Practice Sessions Lab Manual

Operating System and Administration
20CS42P

IV Semester
Computer Science and Engineering

01 – Operating system (OS) and Administration

An ***Operating system*** (OS) is system software that manages computer hardware, software resources, and provides common services for computer programs. In other words, an *Operating System* (OS) is software that acts as an interface between computer hardware components and the user.

1. Types of OS installation

Attended installation:

- An installation process usually **needs a user who attends** it to make choices, such as accepting or declining an end-user license agreement (EULA), specifying preferences such as the installation location, supplying passwords or assisting in product activation.
- In graphical environments, installers that offer a **wizard-based interface** are common.
- Attended installers may ask users to help mitigate the errors.
    - For example, if the disk in which the computer program is being installed was full, the installer may ask the user to specify another target path or clear enough space in the disk.

Silent installation:

- An installation that does not display messages or windows during its progress.
- "Silent installation" is NOT the same as "unattended installation". [All silent installations are unattended but not all unattended installations are silent.]
- In bigger organizations where thousands of users work, deploying the applications becomes a typical task and for that reason, silent installation is

performed so that the application is installed in background without affecting the work of user.

Unattended installation:

- An installation that is performed without user interaction during its progress or with no user present at all.
- One of the reasons to use this approach is to automate the installation of a large number of systems.
- An unattended installation either does not require the user to supply anything or has received all necessary input prior to the start of installation. Such input may be in the form of command line switches or an answer file, a file that contains all the necessary parameters.
    - For example, if the installation medium was faulty, the installer should fail the installation, as there is no user to fix the fault or replace the medium. Unattended installers may record errors in a computer log for later review.

Headless installation:

- Installation performed without using a computer monitor connected.
- In attended forms of headless installation, another machine connects to the target machine (for example, via a local area network) and takes over the display output.
- Since a headless installation does not need a user at the location of the target computer, unattended headless installers may be used to install a program on multiple machines at the same time.

Scheduled or Automated installation:

- An installation process that runs on a preset time or when a predefined condition meet the requirements, as opposed to an installation process that starts explicitly on a user's command.
    - For example, a system administrator willing to install a later version of a computer program that is being used can schedule that installation to occur when that program is not running.
- An operating system may automatically install a device driver for a device that the user connects.

Clean installation:

- A clean installation is one that is done in the absence of any interfering elements such as old versions of the computer program being installed or leftovers from a previous installation.
- The clean installation of an operating system is an installation in which the target disk partition is erased before installation.
- Since the interfering elements are absent, a clean installation may succeed where an unclean installation may fail or may take significantly longer.

Network installation:

Network installation (**netinstall**), is an installation of a program from a shared network resource that may be done by installing a minimal system before proceeding to download further packages over the network.

This may simply be a copy of the original media but software publishers which offer site licenses for institutional customers may provide a version intended for installation over a network.

=== * ===

2. Boot methods

Booting is the process of starting a computer as initiated via hardware such as a button or by a software command.

Types of Booting:

- **Warm Booting**: The Warm Booting is that in which system starts from the starting or from initial state means.
    - In the Warm Booting the system will be started from its beginning state means, first, the user will press the **Power Button**, then this will read all the instructions from the ROM and the Operating System will be automatically gets loaded into the System (RAM).
- **Cold Booting**: The Cold Booting is that in which System automatically starts when the system is in a running state.
    - For example, due to Light Fluctuation, the system will automatically **restarts**. In this, chances of damaging of system are more. The system will now be start from its initial state, so some files may be damaged because they are not properly stored into the system.

=== * ===

3. File System and Formatting:

File System:

- A file system is a process of managing how and where data on a storage disk, which is also referred to as file management or FS.
- It is a logical disk component that compresses files separated into groups, which is known as directories.
- The file system enables user to view a file in the current directory as files are often managed in a hierarchy.

- It is abstract to a human user and related to a computer; hence, it manages a disk's internal operations.
- NTFS is the most common file system in modern times (Windows OS).
- Without file management, it would be impossible for a file with the same name to exist and also impossible to remove installed programs and recover specific files.

Examples of File Systems:

The examples of file systems are given below:

- **FAT:** FAT is a type of file system, which is developed for hard drives. It stands for **File Allocation Table**. On hard drives and other computer systems, it helps to manage files on Microsoft operating systems. In devices like digital cameras, flash memory, and other portable devices, it is also often found that is used to store file information. It also helps to extend the life of a hard drive as it minimizes the wear and tears on the hard disc. Now a days later versions of Microsoft Windows like Windows XP, Vista, 7, and 10 as use NTFS.
    - The **FAT8, FAT12, FAT32, FAT16** are all the different types of FAT (for file allocation table).
- **GFS:** A GFS is a file system, which stands for Global File System. It has the ability to make enable multiple computers to act as an integrated machine. When the physical distance of two or more computers is high, and they are unable to send files directly with each other, a GFS file system makes them capable of sharing a group of files directly. A computer can organize its I/O to preserve file systems with the help of a global file system.
- **HFS:** HFS (Hierarchical file system) is the file system that is used on a Macintosh computer for creating a directory at the time a hard disk is formatted. Generally, its basic function is to organize or hold the files on a

Macintosh hard disk. Apple is not capable of supporting to write to or format HFS disks since when OS X came on the market. Also, HFS-formatted drives are not recognized by Windows computers as HFS is a Macintosh format. With the help of WIN32 or NTFS file systems, Windows hard drives are formatted.

- **NTFS:** NTFS is the file system, which stands for NT file system and stores and retrieves files on Windows NT operating system and other versions of Windows like Windows 2000, Windows XP, Windows 7, and Windows 10. Sometimes, it is known as the **New Technology File System**. As compared to the FAT and HFS file system, it provides better methods of file recovery and data protection and offers a number of improvements in terms of extendibility, security, and performance.
- **UDF:** A UDF is a file system, stands for **Universal Disk Format** and used first developed by OSTA (Optical Storage Technology Association) for ensuring consistency among data written to several optical media. It is used with CD-ROMs and DVD-ROMs and is supported on all operating systems. Now, it is used in the process of CD-R's and CD-RW's, called packet writing.

Formatting:

Formatting is a process of preparing the storage device to store the data. Formatting storage device will erase the earlier contents of the device.

=== * ===

4. Post installation tasks:

Post Installation task is the set of steps to be carried out to ensure that the installation is complete and went smoothly.

Post Installation Tasks for Ubuntu Operating System:

- **Online accounts**

  The first step allows user to configure online accounts, in case user want to integrate the desktop with different services.

- **Livepatch**

  Livepatch is a service that allows the installation of some updates that would generally require a system reboot, such as those of the kernel.

- **Help improve Ubuntu**

  In this step user can choose whether or not to send data from his system to Ubuntu. The option is activated by default. The user can verify the secrecy of the data being sent beforehand. The results are used to improve Ubuntu.

- **Privacy**

  If required, a user can enable location services so that apps can determine user geographic location. All the applications installed by default on Ubuntu are free software.

- **You are ready to start!**

  The last screen shows some featured applications -some of which are not free software with the option to open the software center to install them.

=== * ===

02 – Install and configure Virtual Machine – VmWare

VmWare:

VmWare is a cross-platform virtualization application. It installs on existing operating systems and also it extends the capabilities of the existing computer so that it can run multiple operating systems (it means, inside multiple virtual machines) at the same time.

- **Host operating system (host OS):** the operating system of the physical computer on which VirtualBox was installed.
- **Guest operating system (guest OS):** the operating system that is running inside the virtual machine.
- **Virtual machine (VM).** When running, a VM is the special environment that VirtualBox creates for guest operating system.

VmWare's main features:

• Portability
• No hardware virtualization required
• Guest Additions: shared folders, seamless windows, 3D virtualization
• Multigeneration branched snapshots.
• Clean architecture; unprecedented modularity.
	• Great hardware support
	  ○ Guest multiprocessing (SMP)
	  ○ USB 2.0 device support
	  ○ Hardware compatibility
	  ○ Full ACPI support (Advanced Configuration and Power Interface)
	  ○ Multiscreen resolutions
	  ○ Built-in iSCSI support
	• Remote machine display

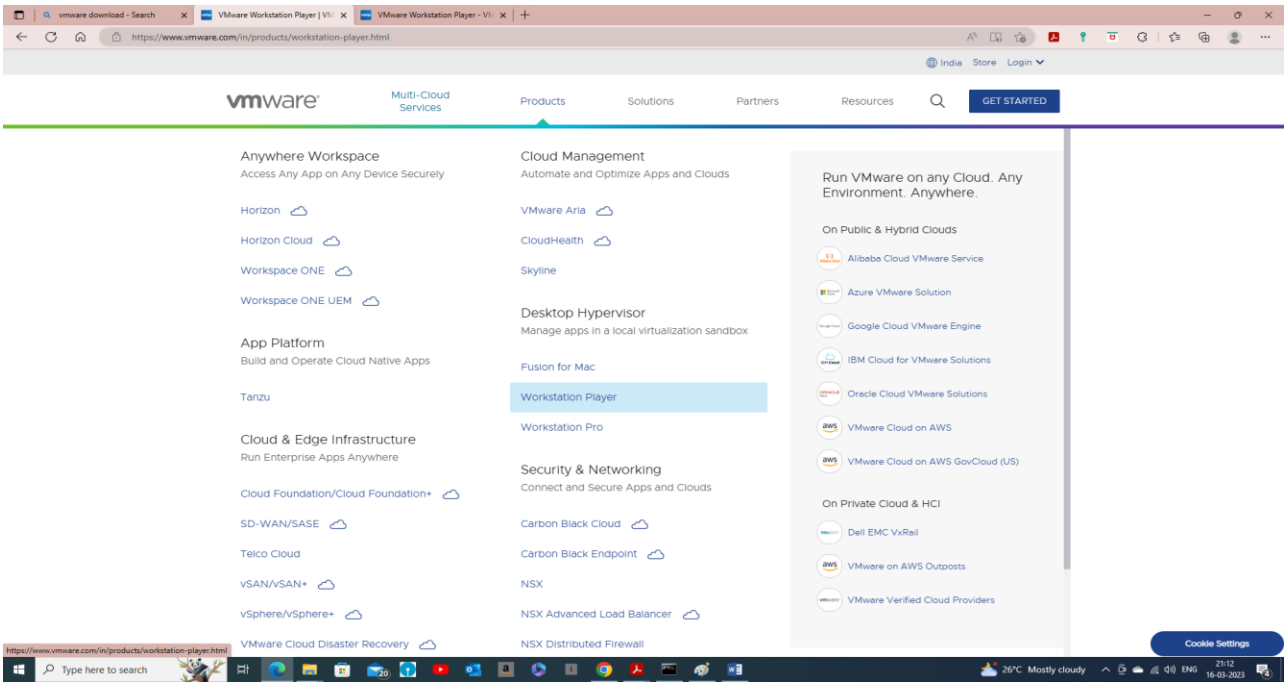
Host operating systems:

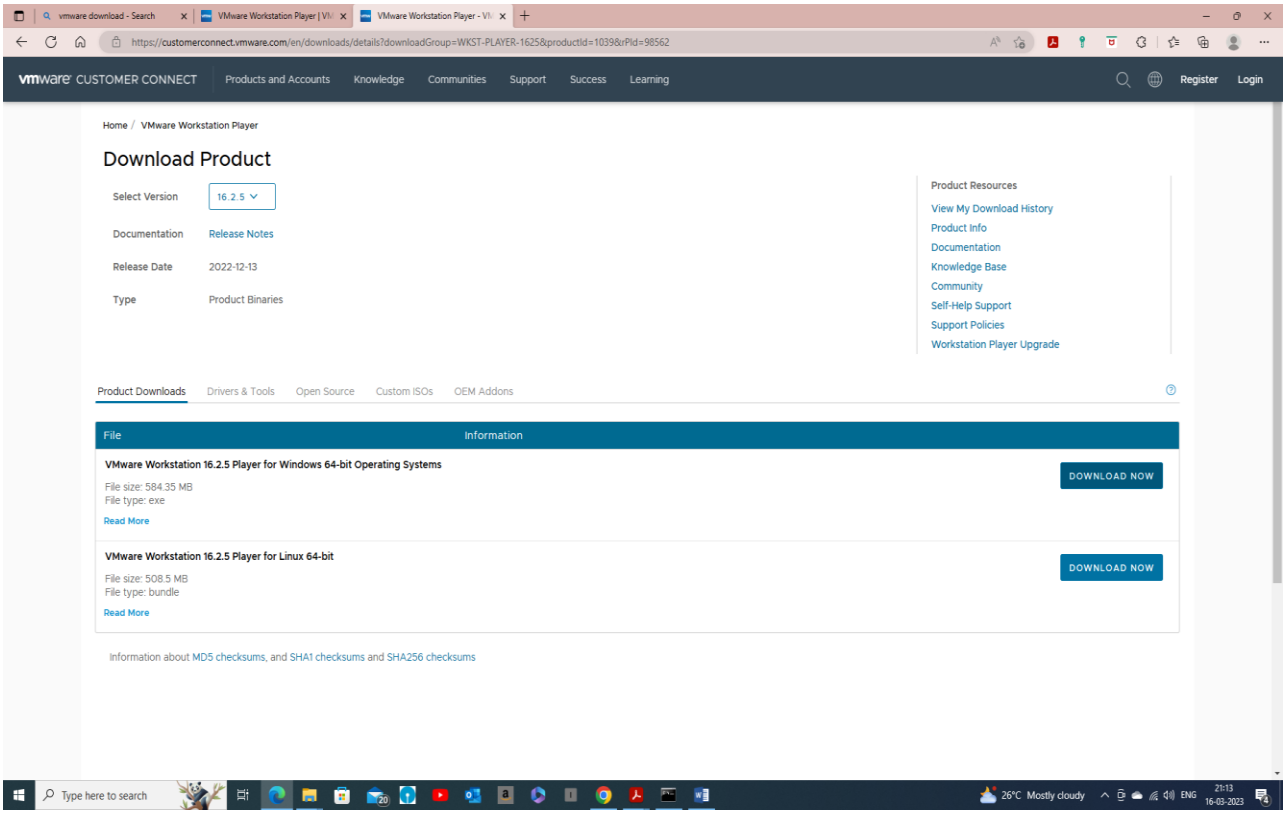
- **Windows hosts** (Windows XP, Windows 7, Windows Server 2003, 2008, Vista etc)

- **Mac OS X hosts** (Leopard 32-bit, Snow Leopard 32/64 - bit)
- **Linux hosts** (Ubuntu, Debian, Fedora, Mandriva etc)

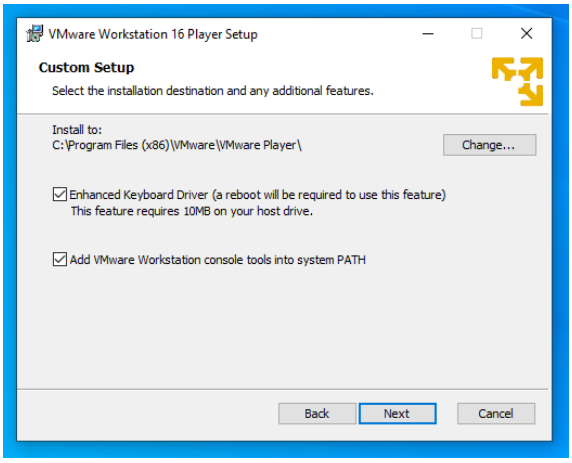
1. Steps to Install Virtual Machine:

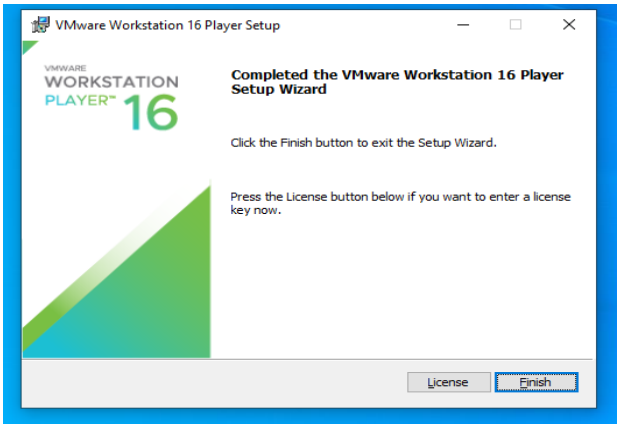
1. To install any Operating System on top of an existing Operating System first we need to install Virtual Machine Software.
2. To do so, we assume that Currently we are using Windows 10 or 11 OS which acts as a Host Operating System and We install any latest version of Ubuntu Operating system which acts as a Guest OS.
3. Either use Virtual Box or Vmware Software to install a Virtual Machine.
4. Download and Install the Vmware Workstation Player Software from its website https://www.vmware.com/in/products/workstation-player.html as shown below,
   Note: Before installing the software check the System Requirements for compatibility.

