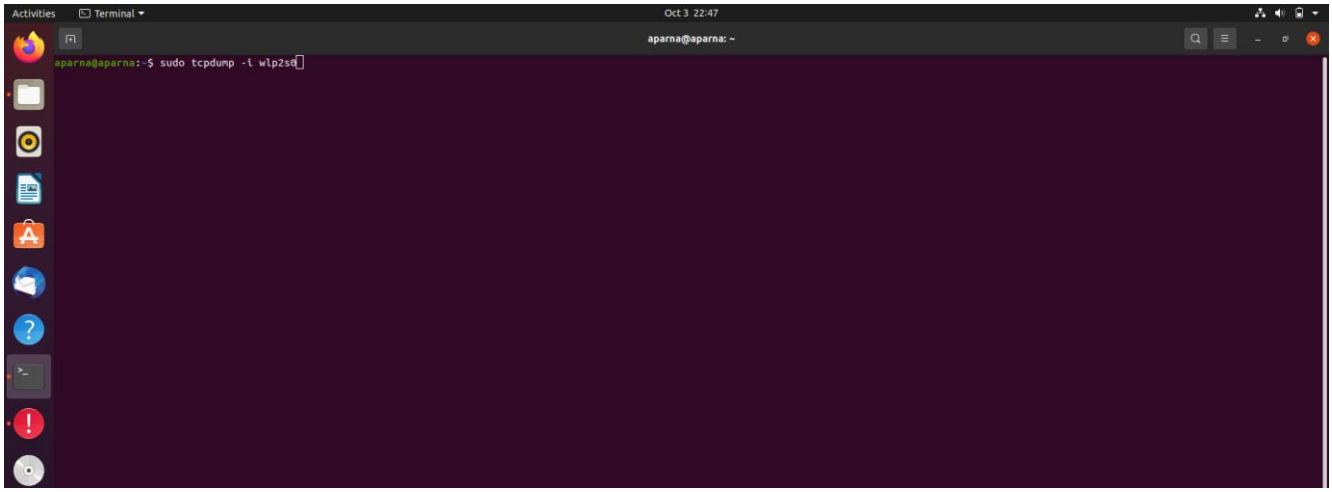
```
sudo tcpdump
```

This will capture the packets from the current interface of the network through which the system is connected to the internet.



1. To capture packets from a specific network interface

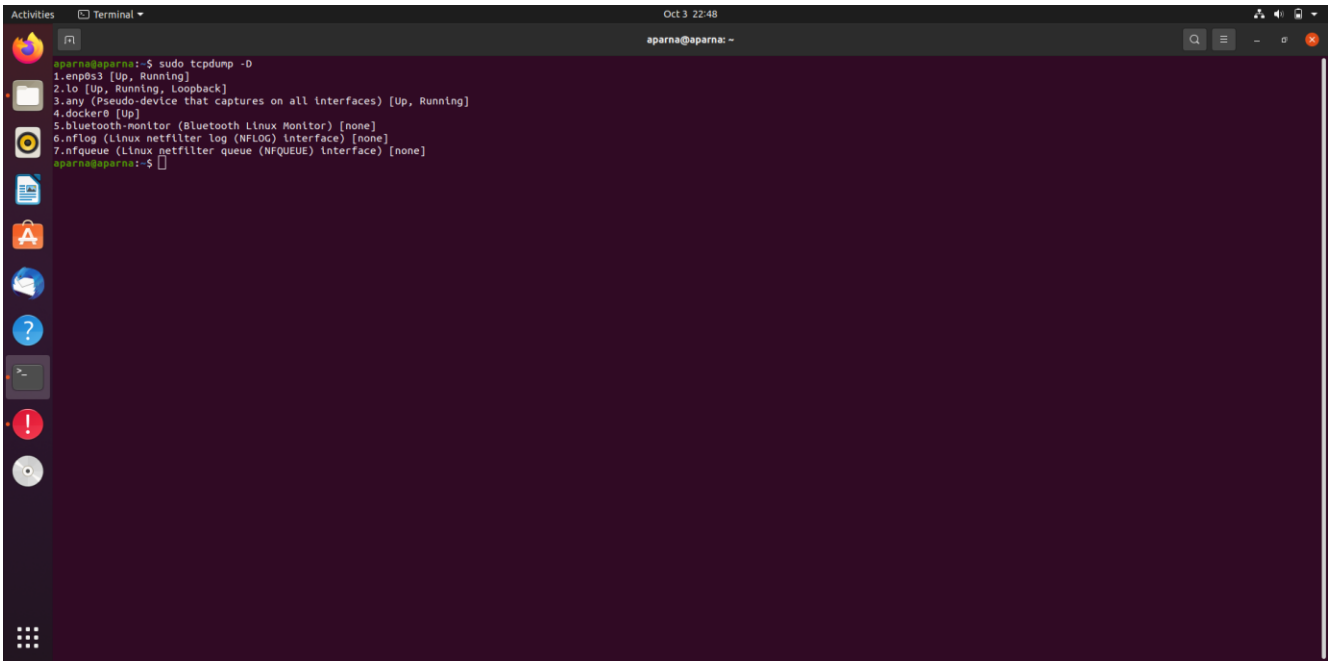
```
sudo tcpdump -i wlp2s0
```



This command will now capture the packets from wlp2s0 network interface.

3) To display all available interfaces

```
sudo tcpdump -D
```



Wireshark

Wireshark is a software tool used to monitor the network traffic through a network interface. It is the most widely used network monitoring tool today.