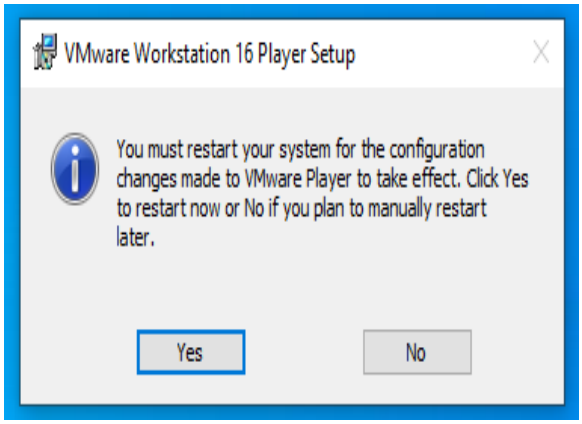



vmware CUSTOMER CONNECT
Products and Accounts
Knowledge
Communities
Support
Success
Learning
Register
Login
Home / VMware Workstation Player
Download Product
Select Version
16.2.5
Documentation
Release Notes
Release Date
2022-12-13
Type
Product Binaries
Product Resources
View My Download History
Product Info
Documentation
Knowledge Base
Community
Self-Help Support
Support Policies
Workstation Player Upgrade
Product Downloads
Drivers & Tools
Open Source
Custom ISOs
OEM Addons
File
Information
VMware Workstation 16.2.5 Player for Windows 64-bit Operating Systems
File size: 584.35 MB
File type: exe
Read More
DOWNLOAD NOW
VMware Workstation 16.2.5 Player for Linux 64-bit
File size: 508.5 MB
File type: bundle
Read More
DOWNLOAD NOW
Information about MD5 checksums, and SHA1 checksums and SHA256 checksums
Type here to search
26°C Mostly cloudy
ENG
21:13
16-03-2023


VMWARE
WORKSTATION
PLAYER™
16


VMware Workstation 16 Player Setup
Custom Setup
Select the installation destination and any additional features.
Install to:
C:\Program Files (x86)\VMware\VMware Player\
Change...
Enhanced Keyboard Driver (a reboot will be required to use this feature)
This feature requires 10MB on your host drive.
Add VMware Workstation console tools into system PATH
Back
Next
Cancel

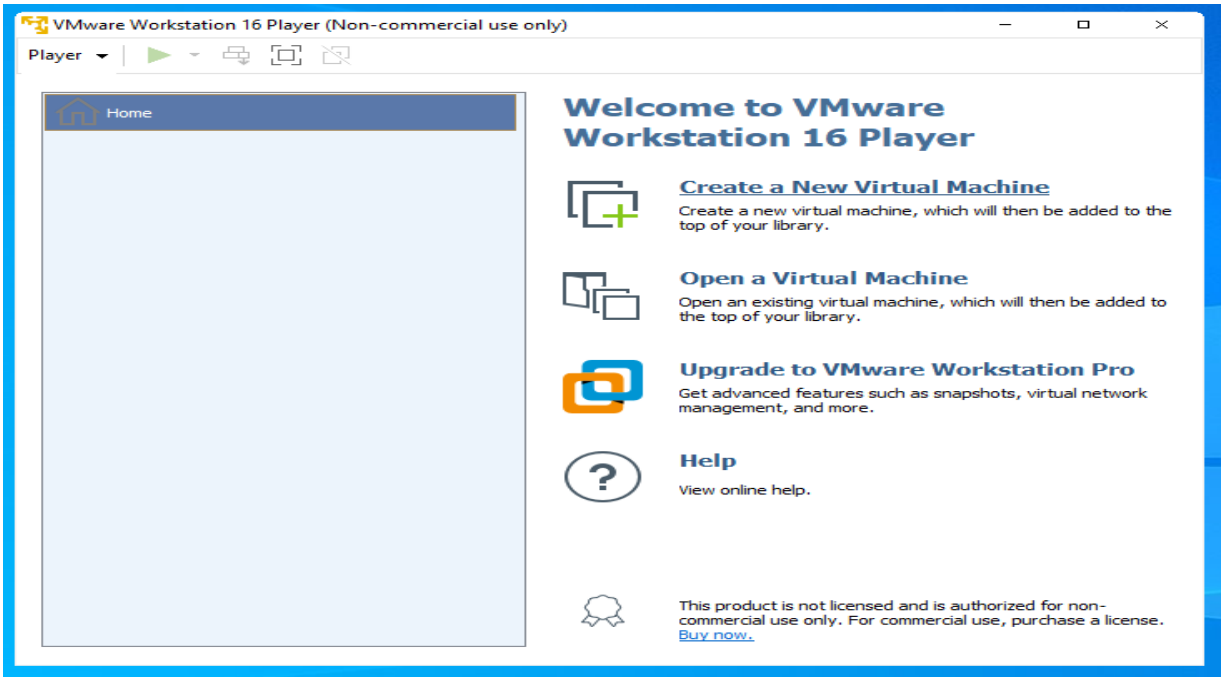


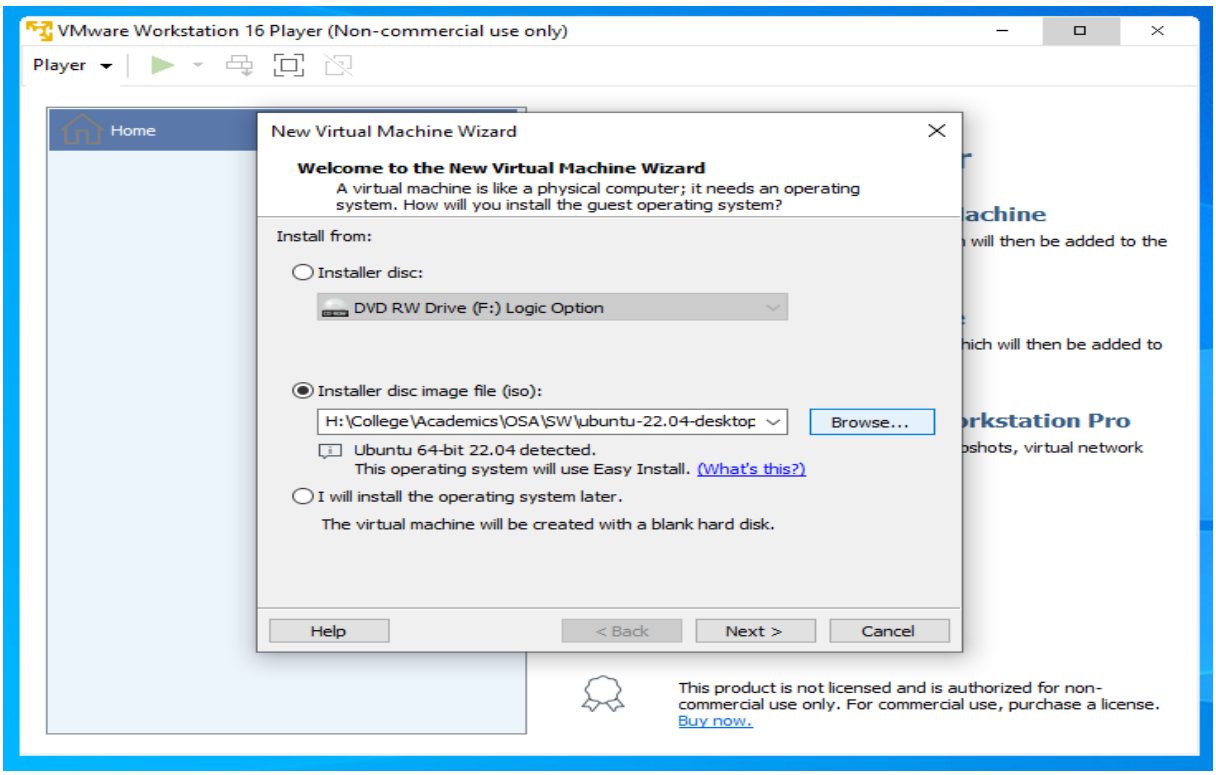


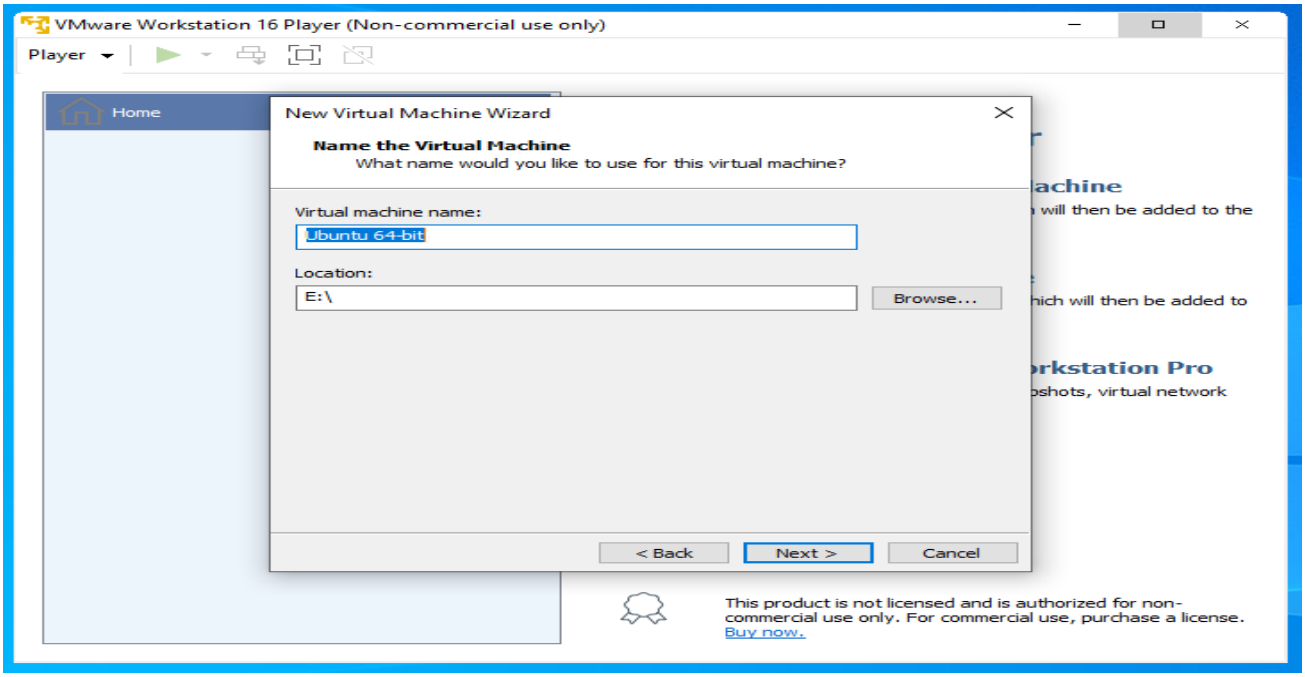
5. Once, installation is complete restart the system if required.
6. Now, Download the Ubuntu OS image from its website https://ubuntu.com/download/desktop.

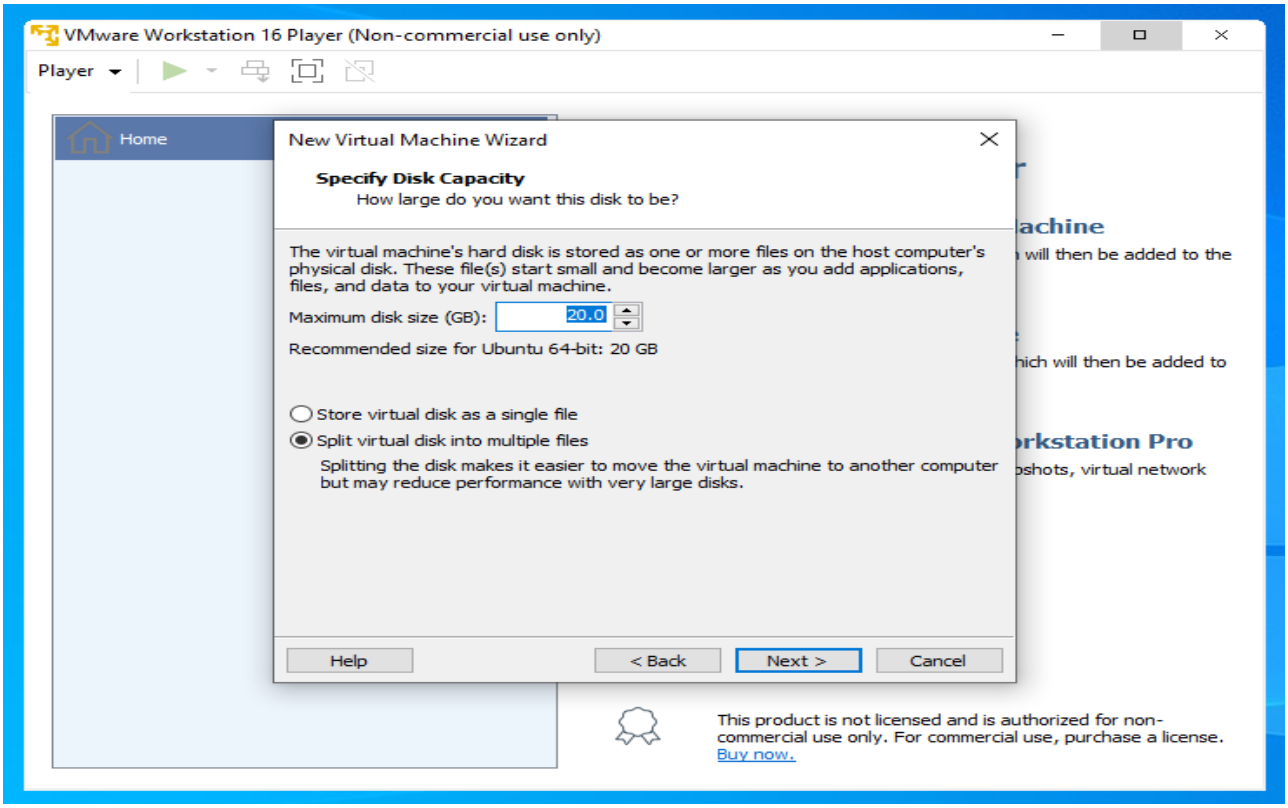
Note: Before installing the software check the System Requirements for compatibility.

7. Open the Vmware Application and proceed to install Ubuntu OS using the ISO file downloaded from the Ubuntu website as shown below,

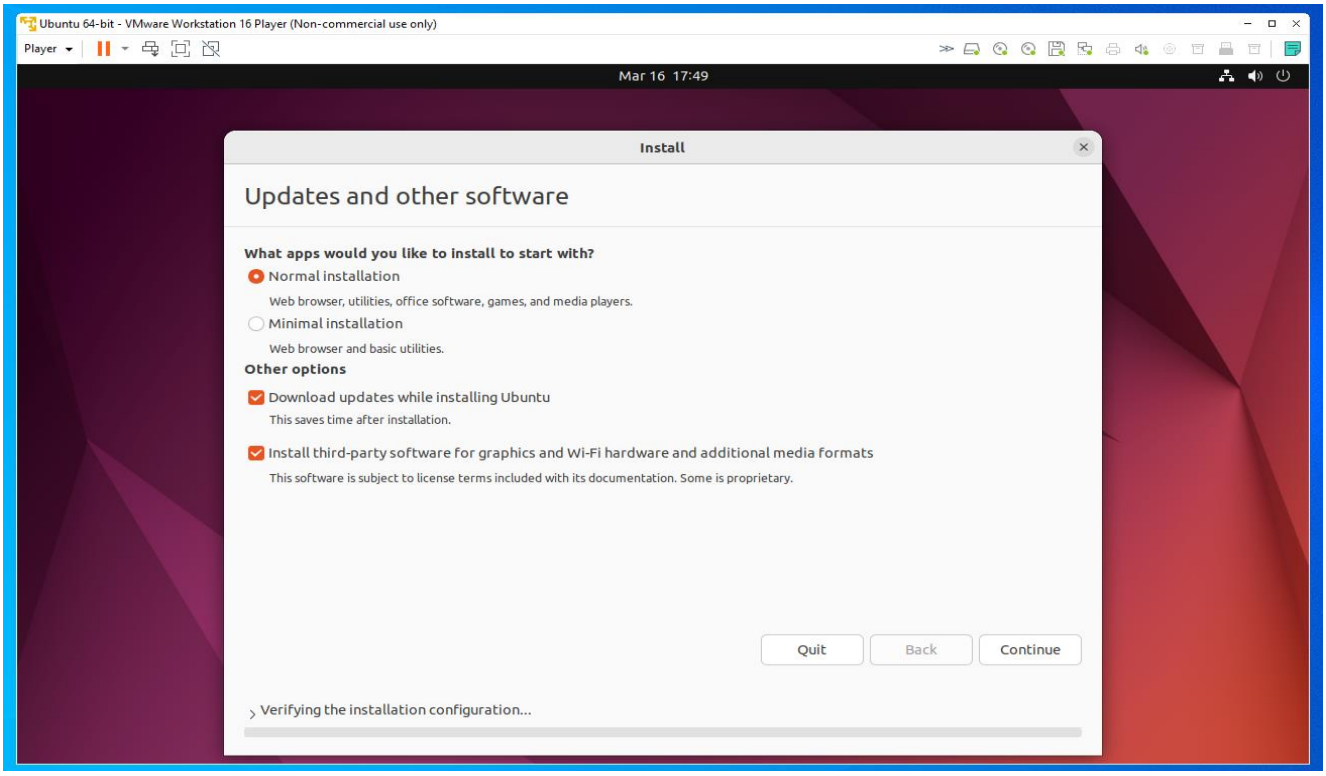

VMware Workstation 16 Player (Non-commercial use only)
Player
Home
Welcome to VMware Workstation 16 Player
Create a New Virtual Machine
Create a new virtual machine, which will then be added to the top of your library.
Open a Virtual Machine
Open an existing virtual machine, which will then be added to the top of your library.
Upgrade to VMware Workstation Pro
Get advanced features such as snapshots, virtual network management, and more.
Help
View online help.
This product is not licensed and is authorized for non-commercial use only. For commercial use, purchase a license.
Buy now.


VMware Workstation 16 Player (Non-commercial use only)
Player
Home
New Virtual Machine Wizard
Welcome to the New Virtual Machine Wizard
A virtual machine is like a physical computer; it needs an operating system. How will you install the guest operating system?
Install from:
Installer disc:
DVD RW Drive (F:) Logic Option
Installer disc image file (iso):
H:\College\Academics\OSA\SW\ubuntu-22.04-desktop
Browse...
Ubuntu 64-bit 22.04 detected.
This operating system will use Easy Install. (What's this?)
I will install the operating system later.
The virtual machine will be created with a blank hard disk.
Help
< Back
Next >
Cancel
This product is not licensed and is authorized for non-commercial use only. For commercial use, purchase a license.
Buy now.

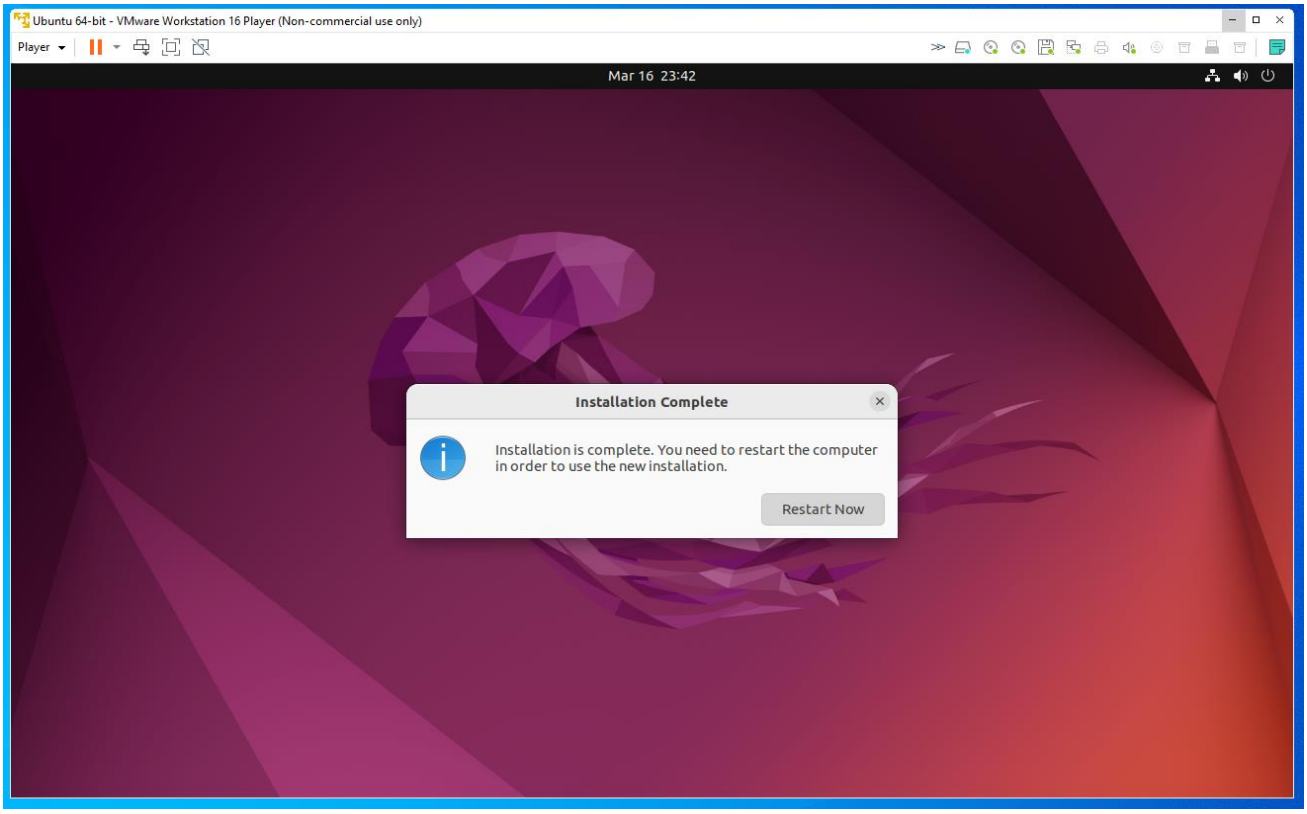

VMware Workstation 16 Player (Non-commercial use only)
Player
Home
New Virtual Machine Wizard
Name the Virtual Machine
What name would you like to use for this virtual machine?
Virtual machine name:
Ubuntu 64-bit
Location:
E:\
Browse...
< Back
Next >
Cancel
This product is not licensed and is authorized for non-commercial use only. For commercial use, purchase a license.
Buy now.


VMware Workstation 16 Player (Non-commercial use only)
Player
Home
New Virtual Machine Wizard
Specify Disk Capacity
How large do you want this disk to be?
The virtual machine's hard disk is stored as one or more files on the host computer's physical disk. These file(s) start small and become larger as you add applications, files, and data to your virtual machine.
Maximum disk size (GB):
20.0
Recommended size for Ubuntu 64-bit: 20 GB
Store virtual disk as a single file
Split virtual disk into multiple files
Splitting the disk makes it easier to move the virtual machine to another computer but may reduce performance with very large disks.
Help
< Back
Next >
Cancel
This product is not licensed and is authorized for non-commercial use only. For commercial use, purchase a license.
Buy now.

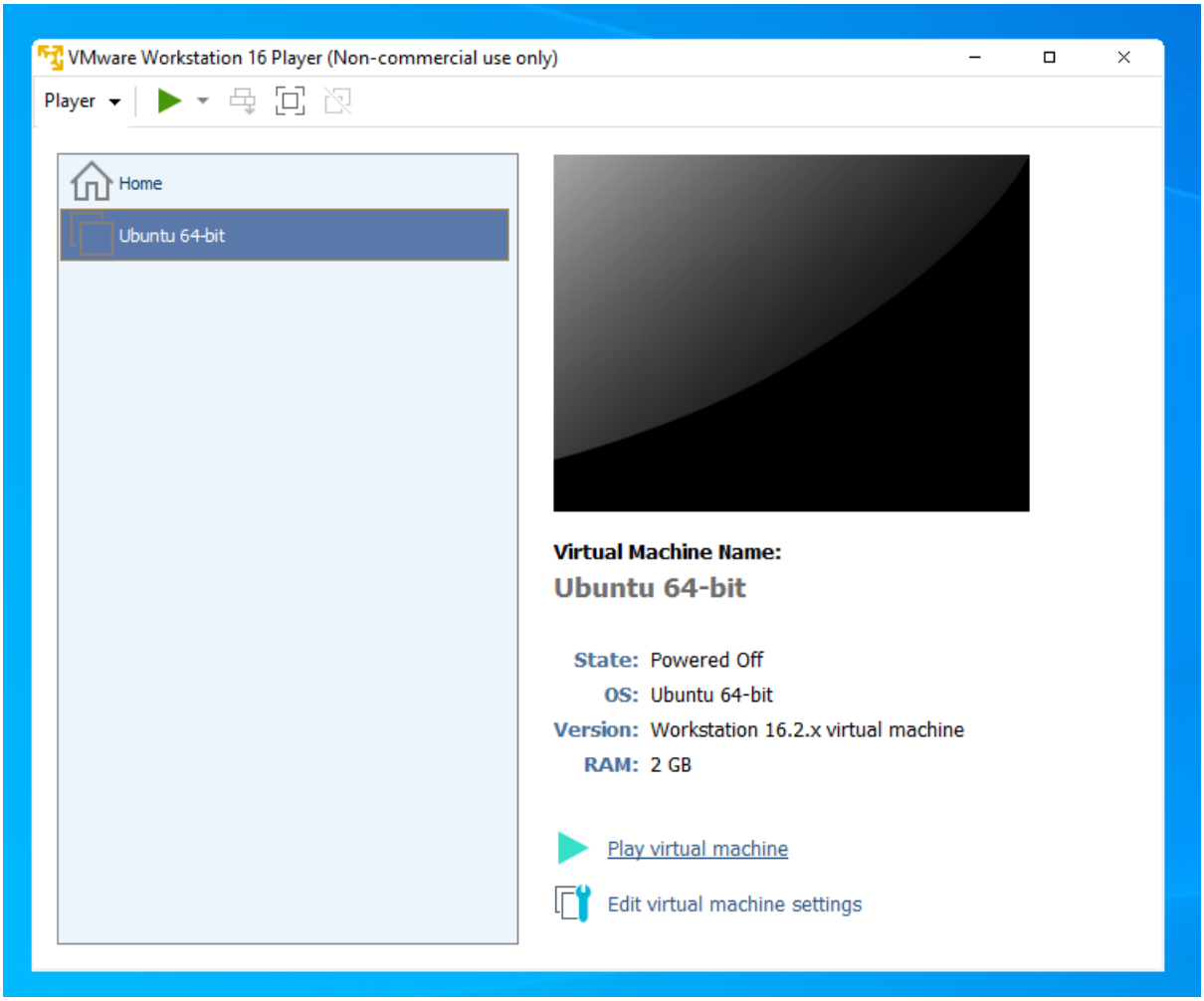
8. If you are connected to the internet during installation it's a better practice to download the required software packages which consumes time in installation or else you can skip this step.