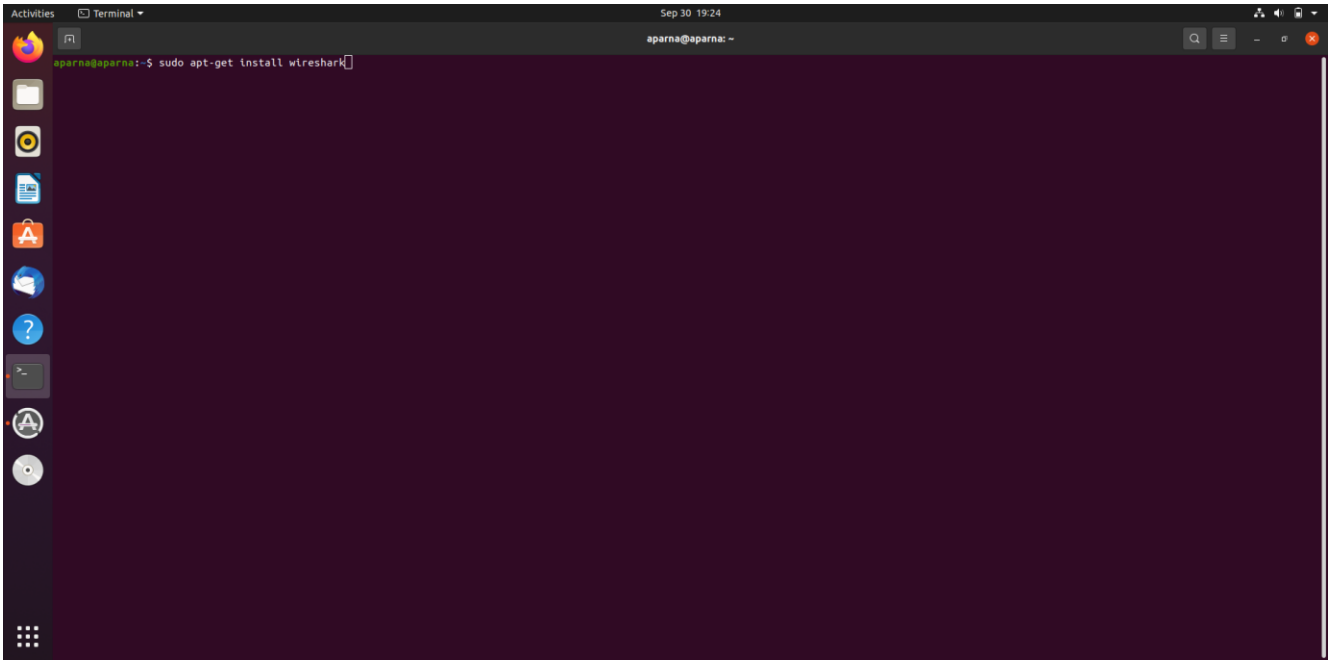
Wireshark is loved equally by system administrators, network engineers, network enthusiasts, network security professionals and black hat hackers. The extent of its popularity is such, that experience with Wireshark is considered as a valuable/essential trait in a computer networking related professional.

There are many reasons why Wireshark is so popular :

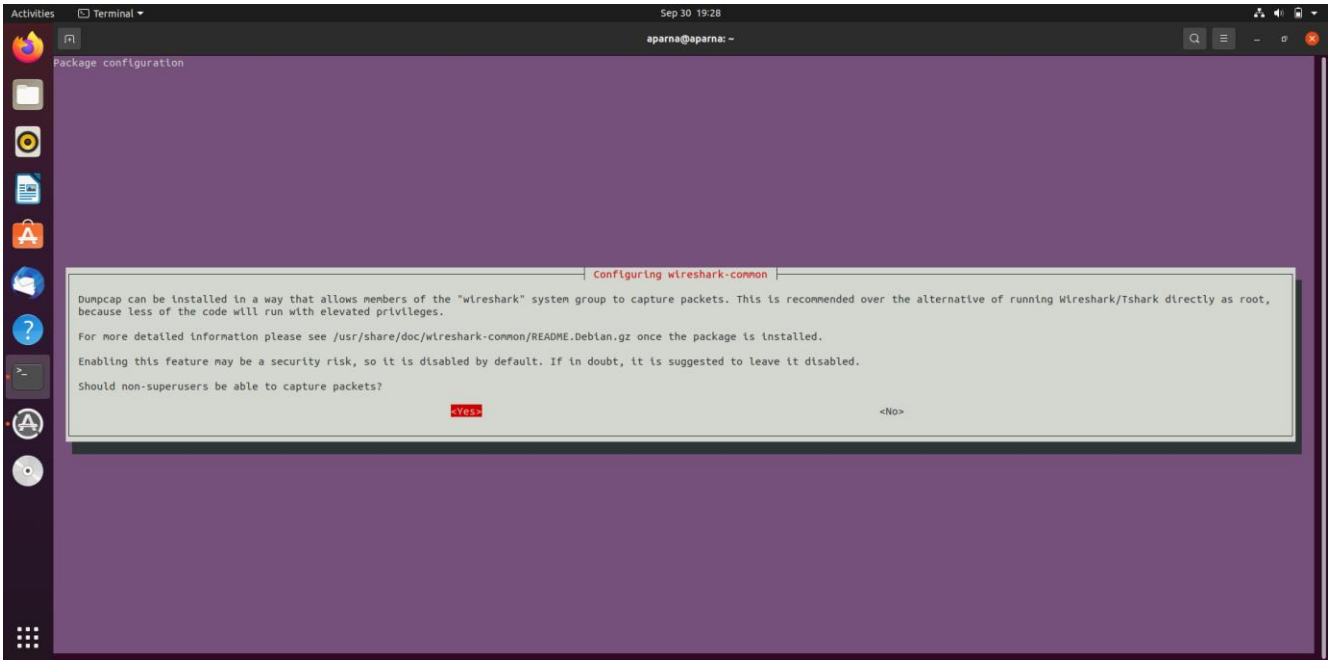
- It has a great GUI as well as a conventional CLI(T Shark).
- It offers network monitoring on almost all types of network standards (ethernet, wlan, Bluetooth etc)
- It is open source with a large community of backers and developers.
- All the necessary components for monitoring, analysing and documenting the network traffic are present. It is free to use.