9. Once installation is complete restart the Virtual Machine.



10. After installation is complete, you can rerun the Vmware Software to Play the virtual machine (Ubuntu)



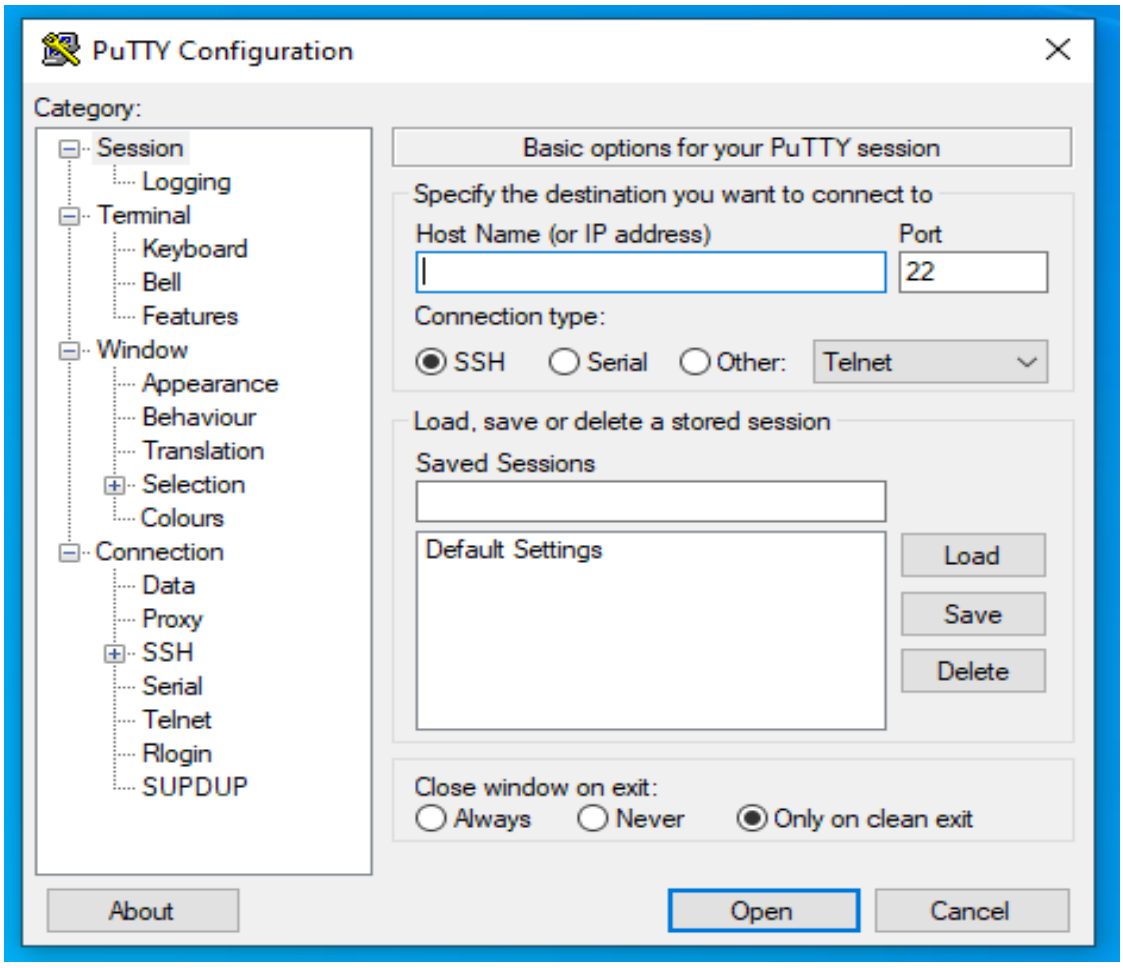
=== * ===

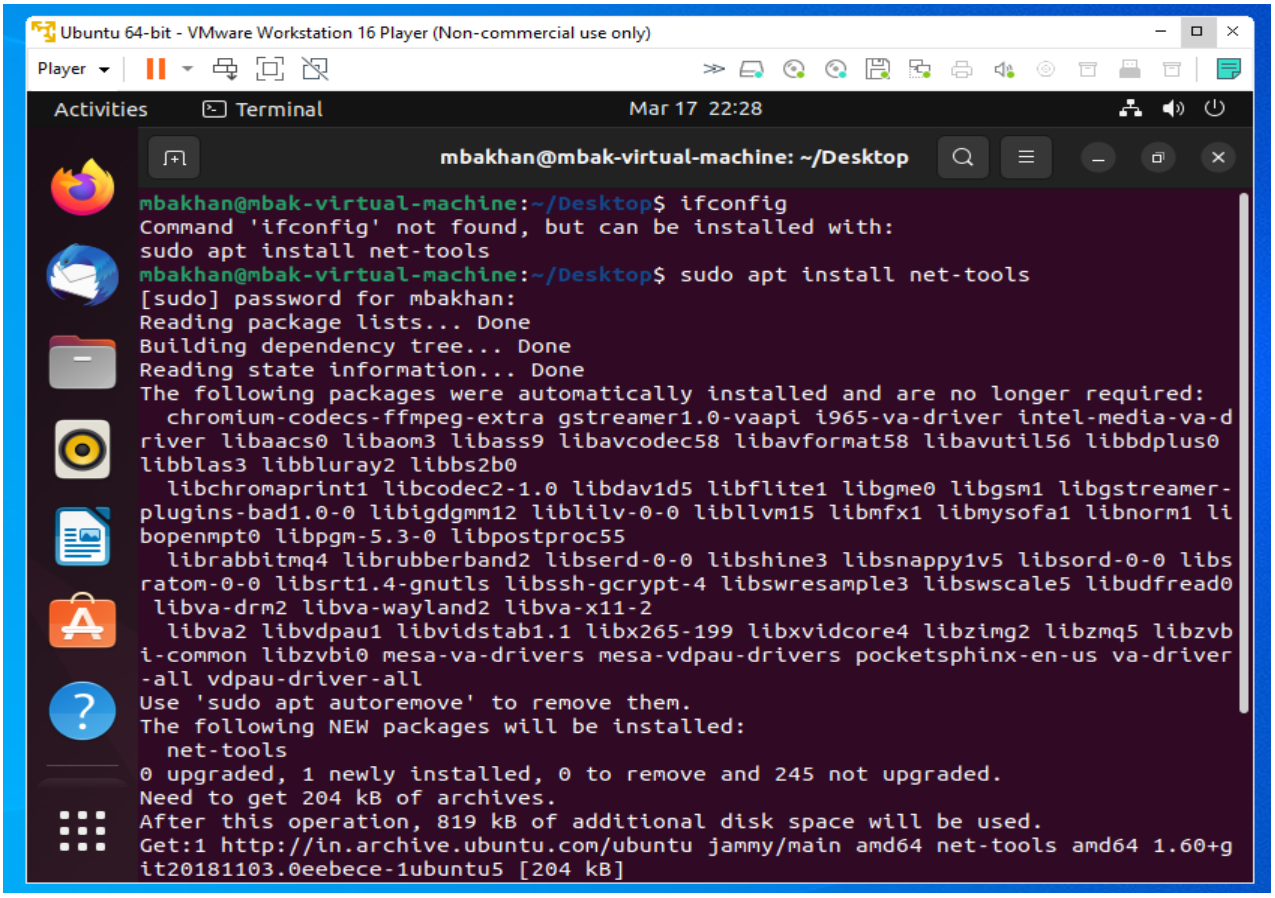
2. Steps to Install Terminal Emulator (PuTTY) in Windows and Connect Linux VM:

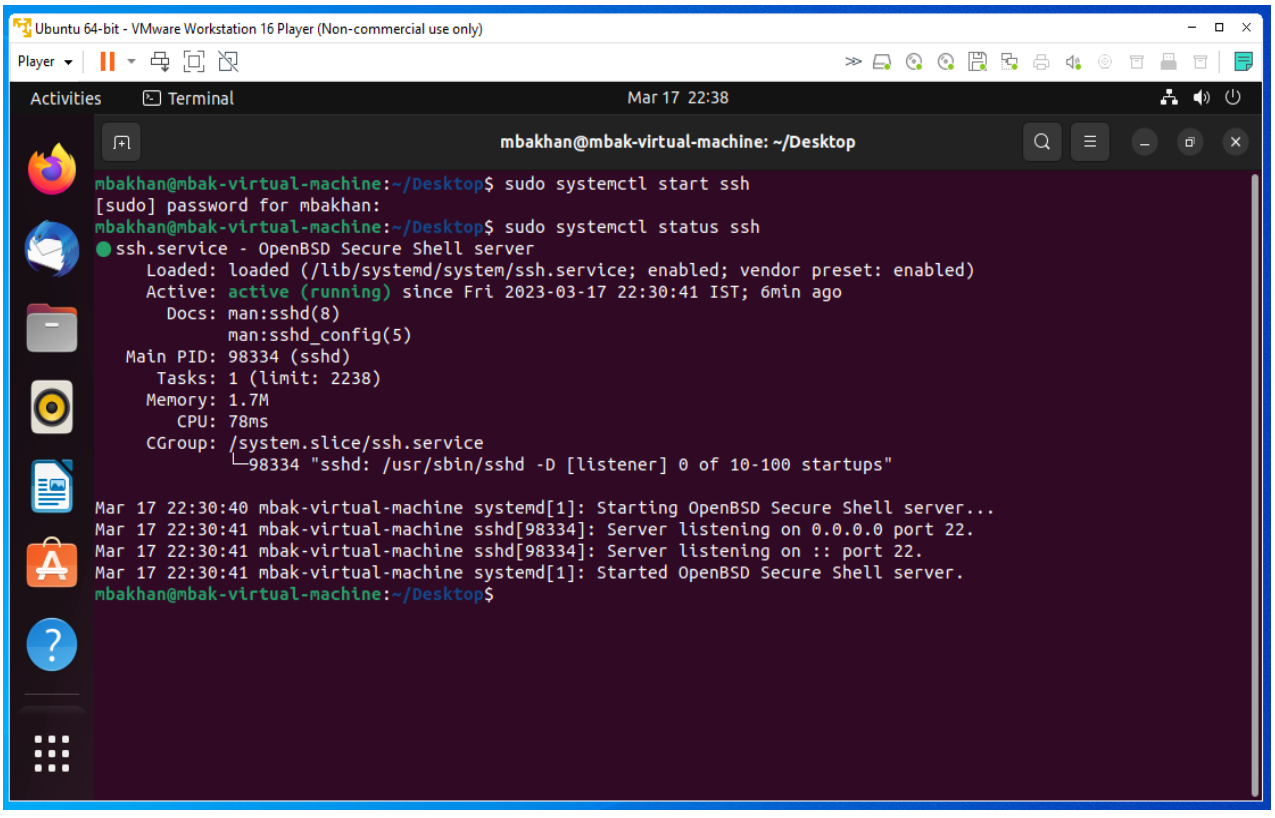
- A terminal emulator is a software that allows to you interact with the host machine with the help of commands.
- Terminal Emulator is a lifeline for every Linux distro as it enables you to utilize the real strength of Linux.
- Download and Install the PuTTY software from the link below https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
- To connect to VM or any PC running Linux OS using PuTTY LAN or Net connectivity is needed.

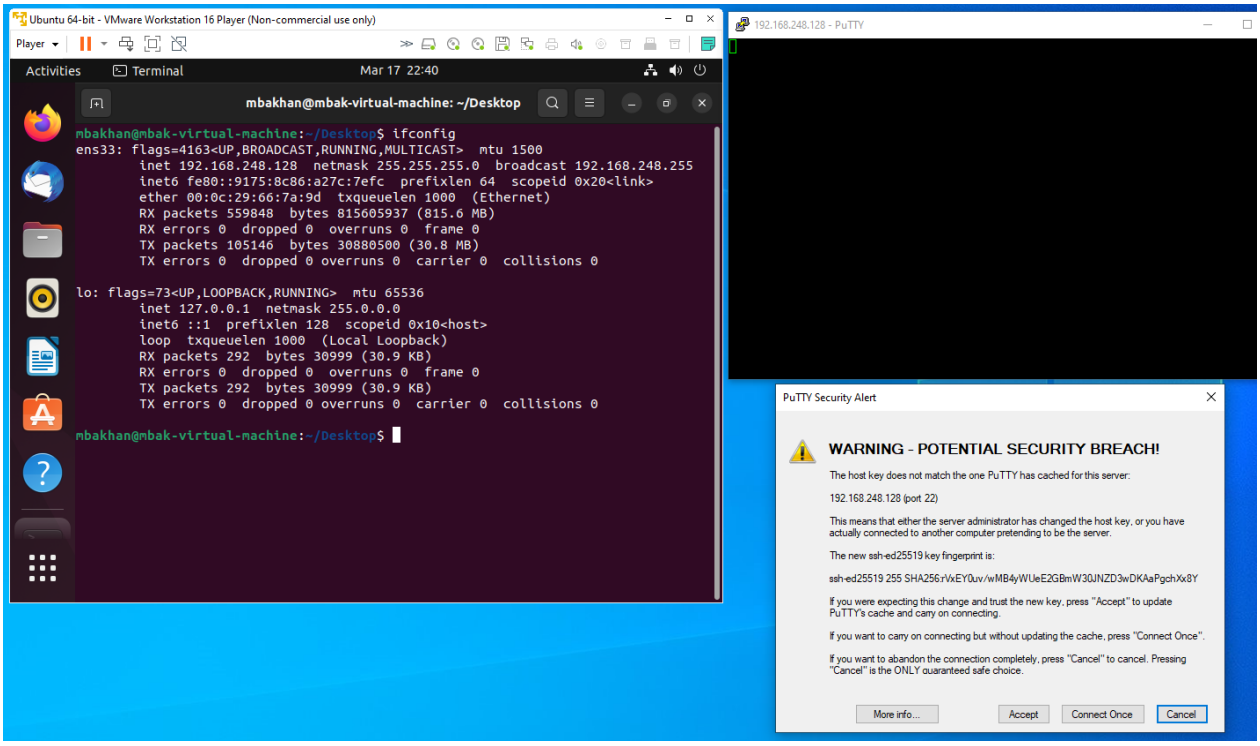
- Next, Play the Virtual Machine and Open the terminal by pressing Ctrl + Alt + T.
- Check the ip address of the Virtual Machine using the command ifconfig or ip address show.
- Install and start the Open SSH service using below commands,
  $ sudo apt-get install openssh-server
  $ sudo systemctl start ssh
  $ sudo systemctl status ssh

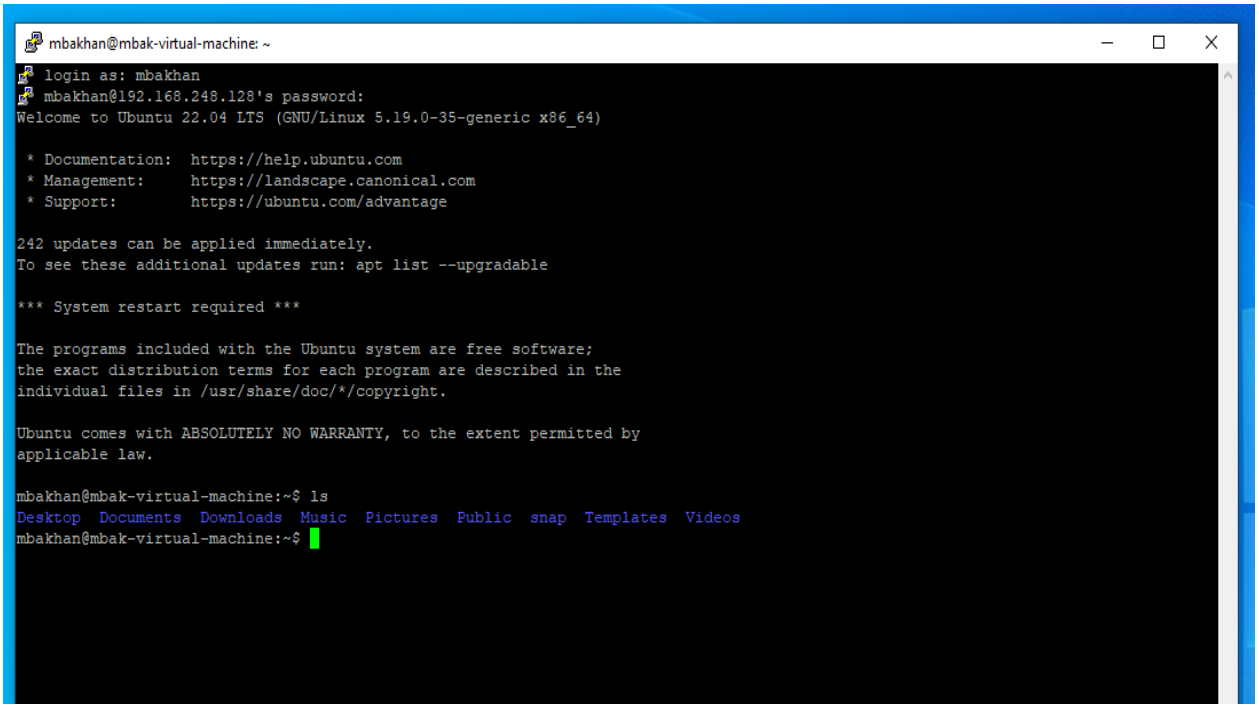
- Now, come back to Windows and open PuTTY application and enter the ip address or host name of the Linux VM.
- If everything goes fine then, you can establish a connection and access the Linux VM as if you are working in the Linux Vm.




Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
Mar 17 22:28
mbakhan@mbak-virtual-machine: ~/Desktop
mbakhan@mbak-virtual-machine:~/Desktop$ ifconfig
Command 'ifconfig' not found, but can be installed with:
sudo apt install net-tools
mbakhan@mbak-virtual-machine:~/Desktop$ sudo apt install net-tools
[sudo] password for mbakhan:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi i965-va-driver intel-media-va-driver libaacs0 libaom3 libass9 libavcodec58 libavformat58 libavutil56 libbdplus0 libblas3 libbluray2 libbs2b0
libchromaprint1 libcodec2-1.0 libdav1d5 libflite1 libgme0 libgsm1 libgstreamer-plugins-bad1.0-0 libigdgmm12 liblilv-0-0 libllvm15 libmfx1 libmysofa1 libnorm1 libopenmpt0 libpgm-5.3-0 libpostproc55
librabbitmq4 librubberband2 libserd-0-0 libshine3 libsnappy1v5 libsord-0-0 libsratom-0-0 libsrt1.4-gnutls libssh-gcrypt-4 libswresample3 libswscale5 libudfread0 libva-drm2 libva-wayland2 libva-x11-2
libva2 libvdpau1 libvidstab1.1 libx265-199 libxvidcore4 libzimg2 libzmq5 libzvbi-common libzvbi0 mesa-va-drivers mesa-vdpau-drivers pocketsphinx-en-us va-driver-all vdpau-driver-all
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
net-tools
0 upgraded, 1 newly installed, 0 to remove and 245 not upgraded.
Need to get 204 kB of archives.
After this operation, 819 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 net-tools amd64 1.60+git20181103.0eebece-1ubuntu5 [204 kB]


Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
Mar 17 22:38
mbakhan@mbak-virtual-machine: ~/Desktop
mbakhan@mbak-virtual-machine:~/Desktop$ sudo systemctl start ssh
[sudo] password for mbakhan:
mbakhan@mbak-virtual-machine:~/Desktop$ sudo systemctl status ssh
ssh.service - OpenBSD Secure Shell server
Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
Active: active (running) since Fri 2023-03-17 22:30:41 IST; 6min ago
Docs: man:sshd(8)
man:sshd_config(5)
Main PID: 98334 (sshd)
Tasks: 1 (limit: 2238)
Memory: 1.7M
CPU: 78ms
CGroup: /system.slice/ssh.service
└─98334 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
Mar 17 22:30:40 mbak-virtual-machine systemd[1]: Starting OpenBSD Secure Shell server...
Mar 17 22:30:41 mbak-virtual-machine sshd[98334]: Server listening on 0.0.0.0 port 22.
Mar 17 22:30:41 mbak-virtual-machine sshd[98334]: Server listening on :: port 22.
Mar 17 22:30:41 mbak-virtual-machine systemd[1]: Started OpenBSD Secure Shell server.
mbakhan@mbak-virtual-machine:~/Desktop$

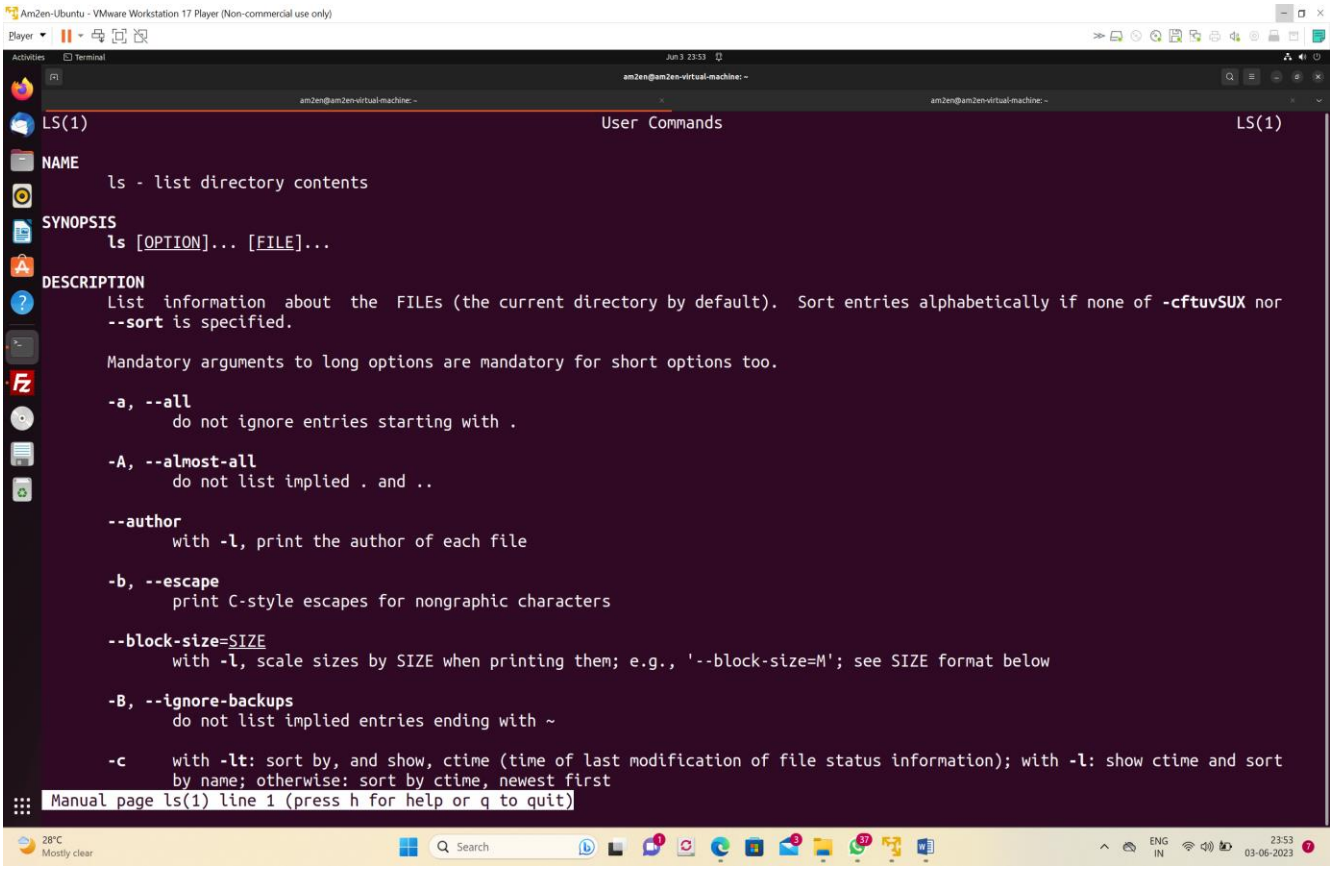

Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
Mar 17 22:40
mbakhan@mbak-virtual-machine: ~/Desktop
mbakhan@mbak-virtual-machine:~/Desktop$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.248.128 netmask 255.255.255.0 broadcast 192.168.248.255
inet6 fe80::9175:8c86:a27c:7efc prefixlen 64 scopeid 0x20<link>
ether 00:0c:29:66:7a:9d txqueuelen 1000 (Ethernet)
RX packets 559848 bytes 815605937 (815.6 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 105146 bytes 30880500 (30.8 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 292 bytes 30999 (30.9 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 292 bytes 30999 (30.9 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
mbakhan@mbak-virtual-machine:~/Desktop$
192.168.248.128 - PuTTY
PuTTY Security Alert
WARNING - POTENTIAL SECURITY BREACH!
The host key does not match the one PuTTY has cached for this server:
192.168.248.128 (port 22)
This means that either the server administrator has changed the host key, or you have actually connected to another computer pretending to be the server.
The new ssh-ed25519 key fingerprint is:
ssh-ed25519 255 SHA256:rVxEY0uv/wMB4yWUeE2GBmW30JNZD3wDKAaPgchXx8Y
If you were expecting this change and trust the new key, press "Accept" to update PuTTY's cache and carry on connecting.
If you want to carry on connecting but without updating the cache, press "Connect Once".
If you want to abandon the connection completely, press "Cancel" to cancel. Pressing "Cancel" is the ONLY guaranteed safe choice.
More info...
Accept
Connect Once
Cancel


mbakhan@mbak-virtual-machine: ~
login as: mbakhan
mbakhan@192.168.248.128's password:
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.19.0-35-generic x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
242 updates can be applied immediately.
To see these additional updates run: apt list --upgradable
*** System restart required ***
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
mbakhan@mbak-virtual-machine:~$ ls
Desktop Documents Downloads Music Pictures Public snap Templates Videos
mbakhan@mbak-virtual-machine:~$

Note: This experiment can be extended to n number of Windows pc's connected to one Linux pc by setting up LAN.

3. Significance of man Command

- **man** command in Linux is used to display the **user manual** of any command that 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.
- **Every manual is divided into the following sections**:
    - Executable programs or shell commands
    - System calls (functions provided by the kernel)
    - Library calls (functions within program libraries
    - Games
    - Special files (usually found in /dev)
    - File formats and conventions eg /etc/passwd
    - Miscellaneous (including macro packages and conventions),
    - System administration commands
- **Syntax**
    - man [SECTION-NUM] [COMMAND NAME]


Am2en-Ubuntu - VMware Workstation 17 Player (Non-commercial use only)
Player
Activities
Terminal
Jun 3 23:53
am2en@am2en-virtual-machine: ~
LS(1) User Commands LS(1)
NAME
ls - list directory contents
SYNOPSIS
ls [OPTION]... [FILE]...
DESCRIPTION
List information about the FILEs (the current directory by default). Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.
Mandatory arguments to long options are mandatory for short options too.
-a, --all
do not ignore entries starting with .
-A, --almost-all
do not list implied . and ..
--author
with -l, print the author of each file
-b, --escape
print C-style escapes for nongraphic characters
--block-size=SIZE
with -l, scale sizes by SIZE when printing them; e.g., '--block-size=M'; see SIZE format below
-B, --ignore-backups
do not list implied entries ending with ~
-c with -lt: sort by, and show, ctime (time of last modification of file status information); with -l: show ctime and sort by name; otherwise: sort by ctime, newest first
Manual page ls(1) line 1 (press h for help or q to quit)
28°C
Mostly clear
Search
ENG
IN
23:53
03-06-2023

=== * ===

03 – File and Directory Commands

I] FILE AND DIRECTORY COMMANDS

1. **mkdir** command

This command is used to create or make new directory.

Syntax	$ **mkdir** [options] *directoryname*

SN	Example	Meaning
1	$ **mkdir** *cse*	creates a new directory called *"cse"*
2	$ **mkdir** *cse ce me*	creates 3 new directory namely *"cse", "ce", "me"*
3	$ **mkdir – v** *cse*	creates a new directory called *"cse"* in verbose mode(v) it prints confirmation message *directory is created*
4	$ **mkdir –m** mode *directoryname*	Create a new directory with given directory permissions $ mkdir –m 746 cse drwxr--rw- Permissions will be set to cse directory apart from default permissions.

2. **rmdir** command

This command is used to remove empty directories present in your system.

Syntax	$ **rmdir** [options] *directoryname*

SN	Example	Meaning
1	$ **rmdir** *cse*	removes the directory called *"cse"*

3	$ **rmdir – v** *cse*	removes the directory called *"cse"* in verbose mode

3. **touch** command

This command is used to update the timestamp of the file and also to create a file if not present.

Syntax	$ **touch** test.txt

4. **Redirect operator >**

This is also used to create an empty file.

Syntax	$ > test2.txt

5. **cat** command

The cat command is short for **concatenate**. It can be used to output the contents of several files, one file, or even part of a file. It can also be used to create a file when used along with redirect operator.

Syntax	$ **cat** test3.txt $ **cat** > mbak.txt --- --- Ctrl + Z or Ctrl + D Command to append a text to the file. $ **cat** >> mbak.txt --- --- --- Ctrl + Z or Ctrl + D

6. **rm** command

This command is used to delete or remove files.

Syntax	$ **rm** [options] *filename / directoryname*

SN	Example	Meaning
1	$ rm f1.c	removes the file *"f1.c"*
2	$ **rm -i** *f1.c*	prompt before deleting the file *"f1.c"*
3	$ **rm - d** *cse*	removes the directory *"cse"*
4	$ **rm** **.c*	removes all the **c** files

7. **mv** command

This command is used to move or rename the files and directories.

Syntax	$ **mv** [options] *source destination*

SN	Example	Meaning
1	$ **mv** f1.c fourthsem	Moves **f1.c** to the directory ***fourthsem***
2	$ **mv** f1.c f2.c	Renames the file **f1.c** to **f2.c**

8. **cp** command

This command is used to copy the contents of files and directories to other files and directories. If the destination file is not present then the new file is created and copied. If already present then the old content is overwritten with the new one.

Syntax	$ **cp** [options] *source destination*

SN	Example	Meaning
1	$ **cp** f1.c f2.c	Copies the contents of **f1.c** to **f2.c**

Note: The options –i and –d used with **rm** command can also be used with **mv** and **cp** commands.

II]

1. **ls** command

This command is used to list the files and directories present in the system.

Syntax	$ **ls** [options] *filename/dirname*

SN	Example with options	Meaning
1	$ ls	Lists all files & directories in the current directory
2	$ ls -r	Lists all files & directories in reverse order (z-a)
3	$ ls -lh	Lists all files in long format along with file size in human readable format
4	$ ls -t	Lists all files sorted by time and date
5	$ ls -a	Lists all files & also hidden files starting with **.** (dot)
6	$ ls -l	Lists all files & directories in a long list format

2. **comm** command

This command is used to compare two files line by line.

Syntax	$ **comm** [options] *filen1 filen2*

SN	Example	Meaning
1	$ **comm** *f1.txt f2.txt*	Compares the file **f1.txt** with **f2.txt**
2	$ **comm -1** *f1.txt f2.txt*	Compares the files **f1.txt** with **f2.txt** and removes the column 1 from output
3	$ **comm -2** *f1.txt f2.txt*	Compares the files **f1.txt** with **f2.txt** and removes the column 2 from output

Consider the files f1.txt and f2.txt with the below content in it and execute the above command.

f1.txt	**f2.txt**
cheese	butter
curd	cheese
milk	milk

3. **cmp** command

This command is used to compare two files byte by byte. It displays the byte and line number where first difference is found. If no difference is found no output will be displayed.

Syntax	$ **cmp** [options] *file1 file2*

SN	Example	Meaning
1	$ **cmp** *f1.txt f2.txt*	Compares the files *f1.txt* and *f2.txt* ***Output:-*** *differ: byte 15, line 3*

4. **File manipulating** commands

Some of the file manipulating commands are

1. mkdir : used to create directories
2. rmdir : used to remove directories
3. cp : used to copy files and directories
4. mv : used to move or rename files and directories

We have already discussed the above commands, please refer above!

5. **chmod** command

- This command is used for changing the file permissions.
- **chmod** stands for "change mode"
- We can change permission for all 3 categories of user (user, group, others)
- The command can be specified in **two** ways

a) Relative Permissions

When using this method, chmod changes the permissions specified in the command and leaves the other permission unchanged.

Syntax	$ **chmod** [options] permissions *filename*

SN	Example	Output
1	$ **chmod** u + x f.txt	Adds execute permission on file f.txt for the **user**
2	$ **chmod** o - x f.txt	Removes execute permission on file f.txt for the **others**
3	$ **chmod** a + x f.txt	Adds execute permission on file f.txt for the **user, group and others (all)**
4	$ **chmod** o + x f.txt f1.txt	Adds execute permission on file f.txt and f1.txt for the **others**

b) Absolute Permissions

In this method, **chmod** uses 3-digit code.