Wireshark installation

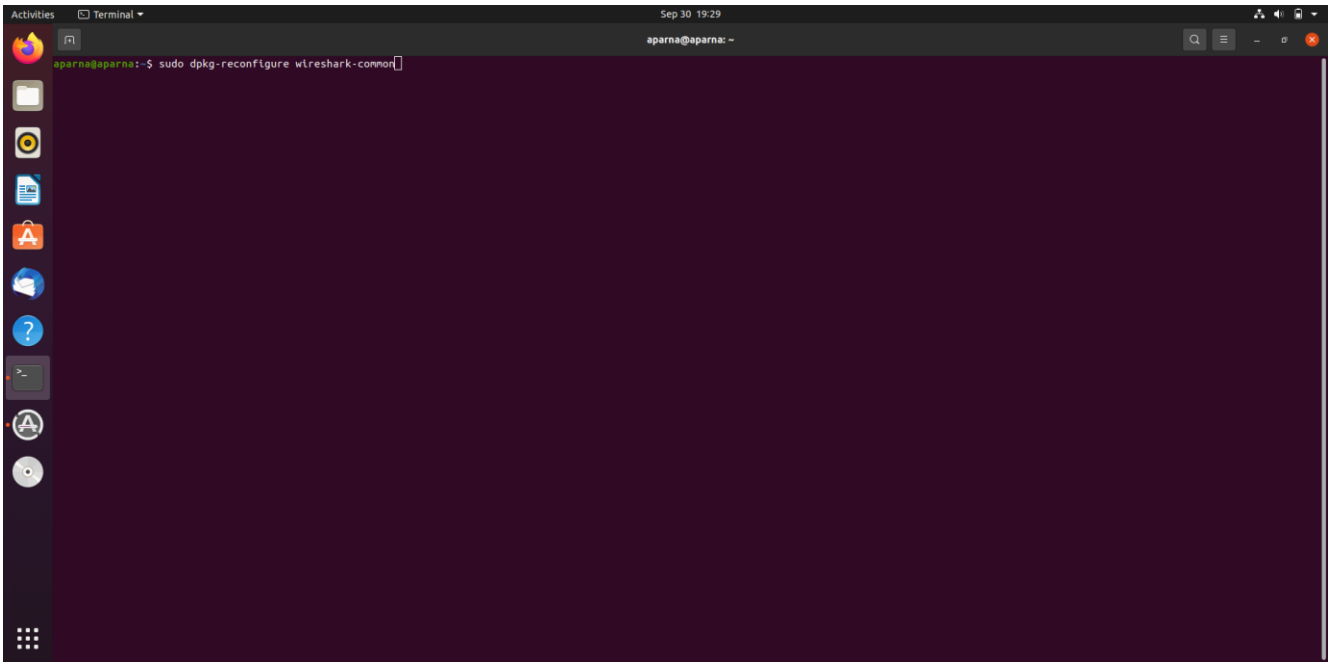
`sudo apt install wireshark` command is used to install wireshark in linux