Digits assigned for **r**ead = 4, **w**rite = 2 and **e**xecute = 1

This command changes the permissions specified in the command by removing

the other permissions.

SN	Example	Output
1	$ **chmod** 777 f.txt	Adds read, write and execute permission on file f.txt for **all**
2	$ **chmod** 766 f.txt	User = r w x Group = r w Others = r w

6. File compression and decompression Commands

a. **gzip, zip & tar:** These commands are used to compress the given file.

Syntax	**Description**	**Example**
gzip {filename}	Gzip compress the size of the given files	**gzip** mydata.doc **gzip** *.jpg **ls –l**

zip {.zip-filename} {filename-to-compress}	zip is used to compress the given file	**zip** mydata.zip mydata.doc
tar -zcvf {.tgz-file} {files} **tar -jcvf {.tbz2-file} {files}**	The GNU tar is archiving utility used to compressing large file(s)	**tar** -zcvf data.tgz *.doc **tar** -zcvf pics.tar.gz *.jpg *.png **tar** -jcvf data.tbz2 *.doc **ls -l**

b. gzip –d, unzip & tar -x: These commands are used to decompress the given file.

Syntax	**Description**	**Example**
gzip -d {.gz file}	Decompress a file that is created using gzip command. File is restored to their original form using this command.	**gzip** -d mydata.doc.gz
unzip {.zip file}	Extract the compressed files	**unzip** mydata.zip
tar -zxvf {.tgz-file} **tar** -jxvf {.tbz2-file}	Untar or decompres a file(s) that is created using tar compressing through gzip and bzip2 filter	**tar** -zxvf data.tgz **tar** -zxvf pics.tar.gz *.jpg **tar** -jxvf data.tbz2

To list the content of the compressed files we can use **–l** option along with gzip and unzip commands.

Example: **gzip –l** mydata.doc.gz and **unzip –l** mydata.zip

III]

1. Text processing commands

Commands, which help in effectively manipulating and processing text files, are called as text processing commands.

Example: sort, uniq, sed, awk, head, tail etc.

i. sort command

This command sorts the content of a file, line by line.

Syntax	$ **sort** [options] filename

Consider a file sort.txt (use tab after each word) with the below content in it.

02 marie
01 joe
04 dave
03 albert

SN	Example	Output
1	$ **sort** *sort.txt*	Sorts the content of the file **sort.txt** 01 joe 02 marie 03 albert 04 dave
2	$ **sort** - r *sort.txt*	Sorts the content of the file **sort.txt** in **reverse order** 04 dave 03 albert 02 marie 01 joe
3	$ **sort** - k2 *sort.txt*	Sorts the content of the file **sort.txt** based on the key (k) 2 representing the second column 03 albert 04 dave 01 joe 02 marie

ii. **uniq** command

This command reports or filters the repeated lines in a file.

Syntax	$ **uniq** [options] *filename*

Consider a file u.txt with the below content in it.

u.txt
this is a line
this is a line
this is a line
this is also a line
this is also a line
this is also also a line

SN	Example	Output
1	$ **uniq** *u.txt*	Displays only single copy of each line present in **u.txt** this is a line this is also a line this is also also a line
2	$ **uniq** - c *u.txt*	Displays only single copy of each line present in **u.txt** along with their count 3 this is a line 2 this is also a line 1 this is also also a line
3	$ **uniq** - d *u.txt*	displays only the lines at are duplicate in **u.txt** this is a line this is also a line

iii. **cut** command

This command removes or cuts the section of each line from a specified file.

Syntax	$ **cut** [options] *filename*

Consider a file cut.txt (use tab after each word) **with the below content in it.**

one two three four five six

apple grapes mango peach berries kiwi

SN	Example	Output
1	$ **cut - f 4** cut.txt	four peach
2	$ **cut - f 2-3** cut.txt	two three grapes mango
3	$ **cut - f 1, 4, 6** cut.txt	one four six apple peach kiwi

In the above examples **f** stands for **field**

iv. **paste** command

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** [options] *file1 file2*

SN	Example	Output
1	$ **paste** *a.txt b.txt*	Displays the content of **a.txt** and **b.txt** in parallel here tab is used as a delimiter
2	$ **paste** - d ':' *a.txt b.txt*	Displays the content of **a.txt** and **b.txt** in parallel here : is used as a delimiter
3	$ **paste** - s *a.txt b.txt*	Displays the content of **a.txt** and **b.txt** one after the another (in serial)

Consider 3 files state, capital and number with the below contents in it.

$ cat state
Arunachal Pradesh
Assam
Andhra Pradesh
Karnataka
Chhattisgrah

$ cat capital
Itanagar
Dispur
Hyderabad
Bangalore
Raipur
And a number file containing 5 numbers in it line by line.

$ paste number state capital
1 Arunachal Pradesh Itanagar
2 Assam Dispur
3 Andhra Pradesh Hyderabad
4 Karnataka Bangalore
5 Chhattisgrah Raipur

v. head command

This command prints the FIRST 10 lines of the specified file

Syntax	$ **head** [options] *filename*

SN	Example	Output
1	$ **head** f1.txt	Prints the first 10 lines of the file **f1.txt**
2	$ **head - 4** f1.txt	Prints only the first 4 lines of the file **f1.txt**
3	$ **head - c 4** f1.txt	Prints only the first 4 characters of the file **f1.txt**

vi. **tail** command

This command prints the LAST 10 lines of the specified file

Syntax	$ **tail** [options] *filename*

SN	Example	Output
1	$ tail f1.txt	Prints the last 10 lines of the file **f1.txt**
2	$ tail - 4 f1.txt	Prints only the last 4 lines of the file **f1.txt**
3	$ **tail - c 4** f1.txt	Prints only the last 4 characters of the file **f1.txt**

vii. **grep** command

- grep stands for **global regular expression print**
- This command processes or analyse the given file line by line and prints any lines that matches the specified pattern.

Syntax	$ **grep** [options] pattern *filename*

- **Regular Expression:** They provide the way for matching patterns in a given input.

SN	Regex (Reg-regular, ex-expression) Patterns	Meaning
1	**. (dot)**	Matches any character expect \n
2	*** (asterisk)**	Matches 0 or more occurrences of regular expression
3	**+ (plus)**	Matches 1 or more occurrences of regular expression
4	**? (question mark)**	Matches 0 or 1 occurrences of regular expression
5	**^ (caret)**	Matches the beginning of a line
6	**\ (backslash)**	Any special character should be preceded by \

Consider a file reg.txt with the below content in it.

hello this is grep demo
good morning my friends
This is abc from 4th sem cse
this is my linuxlab
bye take care this
is abc logging off see you.

SN	Example	Meaning
1	$ **grep** *"linuxlab"* **reg.txt**	Displays/Matches any line that contains the word ***"linuxlab"*** *this is my* ***linuxlab***
2	$ **grep** *"this."* **reg.txt**	Displays any line that contains the word ***"this"*** and ***"this"*** should not be present at end of the line. *hello* ***this*** *is grep demo* ***this*** *is my linuxlab*
3	$ **grep** *"this*"* **reg.txt**	Displays any line that contains the word ***"this"*** *hello* ***this*** *is grep demo* ***this*** *is my linuxlab* *bye take care* ***this***
4	$ **grep** *"[^t]his"* **reg.txt**	Displays any line that contains the spelling ***"his"*** and beginning character of ***"his"*** should not be ***"t"*** T**his** is abc from 4th sem cse

IV] Pipes

- A pipe is a form of redirection (transfer of standard output to some other destination) that is used in Linux and other Unix-like operating systems to send the output of one command/program/process to another command/program/process for further processing.

- The Unix/Linux systems allow stdout of a command to be connected to stdin of another command.
- It can be done by using the pipe character '|'.
- Pipes are unidirectional i.e data flows from left to right through the pipeline.

Syntax	**$ command 1 | command 2 | ….. | command n**

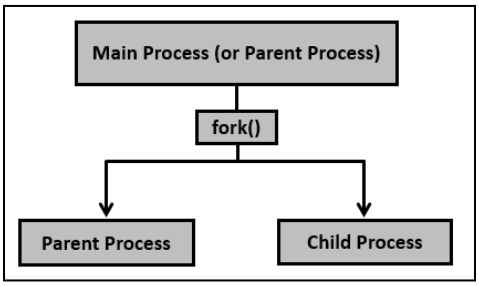
SN	Example	Output
1	$ ls –l | wc -l	Prints the number of lines in the output of ls –l command without displaying the output of ls –l on the screen

=== * ===

04 – Process creation and Management

Process

- A **process** is any active (running) instance of a program. In other words, process is a program in execution.
- A new process can be created **by the fork() system call**.
- The new process consists of a copy of the address space of the original process. fork() creates new process from existing process.
- Existing process is called the **parent process** and the process is created newly is called **child process**.



- Each process is given a unique process identification number (PID).
- PID is usually five-digit number.
    - This number is used to manage each process.
    - user can also use the process name to manage process.

Starting a Process

A process executes in two ways they are,

- Foreground Processes
- Background Processes

Foreground Processes

- Every process by default runs in Foreground (which means the output is printed on the screen.) The best example for the foreground process is the **ls** command, which prints the output on the screen by listing the files and directories.
- When a program is running in the foreground, user cannot start another process without completing the previous process. [It is because of this reason foreground process is considered a time-consuming process.]

Background Processes

- When a process starts running and it is not visible on the screen it is called a background process.
- User can simultaneously run 'n' number of commands in the background process.
- To enable the background process, provide ampersand symbol (**&**) at the end of the command.
- **Example : ls &**

ps

The default output of ps is a simple list of the processes running in current terminal.

- **Example: ps**

  PID TTY TIME CMD

  23989 pts/0 00:00:00 bash

  24148 pts/0 00:00:00 ps

- Every running process (**-e**) and a full listing (**-f**) could be obtained with these options.

=== * ===

I] PROCESS MANAGEMENT SYSTEM CALLS

Write the following Python programs and execute it in the terminal

Process1.py

```
# Importing os module
import os
# Creating child processes using fork() method
os.fork()
print("Welcome to Python Programming – print after
       first fork call")
os.fork()
# This will be executed by both parent & child processes
print("Welcome to Python Programming – print after
       second fork call")
```

```
am2en@am2en-virtual-machine:~$ nano Process1.py
am2en@am2en-virtual-machine:~$ python3 Process1.py
Welcome to Python Programming - print after first fork call
Welcome to Python Programming - print after first fork call
Welcome to Python Programming - print after second fork call
Welcome to Python Programming - print after second fork call
Welcome to Python Programming - print after second fork call
Welcome to Python Programming - print after second fork call
am2en@am2en-virtual-machine:~$
```

Process2.py

```
# Importing os module

import os

# Creating child processes using fork() method

os.fork()

# This will be executed by both parent & child processes

print("Welcome to Python Programming")

print(os.getpid())

print(os.getppid())
```

```
am2en@am2en-virtual-machine:~$ nano Process2.py
am2en@am2en-virtual-machine:~$ python3 Process2.py
Welcome to Python Programming
2386
2230
Welcome to Python Programming
2387
2386
am2en@am2en-virtual-machine:~$
```

=== * ===

II] PROCESS RELATED COMMANDS

1. ps command – **P**rocess **S**tatus

This command is used to get the information about the processes running in the system.

Syntax	$ **ps** [options]

To display the processes running in the current shell use ps command without any options or parameters.

```
am2en@am2en-virtual-machine:~$ ps
    PID TTY          TIME CMD
   2020 pts/0    00:00:00 bash
   2101 pts/0    00:00:00 ps
am2en@am2en-virtual-machine:~$
```

Where,

PID: Process ID

TTY: Terminal TYpe

TIME: Total time the process has been running

CMD: Name of the command that launches the process

SN	Example	Meaning
1	$ **ps** ux	Displays all processes running under a user
2	$ **ps** PID	Displays information about a particular process
3	$ **ps** -el	Displays all processes present in a system

2. sleep command

This command is used to delay for specified amount of time.

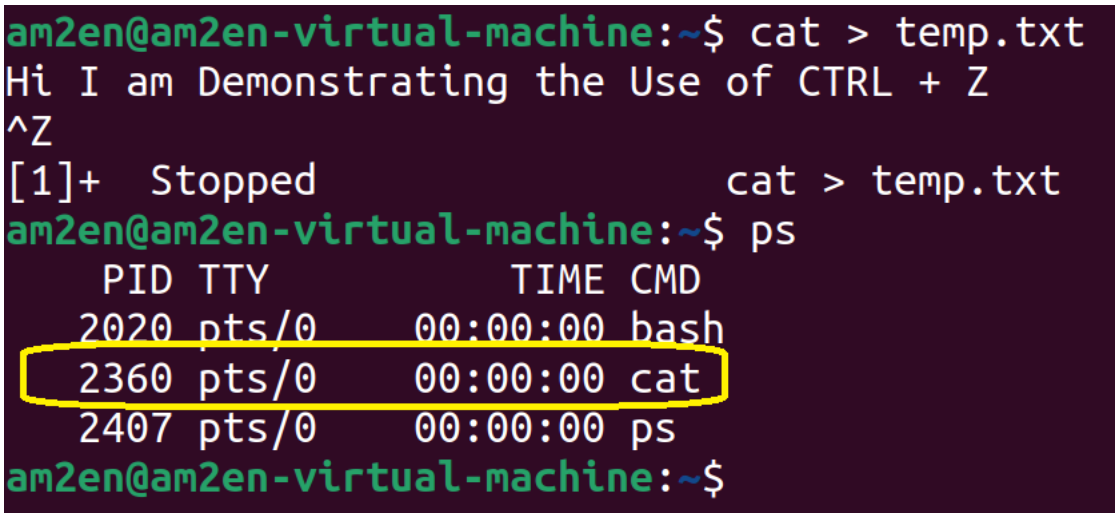
Syntax	$ **sleep** NUMBER [SUFFIX]

SUFFIX may b ‘s’ for seconds, ‘m’ for minutes, ‘h’ for hours and ‘d’ for days.

SN	Example	Meaning
1	$ **sleep** 100	Delays for 100 seconds
2	$ **sleep** 5m	Delays for 5 minutes

3. Difference between **CTRL + Z** and **CTRL + C**

- When we use CTRL + Z to return to command line, it makes the command to go to the stopped or blocked state and also returns the Job number as shown below,



- Similarly when we use CTRL + C, it terminates the command after execution,

```
am2en@am2en-virtual-machine:~$ cat > temp.txt
Hi I am demonstrating the use of CTRL + C
^C
am2en@am2en-virtual-machine:~$ ps
    PID TTY          TIME CMD
   2020 pts/0    00:00:00 bash
   2435 pts/0    00:00:00 ps
am2en@am2en-virtual-machine:~$
```

After using either of the two controls, you can use **ps** command to see the difference.

4. **jobs** command

This command lists all the jobs in the system: active, stopped or otherwise. Before exploring this command, let us create a job using sleep command and stop it by pressing CTRL + Z (as discussed above)

Syntax	$ **jobs**

```
am2en@am2en-virtual-machine:~$ sleep 2m
^Z
[1]+  Stopped                 sleep 2m
am2en@am2en-virtual-machine:~$ jobs
[1]+  Stopped                 sleep 2m
am2en@am2en-virtual-machine:~$
```

5. **fg** command

This command is used to bring a background job or process to foreground.

Syntax	$ **fg** [% Job_Num]

```
am2en@am2en-virtual-machine:~$ cat > temp.txt
Hi Welcome to the World of Linux
^Z
[1]+  Stopped                 cat > temp.txt
am2en@am2en-virtual-machine:~$ sleep 500
^Z
[2]+  Stopped                 sleep 500
am2en@am2en-virtual-machine:~$ jobs
[1]-  Stopped                 cat > temp.txt
[2]+  Stopped                 sleep 500
am2en@am2en-virtual-machine:~$ fg
sleep 500
^Z
[2]+  Stopped                 sleep 500
am2en@am2en-virtual-machine:~$ jobs
[1]-  Stopped                 cat > temp.txt
[2]+  Stopped                 sleep 500
am2en@am2en-virtual-machine:~$ fg %1
cat > temp.txt
I will terminate this..
^C
am2en@am2en-virtual-machine:~$ jobs
[2]+  Stopped                 sleep 500
am2en@am2en-virtual-machine:~$ fg %2
sleep 500
```

6. **bg** command

This command is used to send a job or process to the background and also to list the background jobs available in the current shell.

Syntax	$ **bg** [% Job_Num]

```
am2en@am2en-virtual-machine:~$ bg
[2]+ sleep 500 &
am2en@am2en-virtual-machine:~$ fg "%sleep"
sleep 500
^C
am2en@am2en-virtual-machine:~$ jobs
am2en@am2en-virtual-machine:~$ bg
bash: bg: current: no such job
am2en@am2en-virtual-machine:~$ cat > temp.txt &
[1] 2490
am2en@am2en-virtual-machine:~$ jobs
[1]+  Stopped                 cat > temp.txt
am2en@am2en-virtual-machine:~$ bg
[1]+ cat > temp.txt &
am2en@am2en-virtual-machine:~$ fg
cat > temp.txt
^Z
[1]+  Stopped                 cat > temp.txt
am2en@am2en-virtual-machine:~$ jobs
[1]+  Stopped                 cat > temp.txt
am2en@am2en-virtual-machine:~$
```

- We can also use "&" as a suffix to the command to send the job/process to background as shown above.

NOTE: **bg** and **fg** operate on the current job if no Job_Num is provided.

7. **nohup** command – **no h**ang **up**

This command runs the process even after logging out of the shell/terminal.

Syntax	$ **nohup command** [command-arguments]

SN	Example	Meaning
1	$ **nohup** --version	Checks the version of the **nohup** command
2	$ **nohup cat** Temp.txt	The output of the file Temp.txt is redirected to **nohup.out** file and the standard input (stdin) is not available to the user

```
am2en@am2en-virtual-machine:~$ nohup --version
nohup (GNU coreutils) 8.32
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Jim Meyering.
am2en@am2en-virtual-machine:~$ ls
cse  Desktop  Documents  Downloads  mbak  mbak.txt.gz  Music  Pictures  Public  snap  Templates  temp.txt  Temp.txt  temp.txt.zip  t.txt  Videos
am2en@am2en-virtual-machine:~$ cat Temp.txt
Hi
Welcome
to
Linux Lab
am2en@am2en-virtual-machine:~$ nohup cat Temp.txt
nohup: ignoring input and appending output to 'nohup.out'
am2en@am2en-virtual-machine:~$ ls
cse  Desktop  Documents  Downloads  mbak  mbak.txt.gz  Music  nohup.out  Pictures  Public  snap  Templates  temp.txt  Temp.txt  temp.txt.zip  t.txt  Videos
am2en@am2en-virtual-machine:~$ cat nohup.out
Hi
Welcome
to
Linux Lab
am2en@am2en-virtual-machine:~$
```

8. **top** command

This command is used to show the Linux processes. It provides a dynamic real-time

view of the running system.

Syntax	$ **top**

```
top - 23:50:31 up 26 min,  1 user,  load average: 0.08, 0.05, 0.15
Tasks: 286 total,   1 running, 285 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.2 us,  0.2 sy,  0.0 ni, 98.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   3888.8 total,   1241.1 free,   1761.6 used,    886.0 buff/cache
MiB Swap:    923.2 total,    923.2 free,      0.0 used.   1853.9 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1279 am2en     20   0 4293184 269436 135144 S   1.3   6.8   0:37.04 gnome-shell
   2122 am2en     20   0  566584  65440  52112 S   0.7   1.6   0:07.05 gnome-terminal-
   2338 am2en     20   0   13240   4236   3376 R   0.7   0.1   0:00.19 top
     63 root      20   0       0      0      0 I   0.3   0.0   0:01.23 kworker/1:4-mm_percpu_wq
    644 root      20   0  243628   8616   7300 S   0.3   0.2   0:05.89 vmtoolsd
   1682 am2en     20   0  288636  38628  29820 S   0.3   1.0   0:06.54 vmtoolsd
      1 root      20   0  166836  11932   8216 S   0.0   0.3   0:06.48 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.07 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      7 root      20   0       0      0      0 I   0.0   0.0   0:02.36 kworker/0:0-events
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
     11 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_kthread
     12 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_rude_kthread
     13 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_trace_kthread
     14 root      20   0       0      0      0 S   0.0   0.0   0:00.22 ksoftirqd/0
     15 root      20   0       0      0      0 I   0.0   0.0   0:01.22 rcu_preempt
     16 root      rt   0       0      0      0 S   0.0   0.0   0:00.02 migration/0
     17 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
     19 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
     20 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
     21 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/1
     22 root      rt   0       0      0      0 S   0.0   0.0   0:01.67 migration/1
     23 root      20   0       0      0      0 S   0.0   0.0   0:00.19 ksoftirqd/1
     25 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/1:0H-events_highpri
```

Here,

- **PID**: Shows task's unique process id.
- **USER**: User name of owner of task.
- **PR**: The process's priority. The lower the number, the higher the priority.
- **NI**: Represents a Nice Value of task. A Negative nice value implies higher priority, and positive Nice value means lower priority.
- **VIRT**: Total virtual memory used by the task.
- **RES**: How much physical RAM the process is using, measured in kilobytes.
- **SHR**: Represents the Shared Memory size (kb) used by a task.
- **%CPU**: Represents the CPU usage.
- **%MEM**: Shows the Memory usage of task.
- **TIME+**: CPU Time, the same as 'TIME', but reflecting more granularity through hundredths of a second.
- **COMMAND**: The name of the command that started the process.

9. **kill** command

This command is used to kill a process running/stopped in the Linux System.

Syntax	$ **kill** PID

```
am2en@am2en-virtual-machine:~$ cat > Killdemo.txt
Let us learn how to kill a process. But before that lets create a simple cat process.
^Z
[1]+  Stopped                 cat > Killdemo.txt
am2en@am2en-virtual-machine:~$ ps
    PID TTY          TIME CMD
   2452 pts/0    00:00:00 bash
   2496 pts/0    00:00:00 cat
   2587 pts/0    00:00:00 ps
am2en@am2en-virtual-machine:~$ kill 2496
am2en@am2en-virtual-machine:~$ ps
    PID TTY          TIME CMD
   2452 pts/0    00:00:00 bash
   2496 pts/0    00:00:00 cat
   2592 pts/0    00:00:00 ps
am2en@am2en-virtual-machine:~$ fg
cat > Killdemo.txt
Terminated
am2en@am2en-virtual-machine:~$
```

Note: If a process is in stopped or blocked state, then that process can also be killed but to if must bought foreground to validate it.

10. **pkill** Command

This command works similar to **kill** command but instead uses process name (either

full or partial).

Syntax	$ **pkill** Process_Name

```
am2en@am2en-virtual-machine:~$ cat > pkill.txt
Let us create a file just to demonstarte the use of pkill command.
^Z
[1]+  Stopped                 cat > pkill.txt
am2en@am2en-virtual-machine:~$ ps
    PID TTY          TIME CMD
   2948 pts/1    00:00:00 bash
   2984 pts/1    00:00:00 cat
   3002 pts/1    00:00:00 ps
am2en@am2en-virtual-machine:~$ pkill cat
am2en@am2en-virtual-machine:~$ ps
    PID TTY          TIME CMD
   2948 pts/1    00:00:00 bash
   2984 pts/1    00:00:00 cat
   3005 pts/1    00:00:00 ps
am2en@am2en-virtual-machine:~$ fg
cat > pkill.txt
Terminated
am2en@am2en-virtual-machine:~$ ps
    PID TTY          TIME CMD
   2948 pts/1    00:00:00 bash
   3006 pts/1    00:00:00 ps
am2en@am2en-virtual-machine:~$
```

11. **nice** command

In Linux, each process has a **nice value** granted to it. This value influences the scheduling of processes and thereby determining the amount of CPU to spent on each one of them. *"The more the nice value, the less the supposed priority"*.

The nice value ranges from **-20 to 19** where -20 means a process is not nice, therefore must be given more priority, and 19 is the nicest process, therefore allows other processes to access resources before itself.

The two adjoining columns highlighted in green colour as shown in command no. 8 denote the priority-related values for every process. The **PR** column denotes the actual kernel-given priority of the process, whereas the **NI** column provides the Nice Value for each process.

The nice command without any parameters display the nice value of the bash running on the terminal. In addition, the default value is set to zero. This implies that any command running using this terminal will have the same nice value.

Syntax	$ **sudo nice** –n <VALUE> <COMMAND>

Execute the below commands in sequence to change the priority and nice value of a process

1. $ nice
2. $ top

```
am2en@am2en-virtual-machine:~$ nice
0
am2en@am2en-virtual-machine:~$ top
```

```
top - 12:22:02 up 13 min,  2 users,  load average: 0.03, 0.17, 0.23
Tasks: 295 total,   1 running, 291 sleeping,   3 stopped,   0 zombie
%Cpu(s):  5.2 us,  2.5 sy,  0.0 ni, 91.9 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0 st
MiB Mem :   3888.8 total,   1010.5 free,   1824.6 used,   1053.7 buff/cache
MiB Swap:    923.2 total,    923.2 free,      0.0 used.   1781.4 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1225 am2en     20   0 4375812 274320 147696 S   9.6   6.9   0:25.34 gnome-shell
   2158 am2en     20   0  578320  68812  52308 S   5.3   1.7   0:04.72 gnome-terminal-
   2227 am2en     20   0   13384   4240   3380 R   0.7   0.1   0:00.18 top
    292 root      20   0       0      0      0 I   0.3   0.0   0:00.66 kworker/u256:26-events_unbound
    295 root      20   0       0      0      0 I   0.3   0.0   0:00.81 kworker/u256:29-events_freezable_power_
    593 root      20   0       0      0      0 I   0.3   0.0   0:01.07 kworker/0:4-events
```

From the above output, the priority of top and other commands is PR = 20 and NI = 0 which is a default priority of bash shell.

Let us change the nice value using nice command

```
am2en@am2en-virtual-machine:~$ sudo nice -n 10 top
```

```
top - 12:28:19 up 19 min,  3 users,  load average: 0.19, 0.12, 0.17
Tasks: 296 total,   1 running, 291 sleeping,   4 stopped,   0 zombie
%Cpu(s):  0.7 us,  0.2 sy,  0.2 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   3888.8 total,   1006.0 free,   1828.4 used,   1054.4 buff/cache
MiB Swap:    923.2 total,    923.2 free,      0.0 used.   1777.8 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   2158 am2en     20   0  577272  67896  52040 S   0.7   1.7   0:08.49 gnome-terminal-
   2237 root      30  10   13384   4244   3380 R   0.7   0.1   0:00.45 top
    593 root      20   0       0      0      0 I   0.3   0.0   0:01.53 kworker/0:4-events
    611 systemd+  20   0   14828   6220   5428 S   0.3   0.2   0:02.07 systemd-oomd
    676 root      20   0  169896   8448   7156 S   0.3   0.2   0:03.75 vmtoolsd
   1225 am2en     20   0 4374132 274100 147428 S   0.3   6.9   0:32.59 gnome-shell
   1593 am2en     20   0  288872  38832  29900 S   0.3   1.0   0:03.78 vmtoolsd
   2129 root      20   0       0      0      0 I   0.3   0.0   0:00.70 kworker/1:0-events
      1 root      20   0  102672  13448   8272 S   0.0   0.3   0:05.06 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.04 kthreadd
```

The nice command modifies the nice value of the process relatively. If u want to absolutely modify the nice value use **renice** command.

Syntax	$ **sudo renice** –n <VALUE> -p <PID>

```
am2en@am2en-virtual-machine:~$ ps
    PID TTY          TIME CMD
   2363 pts/3    00:00:00 bash
   2394 pts/3    00:00:00 top
   2396 pts/3    00:00:00 ps
am2en@am2en-virtual-machine:~$ sudo renice -n 15 -p 2394
[sudo] password for am2en:
2394 (process ID) old priority 0, new priority 15
am2en@am2en-virtual-machine:~$
```

```
top - 12:37:59 up 29 min,  3 users,  load average: 0.21, 0.13, 0.14
Tasks: 296 total,   1 running, 292 sleeping,   3 stopped,   0 zombie
%Cpu(s):  1.2 us,  0.7 sy,  0.0 ni, 98.0 id,  0.0 wa,  0.0 hi,  0.2 si,  0.0 st
MiB Mem :   3888.8 total,    980.5 free,   1850.9 used,   1057.4 buff/cache
MiB Swap:    923.2 total,    923.2 free,      0.0 used.   1755.1 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   2223 root      20   0   14588   2600   1580 S   0.0   0.1   0:00.00 sudo
   2224 root      30  10   13224   4244   3380 T   0.0   0.1   0:00.31 top
   2231 root      20   0       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0-rcu_par_gp
   2235 root      20   0   14424   5972   5056 S   0.0   0.1   0:00.10 sudo
   2236 root      20   0   14424    968      0 S   0.0   0.0   0:00.00 sudo
   2237 root      30  10   13384   4244   3380 T   0.0   0.1   0:02.03 top
   2238 root      20   0       0      0      0 I   0.0   0.0   0:00.25 kworker/u256:2-events_power_efficient
   2243 root      20   0       0      0      0 I   0.0   0.0   0:00.00 kworker/1:2-rcu_par_gp
   2288 am2en     20   0 2805272  70560  55460 S   0.0   1.8   0:00.80 gjs
   2337 am2en     20   0   35956  19316  11604 S   0.0   0.5   0:00.10 gnome-terminal
   2340 am2en     20   0  383124  27532  20028 S   0.0   0.7   0:00.18 gnome-terminal.
   2363 am2en     20   0   11136   5312   3832 S   0.0   0.1   0:00.03 bash
   2394 am2en     35  15   13240   4232   3368 T   0.0   0.1   0:00.04 top
```

Syntax	$ **sudo nice** –n <VALUE> bash

```
am2en@am2en-virtual-machine:~$ nice
0
am2en@am2en-virtual-machine:~$ sudo nice -n 12 bash
root@am2en-virtual-machine:/home/am2en# nice
12
root@am2en-virtual-machine:/home/am2en# top
```

```
top - 12:42:03 up 33 min,  4 users,  load average: 0.07, 0.08, 0.10
Tasks: 300 total,   1 running, 295 sleeping,   4 stopped,   0 zombie
%Cpu(s):  0.7 us,  0.5 sy,  0.2 ni, 98.5 id,  0.0 wa,  0.0 hi,  0.2 si,  0.0 st
MiB Mem :   3888.8 total,    972.1 free,   1857.9 used,   1058.7 buff/cache
MiB Swap:    923.2 total,    923.2 free,      0.0 used.   1748.0 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1593 am2en     20   0  289592  39520  29900 S   0.7   1.0   0:06.55 vmtoolsd
   2345 am2en     20   0  576884  67448  52036 S   0.7   1.7   0:08.44 gnome-terminal-
   2452 root      32  12   13420   4240   3376 R   0.7   0.1   0:00.19 top
    676 root      20   0  169896   8448   7156 S   0.3   0.2   0:06.36 vmtoolsd
   1225 am2en     20   0 4389068 282684 147624 S   0.3   7.1   0:52.20 gnome-shell
   2129 root      20   0       0      0      0 I   0.3   0.0   0:01.62 kworker/1:0-events
      1 root      20   0  102672  13448   8272 S   0.0   0.3   0:05.17 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.04 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
```

=== * ===

III] COMMANDS TO SCHEDULE TASKS

1. Cron Command

- The cron can be defined as a *software utility* that is provided by a Linux-like OS that operates the scheduled operation on a predetermined period.
- It's a *daemon process* and executes as a *background process.*
- It performs the described tasks at a predefined period when a condition or event is encountered without user intervention.