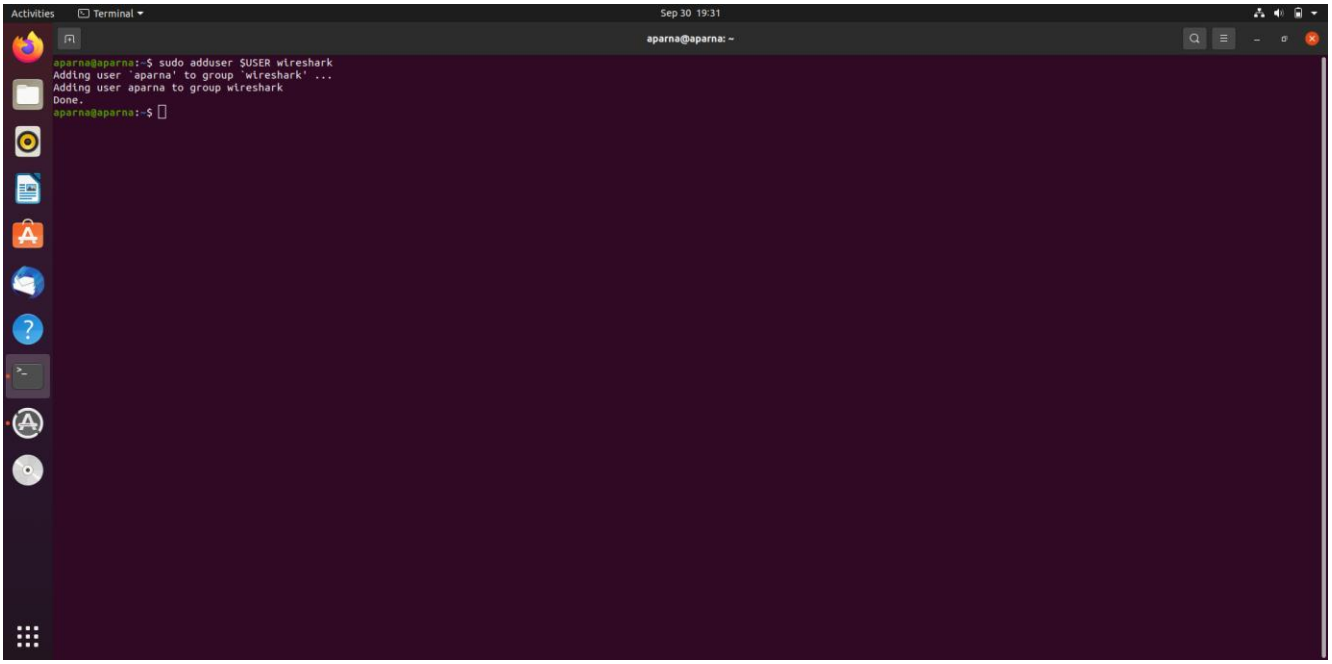
Enter Y to continue



Next give the command *sudo dpkg-reconfigure wireshark-common*

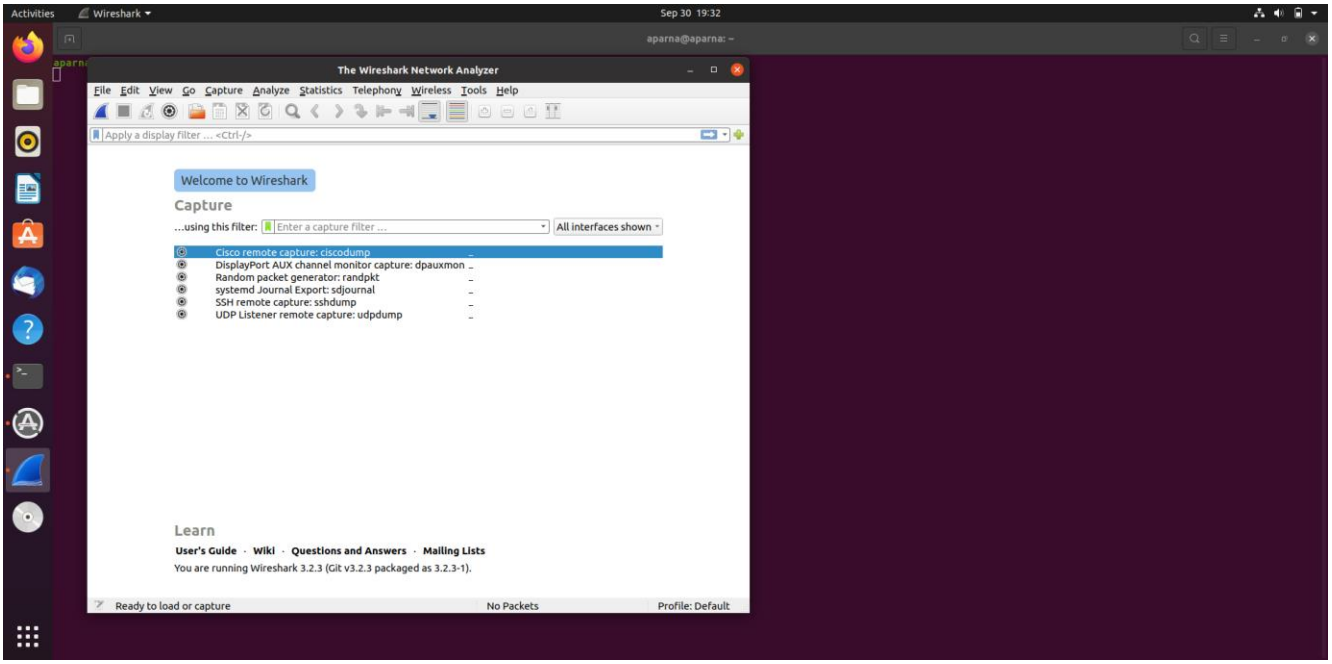


Next *sudo adduser $USER wireshark*



You can give appropriate option according to your need. After this the wireshark will be downloaded to the system

On launching Wireshark, you will see a screen like this:



The basic features of Wireshark are:

1) Packet Monitor: This segment visually shows the packets flowing inside the network. There are colour codes for each type of packets. The packets are shown with following information :

1. Source address
2. Destination address
3. Packet type
4. Hex dump of the packet
5. Contents of the packet in text
6. Source port(if applicable)
7. Destination port(if applicable)

2) Import from a capture file:

This feature lets you import packets dump from a capture file to analyse further. There are many formats supported by Wireshark, some of them are:

- pcapng
- libpcap
- Oracle snoop and atmsnoop
- Finisar (previously Shomiti) Surveyor captures
- Microsoft Network Monitor captures
- Novell LANalyzer captures
- AIX iptrace captures
- Cinco Networks NetXray captures
- Network Associates Windows-based Sniffer and Sniffer Pro captures
- Network General/Network Associates DOS-based Sniffer (compressed or uncompressed) captures
- AG Group/WildPackets/Savvius EtherPeek/TokenPeek/AiroPeek/EtherHelp/PacketGrabber captures
- RADCOM's WAN/LAN Analyzer captures
- Network Instruments Observer version 9 captures
- Lucent/Ascend router debug output
- HP-UX's nettl

•Toshiba's ISDN routers dump output

•ISDN4BSD i4btrace utility

•Traces from the EyeSDN USB S0

•IPLog format from the Cisco Secure Intrusion Detection System pppd logs (pppdump format)

•the output from VMS's TCPIPtrace/TCPtrace/UCX$TRACE utilities

•the text output from the DBS Etherwatch VMS utility

•Visual Networks' Visual UpTime traffic capture

•the output from CoSine L2 debug

•the output from Accellent's 5Views LAN agents

•Endace Measurement Systems' ERF format captures

•Linux Bluez Bluetooth stack hcidump -w traces

•Catapult DCT2000 .out files

•Gammu generated text output from Nokia DCT3 phones in Netmonitor mode

•IBM Series (OS/400) Comm traces (ASCII & UNICODE)

•Juniper Netscreen snoop captures

•Symbian OS btsnoop captures

•Tamosoft CommView captures

•Textronix K12xx 32bit .rf5 format captures

•Textronix K12 text file format captures

•Apple PacketLogger captures

•Captures from Aethra Telecommunications' PC108 software

3) Export to a capture file: Wireshark lets you save the results as a capture file to continue working on them at later point of time. The supported formats are:

•pcapng (*.pcapng)

•libpcap, tcpdump and various other tools using tcpdump's capture format (*.pcap, *.cap, *.dmp)

•Accellent 5Views (*.5vw)

•HP-UX's nettl (*.TRC0, *.TRC1)

•Microsoft Network Monitor – NetMon (*.cap)

•Network Associates Sniffer – DOS (*.cap, *.enc, *.trc, *fdc, *.syc)

•Network Associates Sniffer – Windows (*.cap)

•Network Instruments Observer version 9 (*.bfr)

•Novell LANalyzer (*.tr1)

•Oracle (previously Sun) snoop (*.snoop, *.cap)

•Visual Networks Visual UpTime traffic (*.*).

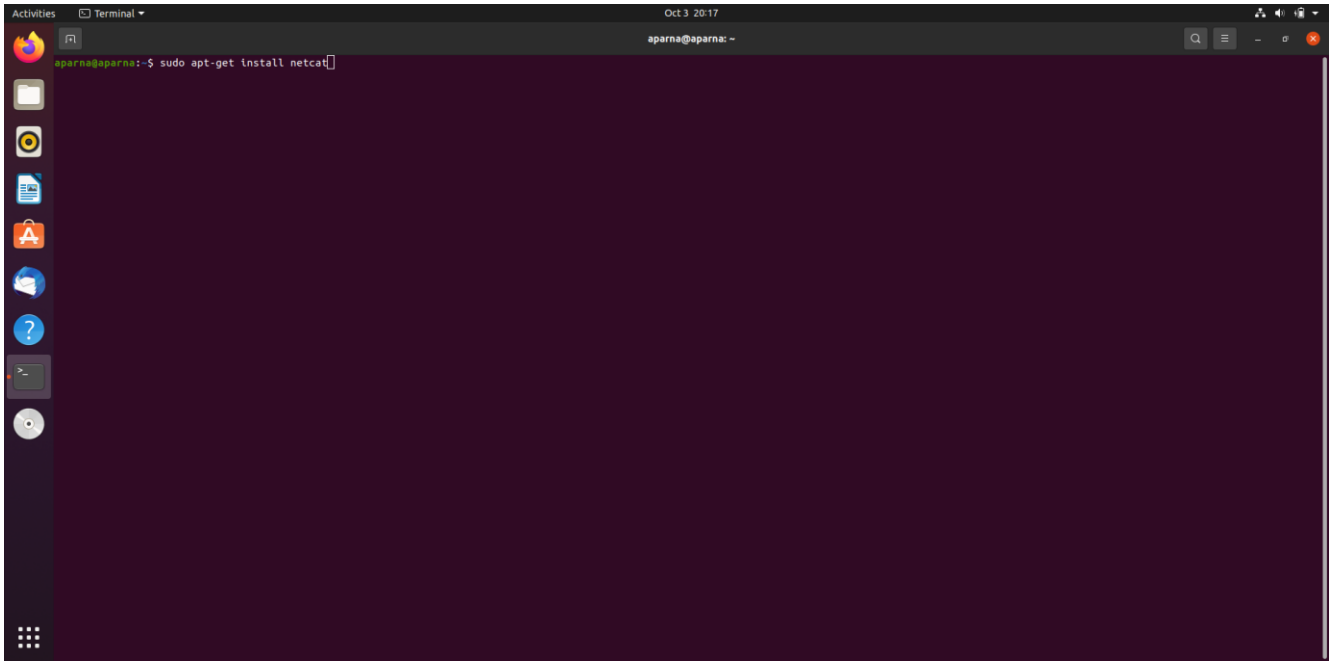
Netcat:

Netcat (or nc in short) is a simple yet powerful networking command-line tool used for performing any operation in Linux related to TCP, UDP, or UNIX-domain sockets.

Netcat can be used for port scanning, port redirection, as a port listener (for incoming connections); it can also be used to open remote connections and so many other things. Besides, you can use it as a backdoor to gain access to a target server.

Installing netcat on linux:

```
sudo apt-get install netcat
```



Port scanning:

Netcat can be used for port scanning: to know which ports are open and running services on a target machine. It can scan a single or multiple or a range of open ports.

The -z option sets nc to simply scan for listening daemons, without actually sending any data to them. The -v option enables verbose mode and -w specifies a timeout for connection that can not be established.

Syntax:

```
nc -vz IP_address port
```

Connection timed out:

A *connection timed out* response indicates that your connection is not working, which could mean your firewall is blocking the port. Test the connection status by adding a rule that accepts connections on the required port.

Connection succeeded

If the initial connection succeeds, Netcat can connect to the service. Look at the connection in more detail.

Syntax:

```
nc -vt IP Address Port
```

Closing the connection

You can terminate the connection by either pressing Ctrl-C or type the service-specific quit command.

Experiment No. 10

Aim

Introduction to Hypervisors and VMs, Xen or KVM , Introduction to Containers: Docker, installation and deployment.

Result

Hypervisor

A **hypervisor**, also known as a virtual machine monitor or VMM, is software that creates and runs virtual machines (VMs). Ahypervisor allows one host computer to support multiple guest VMs by virtually sharing its resources, such as memory and processing.

Hypervisors make it possible to use more of a system's available resources and provide greater IT mobility since the guest VMs are independent of the host hardware. This means they can be easily moved between different servers. Because multiple virtual machines can run off of one physical server with a hypervisor, a hypervisor reduces:

•Space
•Energy
•Maintenance requirements

There are two main hypervisor types, referred to as "Type 1" (or "bare metal") and "Type 2" (or "hosted"). A **type 1 hypervisor** acts like a lightweight operating system and runs directly on the host's hardware, while a **type 2 hypervisor** runs as a software layer on an operating system, like other computer programs.

The most commonly deployed type of hypervisor is the type 1 or bare-metal hypervisor, where virtualization software is installed directly on the hardware where the operating system is normally installed. Because bare-metal hypervisors are isolated from the attack-prone operating system, they are extremely secure. In addition, they generally perform better and more efficiently than hosted hypervisors. For these reasons, most enterprise companies choose bare-metal hypervisors for data center computing needs.

Benefits of hypervisors

There are several benefits to using a hypervisor that hosts multiple virtual machines:

- **Speed:** Hypervisors allow virtual machines to be created instantly, unlike bare-metal servers. This makes it easier to provision resources as needed for dynamic workloads.
- **Efficiency:** Hypervisors that run several virtual machines on one physical machine's resources also allow for more efficient utilization of one physical server. It is more cost- and energy-efficient to run several virtual machines on one physical machine than to run multiple underutilized physical machines for the same task.
- **Flexibility:** Bare-metal hypervisors allow operating systems and their associated applications to run on a variety of hardware types because the hypervisor separates the OS from the underlying hardware, so the software no longer relies on specific hardware devices or drivers.
- **Portability:** Hypervisors allow multiple operating systems to reside on the same physical server (host machine). Because the virtual machines that the hypervisor runs are independent from the physical machine, they are portable. IT teams can shift workloads and allocate networking, memory, storage and processing resources across multiple servers as needed, moving from machine to machine or platform to platform. When an application needs more processing power, the virtualization software allows it to seamlessly access additional machines.

Virtual machine

A **Virtual Machine** (VM) is a compute resource that uses software instead of a physical computer to run programs and deploy apps. One or more virtual "guest" machines run on a physical "host" machine. Each virtual machine runs its own operating system and functions separately from the other VMs, even when they are all running on the same host. This means that, for example, a virtual MacOS virtual machine can run on a physical PC.

Virtual machine technology is used for many use cases across on-premises and cloud environments. More recently, public cloud services are using virtual machines to provide virtual application resources to multiple users at once, for even more cost efficient and flexible computer

What are virtual machines used for?

Virtual machines (VMs) allow a business to run an operating system that behaves like a completely separate computer in an app window on a desktop. VMs may be deployed to accommodate different levels of processing power needs, to run software that requires a different operating system, or to test applications in a safe, sandboxed environment.

Virtual machines have historically been used for server virtualization, which enables IT teams to consolidate their computing resources and improve efficiency. Additionally, virtual machines can perform specific tasks considered too risky to carry out in a host environment, such as accessing virus- infected data or testing operating systems. Since the virtual machine is separated from the rest of the system, the software inside the virtual machine cannot tamper with the host computer.

Advantages of virtual machines

Virtual machines are easy to manage and maintain, and they offer several advantages over physical machines:

- VMs can run multiple operating system environments on a single physical computer, saving physical space, time and management costs.
- Virtual machines support legacy applications, reducing the cost of migrating to a new operating system. For example, a Linux virtual machine running a distribution of Linux as the guest operating system can exist on a host server that is running a non- Linux operating system, such as Windows.
- VMs can also provide integrated disaster recoveryand application provisioning options.

KVM

KVM stands for Kernel-based Virtual Machine. It allows the kernel to operate as a hypervisor. Moreover, it requires a processor with hardware virtualization extensions such as Intel VT or AMD-V. KVM was initially designed for x86 processors. Later, it was ported to processors such as ARM, PowerPC etc. Operating systems such as FreeBSD and illumos contain KVM as loadable kernel modules.
Also, KVM provides hardware-assisted virtualization for many guest operating systems such as Linux, Solaris, Windows, Haiku and OS X. Furthermore, Android 2.2, Darwin 8.-.1 etc. work with some limitations.

Furthermore, some graphical management tools having KVM are as follows.

Kimchi is KVM's web-based virtualization management tool

Virtual Machine Manager allows creating, editing, starting and stopping KVM based virtual machine

OpenQRM allows managing and controlling various data centre infrastructures.

GNOME Boxes – Gnome interface for handling libvirt guests on Linux.

oVirt is an open source virtualization management tool for KVM

Proxmox Virtual Environment is an open source virtualization management package with KVM and LXC.

Xen

Xen or Xen Project is a type 1 hypervisor. It provides services to allow multiple computer operating systems to execute on the same computer hardware simultaneously.

In brief, KVM and Xen are two hypervisors written in C language. The main difference between KVM and Xen is that KVM is a virtualization module in Linux kernel that works similar to a hypervisor while Xen is a type 1 hypervisor that allows multiple operating systems to execute on the same computer hardware simultaneously

Docker Container

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

Container images become containers at runtime and in the case of Docker containers - images become containers when they run on Docker Engine. Available for both Linux and Windows-based applications, containerized software will always run the same, regardless of the infrastructure. Containers isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging.

Docker containers that run on Docker Engine:

- Standard: Docker created the industry standard for containers, so they could be portable anywhere
- Lightweight: Containers share the machine's OS system kernel and therefore do not require an OS per application, driving higher server efficiencies and reducing server and licensing costs
- Secure: Applications are safer in containers and Docker provides the strongest default isolation capabilities in the industry

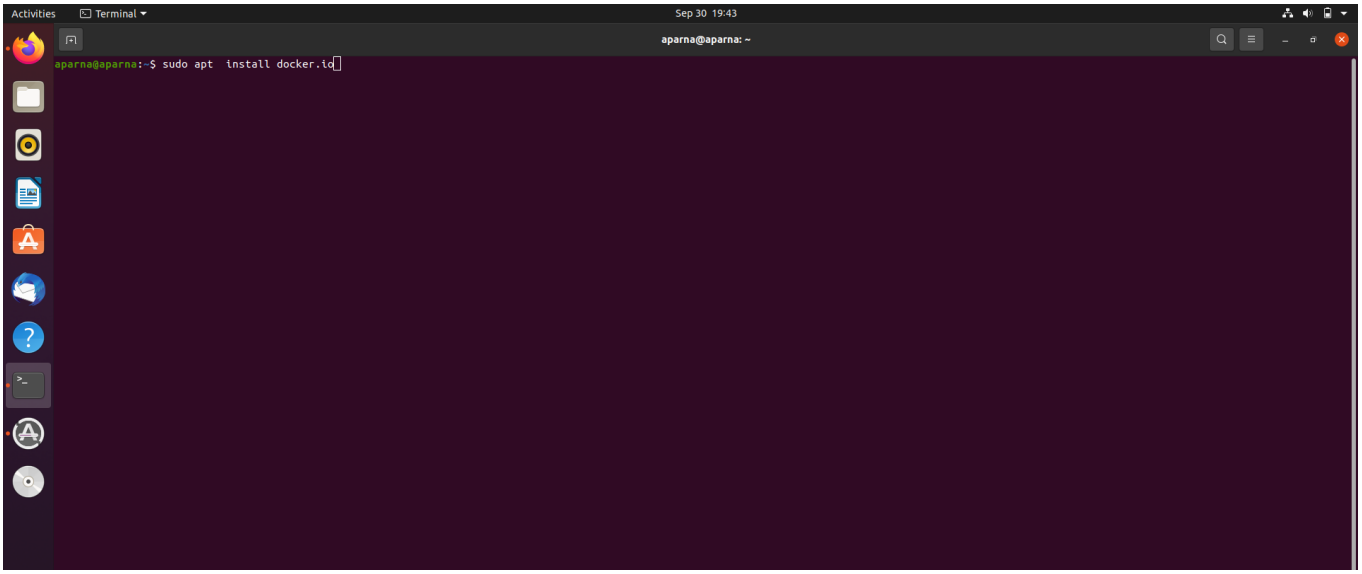
Installing Docker:

1) Updating the local repository

```
sudo apt update
```

2) Installing Docker

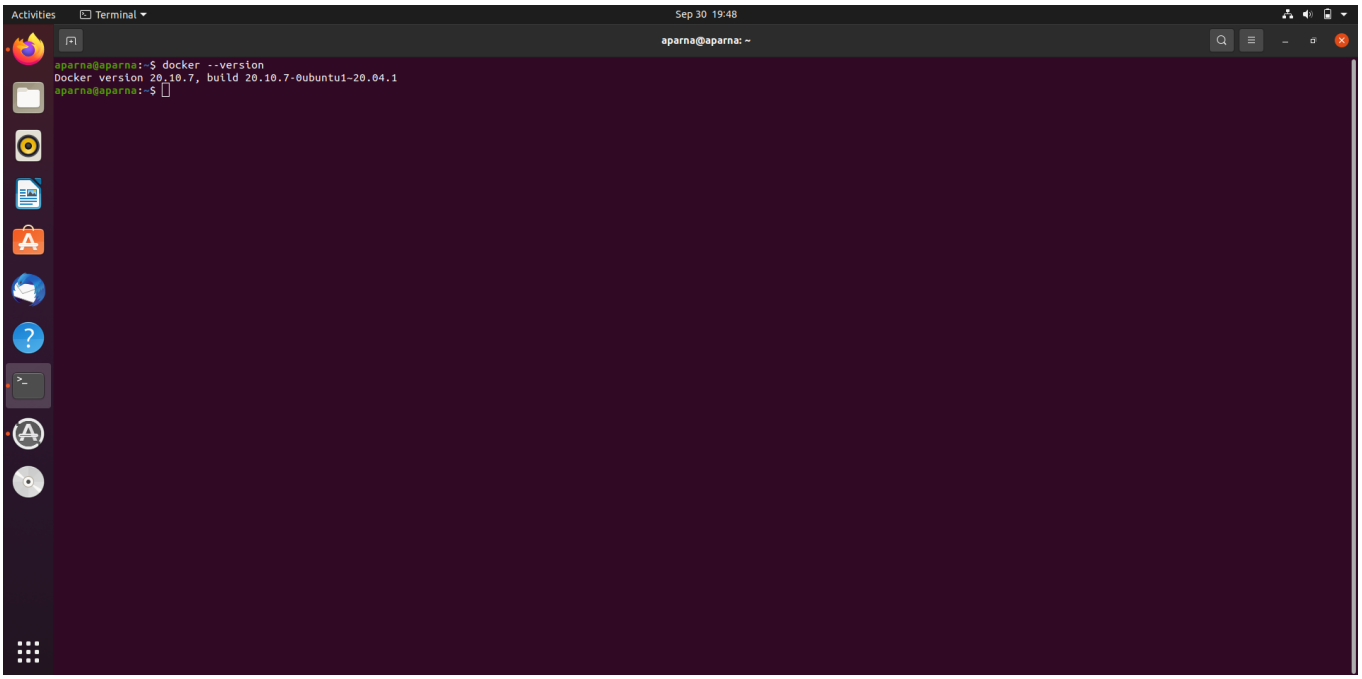
```
sudo apt install docker.io
```



Type **y** and hit Enter to confirm the installation. Once the install is completed, the output notifies you Docker has been installed.

3) Checking docker installation

```
docker --version
```



4) Starting docker service

* Start the Docker service by running:

```
sudo systemctl start docker
```

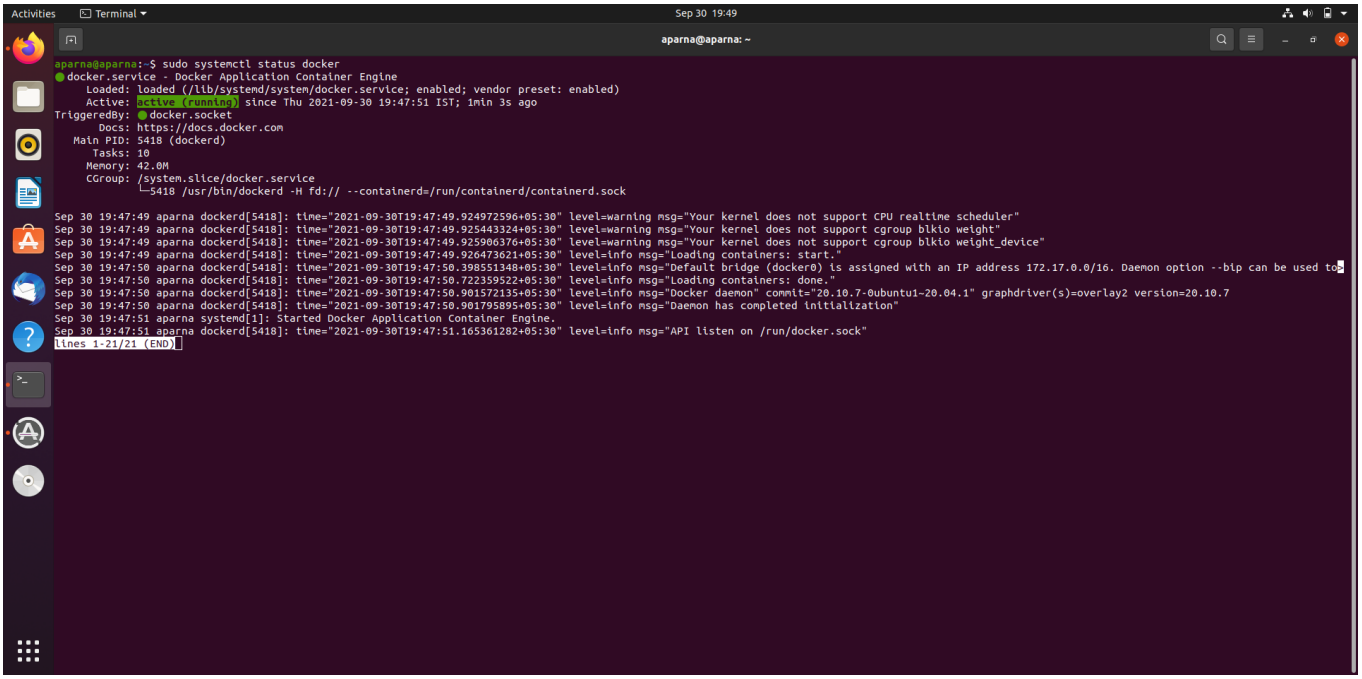
* Then, enable it to run at startup:

```
sudo systemctl enable docker
```

*To check the status of the service, run:

```
sudo systemctl status docker
```

The output should verify Docker is **active (running)**.



Use Docker on Ubuntu 20.04

The basic syntax for docker commands is:

```
sudo docker [option] [command] [argument]
```

Working With Docker Images

Docker images are files that contain the source code, libraries, dependencies, tools, and other files a container needs. You can create Docker images with Dockerfiles or use existing ones available on Docker Hub.

To download a new Docker image, use the command:

```
docker pull [image_name]
```

If you don’t know the exact name of the image, search for it in Docker’s repository with:

```
docker search ubuntu
```

After working with Docker for some time, you will collect a local registry of images. Display a list of all Docker images on the system with:

```
docker images
```

Working With Docker Containers

Docker containers are isolated virtual environments that run based on the Docker image assigned to them.

To run a container based on an existing Docker image, use the command:

```
docker run [image_name]
```

Using the command above runs a container but doesn’t move you inside of it. To run a container in interactive mode and change to the container command prompt, run:

```
docker run -it [image_name]
```

Another useful docker command is listing all the containers on the system. To list all active containers, type:

```
docker container ps
```

To view all containers (active and inactive), run:

```
docker container ps -a
```

Experiment No. 11

Aim

Automation using Ansible: Spin up a new Linux VM using Ansible playbook.

Result

Ansible

Ansible is an **open-source IT engine** that automates application deployment, cloud provisioning, intra service orchestration, and other IT tools. It is an **automation and orchestration tool** popular for the following reasons:

- Simple to Install.
- Free and open source.
- Lightweight and consistent.
- OpenSSH security features make it very secure.

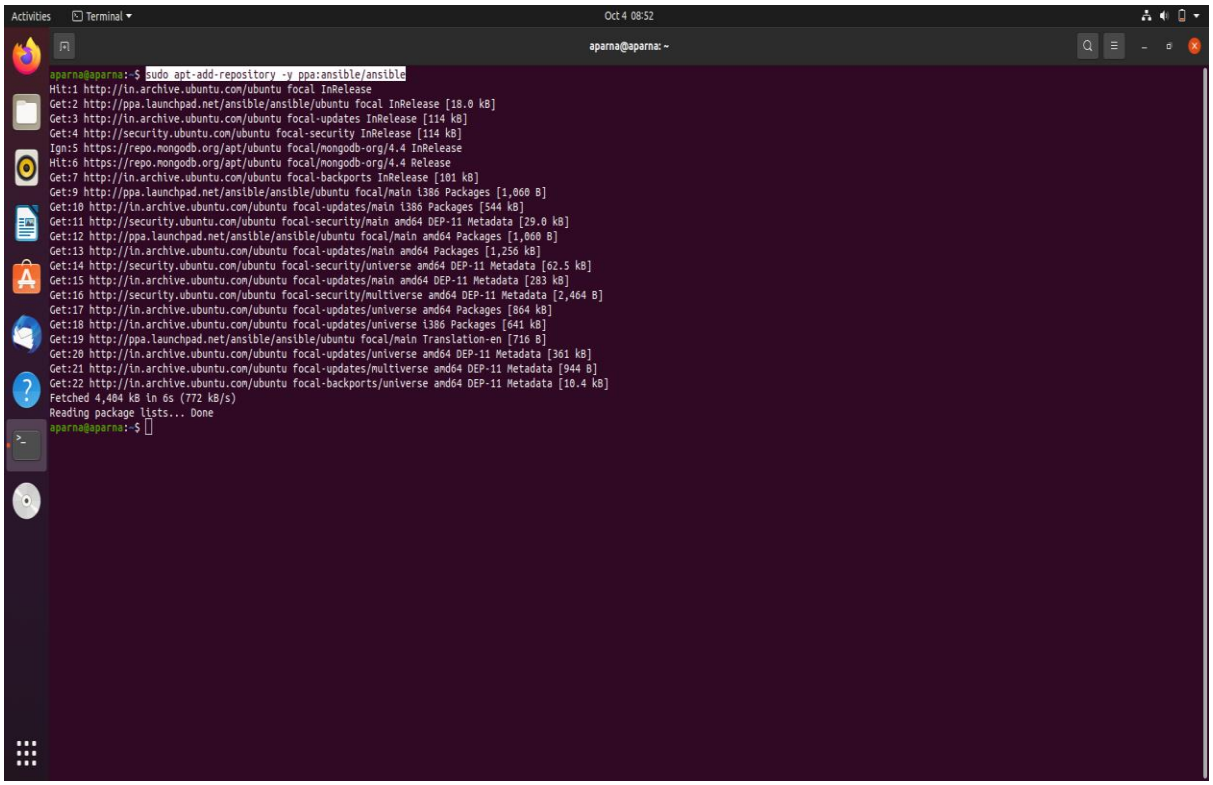
Ansible Concepts

1. **Control node:** Commands and Playbooks can run by invoking /usr/bin/ansible or /usr/bin/ansible-playbook, from any control node. You can use any computer that has Python installed on it as a control node. However, one can not use a computer with Windows OS as a control node. One can have multiple control nodes.
2. **Managed nodes:** Also sometimes called "hosts", Managed nodes are the network devices (and/or servers) you manage with Ansible.
3. **Inventory:** Also sometimes called "hostfile", Inventory is the list of Managed nodes use to organize them. It is also used for creating and nesting groups for easier scaling.
4. **Modules:** These are the units of code executed by Ansible. Each module can be used for a specific purpose. One can invoke a single module with a task, or invoke several different modules in a playbook.
5. **Tasks:** The units of action in Ansible. One can execute a single task once with an ad-hoc command.
6. **Playbooks3:** These are the ordered list of tasks that are saved so you can run those tasks in that order repeatedly. Playbooks are written in YAML and are easy to read, write, share and understand.

Installation of Ansible

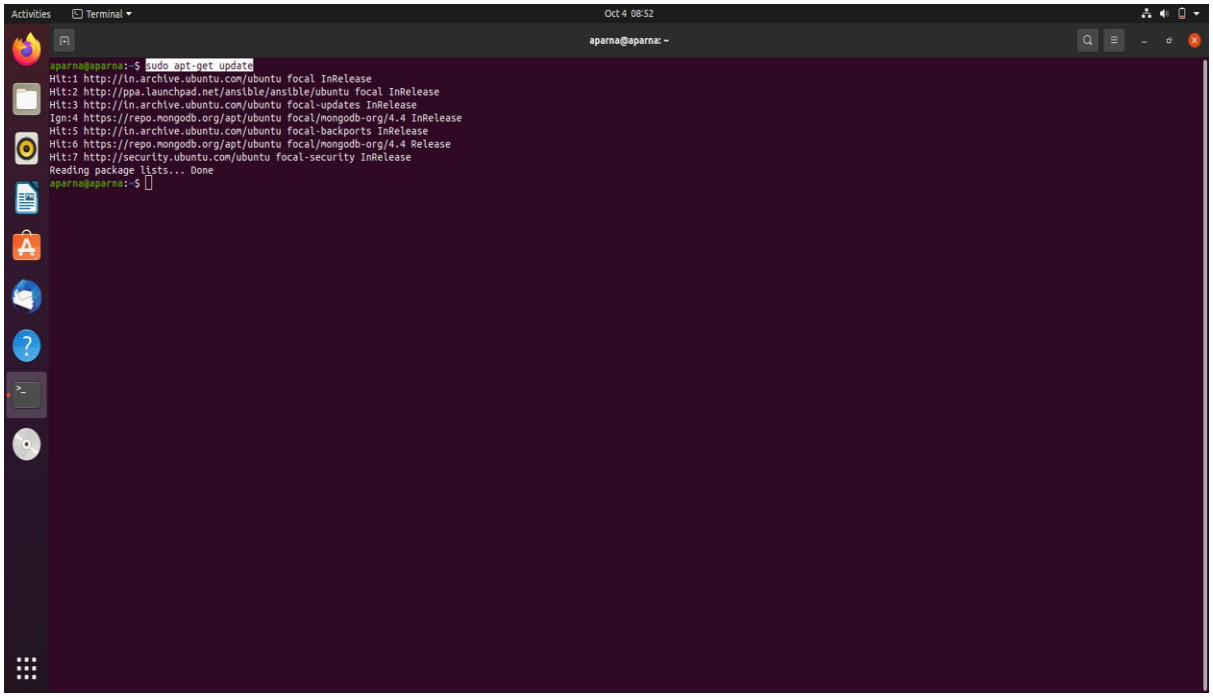
Step 1:Add the Ansible Repository.

```
sudo apt-add-repository -y ppa:ansible/ansible
```



Step 2: Update the system repository listings.

```
sudo apt-get update
```



Step 3: Install the ansible packages.

sudo apt-get install -y ansible