Cron Table Format

```
*    *    *   *    *  Command_to_execute
|    |    |    |   |
|    |    |    |    Day of the Week ( 0 - 6 ) ( Sunday = 0 )
|    |    |    |
|    |    |    Month ( 1 - 12 )
|    |    |
|    |    Day of Month ( 1 - 31 )
|    |
|    Hour ( 0 - 23 )
|
Min ( 0 - 59 )
```

To install **cron**

- **sudo apt-get install cron**

Setting up the Cron Operation

We need to open the *crontab* along with a text editor for configuring the *cron* operation and enter the syntax for a command we wish to execute.

Editing a crontab file

We need to type a command in our terminal window for opening the configuration file of the crontab for a current user as shown below,

- **crontab -e**

Select any of the 3 options displayed to open the editor

```
  GNU nano 6.2                                        /tmp/crontab.yyEcxF/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command

# The below command will be executed every minute
* * * * * echo "GPTM CSE" >> /home/am2en/crondemo.txt

# The below command will be executed at 10:00 AM daily
0 10 * * * echo "GPTM" >> /home/am2en/crondemo1.txt
```

Press ctrl+x then y then enter to save and exit.

```
am2en@am2en-virtual-machine:~$ cat crondemo.txt
GPTM CSE
GPTM CSE
GPTM CSE
GPTM CSE
GPTM CSE
am2en@am2en-virtual-machine:~$
```

- Display ("list") the contents of your crontab.

  crontab –l

- To remove your crontab, effectively un-scheduling all crontab jobs.

  crontab -r

2. **at** command

- The at command is a Linux command-line utility used to schedule a job for later execution.
- The utility reads commands from standard input and groups them into an **at job**, which executes only once.
- The alternative for **at** is a **cron job**.
- However, while at jobs execute only once, cron jobs are recurring events.

To install **at**

- sudo apt-get install at

```
am2en@am2en-virtual-machine:~$ at 10:25
warning: commands will be executed using /bin/sh
at Sun Jun  4 10:25:00 2023
at> echo "Demonstrating at command to execute at 10:25 AM" >> /home/am2en/atdemo.txt
at> ls -l | grep atdemo.txt >> /home/am2en/atdemo.txt
at> <EOT>
job 18 at Sun Jun  4 10:25:00 2023
am2en@am2en-virtual-machine:~$ ls -l | grep atdemo.txt
-rw-rw-r-- 1 am2en am2en  102 Jun  4 10:25 atdemo.txt
am2en@am2en-virtual-machine:~$ cat atdemo.txt
Demonstrating at command to execute at 10:25 AM
-rw-rw-r-- 1 am2en am2en   48 Jun  4 10:25 atdemo.txt
am2en@am2en-virtual-machine:~$ atq
15      Sun Jun  4 10:12:00 2023 a am2en
am2en@am2en-virtual-machine:~$ atrm 15
am2en@am2en-virtual-machine:~$ atq
am2en@am2en-virtual-machine:~$
```

- Checking the queue

  You can look at the queue of at jobs with the **atq** (at queue) command:

- If you need to cancel one of the jobs in the queue, use the **atrm** (at remove) command along with the job number.

The commands related to **at** are shown above.

at vs cron

- The **at** command is something like **cron** in that you can schedule tasks to run at a selected time, but **cron** is used for jobs that are run periodically – even if that means only once a year.
- Most **cron** jobs are set up to be run daily, weekly or monthly, though you control how often and when.
- The **at** command, on the other hand, is used for tasks which are run only once.
- Want to reboot your system at midnight tonight? No problem, **at** can do that for you assuming you have the proper permissions.
- If you want the system rebooted every Saturday night at 2:00AM., use **cron** instead.

=== * ===

05 – Process Synchronization

Commands to exhibit thread concepts:

- Threads are a popular programming abstraction for parallel execution on modern operating systems.
- When threads are forked inside a program for multiple flows of execution, these threads share certain resources (e.g., memory address space, open files) among themselves to minimize forking overhead and avoid expensive IPC (inter-process communication) channel.
- These properties make threads an efficient mechanism for concurrent execution.
- In Linux, threads (also called Lightweight Processes (LWP)) created within a program will have the same "thread group ID" as the program's PID.
- Each thread will then have its own thread ID (TID).
- Threads are nothing more than standard processes, which happen to share certain resources.
- Classic command-line tools such as ps or top, which display process-level information by default, can be instructed to display thread-level information.

Using the ps

The "-T" option for the ps command enables thread views.

ps -T -p <pid>

Example: List the threads for the **mate-terminal** process:

- First find the all process containing word **terminal**

```
admincs-VirtualBox ~ # ps -ef | grep terminal
admincs   2400     1  2 19:06 ?        00:00:02 mate-terminal
root      2562  2418  0 19:08 pts/0    00:00:00 grep --colour=auto terminal
admincs-VirtualBox ~ # ps -T -p 2400
  PID  SPID TTY          TIME CMD
 2400  2400 ?        00:00:02 mate-terminal
 2400  2402 ?        00:00:00 gdbus
 2400  2403 ?        00:00:00 dconf worker
 2400  2406 ?        00:00:00 gmain
admincs-VirtualBox ~ #
```

- The "**SPID**" column represents **thread IDs**, and "CMD" column shows thread names.

Using the top

The top command can show a real-time view of individual threads.

To enable thread views in the top output, invoke top with "-H" option. This will list all Linux threads.

User can also toggle on or off thread view mode while top is running, by pressing 'H' key.

Example 1: top

```
top - 19:24:16 up 22 min,  2 users,  load average: 0.36, 0.17, 0.31
Threads: 312 total,   1 running, 311 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  6.6 sy,  0.0 ni, 91.1 id,  1.7 wa,  0.0 hi,  0.3 si,  0.0 st
KiB Mem:   1025964 total,   711340 used,   314624 free,    75752 buffers
KiB Swap:  1046524 total,        0 used,  1046524 free.   401252 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1572 root      20   0  103168  36612  11884 S  2.9  3.6   0:45.92 Xorg
 2664 root      20   0    5564   1568   1084 R  2.0  0.2   0:03.39 top
 2400 admincs   20   0  292884  22356  14896 S  0.7  2.2   0:22.41 mate-terminal
    4 root      20   0       0      0      0 S  0.3  0.0   0:03.78 kworker/0:0
    6 root      20   0       0      0      0 S  0.3  0.0   0:02.46 kworker/u2:0
    7 root      20   0       0      0      0 S  0.3  0.0   0:03.56 rcu_sched
```

To restrict the top output to a particular process and check all threads running inside the process:

Example 2: top -H -p 1572

```
top - 19:26:00 up 24 min,  2 users,  load average: 0.30, 0.18, 0.30
Threads:   1 total,   0 running,   1 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  8.4 sy,  0.0 ni, 91.2 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:   1025964 total,   711124 used,   314840 free,    75784 buffers
KiB Swap:  1046524 total,        0 used,  1046524 free.   401256 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1572 root      20   0  103168  36612  11884 S  4.0  3.6   0:48.50 Xorg
```

=== * ===

06 – Memory Management

Commands to view memory consumption:

1. free command

free –m

```
am2en@am2en-virtual-machine:~$ free -m
               total        used        free      shared  buff/cache   available
Mem:            3888        1981         193          42        1713        1587
Swap:            923          23         899
am2en@am2en-virtual-machine:~$ free -g
               total        used        free      shared  buff/cache   available
Mem:               3           1           0           0           1           1
Swap:              0           0           0
am2en@am2en-virtual-machine:~$ free -b
               total        used        free      shared  buff/cache   available
Mem:      4077703168  2077995008   202694656    44654592  1797013504  1664364544
Swap:      968056832    24539136   943517696
am2en@am2en-virtual-machine:~$
```

- The m option displays all data in MBs, g in GBs and b in Bytes.
- The total 3888 MB is the total amount of RAM reserved for the virtual machine on the system that is 3GB.
- The used column shows the amount of RAM that has been used by linux, in this case around 1 GB.
- Linux has the habit of caching lots of things for faster performance, so that memory can be freed and used if needed.
- The last line is the swap memory, which in this case is lying entirely free.
- **free –s 5** used to execute the free command every five seconds.

2. /proc/meminfo

- Check memory usage by reading the file at **/proc/meminfo**.
- Know that the /proc file system does not contain **real files**.
    - They are rather **virtual files** that contain dynamic information about the kernel and the system.

    Portion of output is

```
am2en@am2en-virtual-machine:~$ cat meminfo.txt
MemTotal:        3982132 kB
MemFree:          153364 kB
MemAvailable:    1581620 kB
Buffers:           26208 kB
Cached:          1563428 kB
SwapCached:          908 kB
Active:           788144 kB
Inactive:        1386292 kB
Active(anon):       4328 kB
Inactive(anon):   618412 kB
Active(file):     783816 kB
Inactive(file):   767880 kB
```

- Check the values of MemTotal, MemFree, Buffers, Cached, SwapTotal, SwapFree. They indicate same values of memory usage as the free command.

3. vmstat

- **vmstat** command in Linux/Unix is a performance monitoring command of the system as it gives the information about processes, memory, paging, block IO, disk, and CPU scheduling.
- All these functionalities makes the command vmstat also known as **virtual memory statistic reporter**.

```
am2en@am2en-virtual-machine:~$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0  23964 153112  26512 1729592    0    2   183   401  144  239  8  4 87  0  0
am2en@am2en-virtual-machine:~$ vmstat -s
      3982132 K total memory
      2072868 K used memory
       788588 K active memory
      1386504 K inactive memory
       153112 K free memory
        26520 K buffer memory
      1729632 K swap cache
       945368 K total swap
        23964 K used swap
       921404 K free swap
        69455 non-nice user cpu ticks
        52721 nice user cpu ticks
        54018 system cpu ticks
      1279441 idle cpu ticks
         2185 IO-wait cpu ticks
            0 IRQ cpu ticks
         4364 softirq cpu ticks
            0 stolen cpu ticks
      2675078 pages paged in
      5864701 pages paged out
          528 pages swapped in
         6524 pages swapped out
      2102337 interrupts
      3487293 CPU context switches
   1685848992 boot time
        42765 forks
am2en@am2en-virtual-machine:~$
```

4. top command

- The top command is generally used to check memory and cpu usage per process.
- It also reports total memory usage and can be used to monitor the total RAM usage.
- The header on output has the required information.
- Check the KiB Mem and KiB Swap lines on the header. They indicate total, used and free amounts of the memory. The buffer and cache information is present here too.

- **Portion of output is**

```
am2en@am2en-virtual-machine:~$ top >> /home/am2en/topdemo.txt
am2en@am2en-virtual-machine:~$ cat /home/am2en/topdemo.txt
```

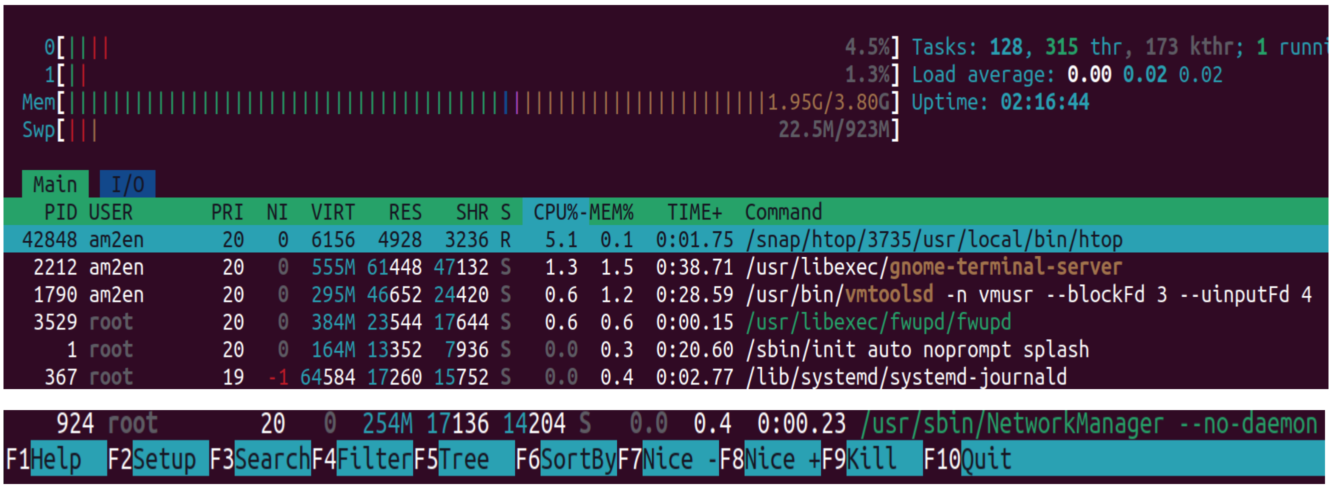
```
top - 11:04:51 up  2:11,  1 user,  load average: 0.07, 0.03, 0.04
Tasks: 301 total,   1 running, 296 sleeping,   4 stopped,   0 zombie
%Cpu(s):  1.3 us,  0.6 sy,  0.0 ni, 97.9 id,  0.0 wa,  0.0 hi,  0.1 si,  0.0 st
MiB Mem :   3888.8 total,    162.3 free,   2008.7 used,   1717.8 buff/cache
MiB Swap:    923.2 total,    899.8 free,     23.4 used.   1560.1 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1388 am2en     20   0 4467168 254748 111656 S   1.2   6.4   2:07.24 gnome-shell
    702 root      20   0  243636   8080   6804 S   0.9   0.2   0:28.50 vmtoolsd
   2212 am2en     20   0  568656  61448  47132 S   0.6   1.5   0:36.96 gnome-terminal-
  42774 am2en     20   0   13236   4312   3452 R   0.6   0.1   0:00.07 top
     15 root      20   0       0      0      0 I   0.3   0.0   0:06.99 rcu_preempt
    626 systemd+  20   0   14828   5976   5188 S   0.3   0.2   0:14.91 systemd-oomd
    883 root      20   0    2812   1144   1048 S   0.3   0.0   0:00.60 acpid
   1273 am2en     20   0   10612   6836   4144 S   0.3   0.2   0:05.46 dbus-daemon
   1336 am2en     39  19  707184  27788  14696 S   0.3   0.7   0:16.49 tracker-miner-f
   1790 am2en     20   0  302172  46652  24420 S   0.3   1.2   0:27.37 vmtoolsd
   1855 am2en     20   0  163012   6692   6084 S   0.3   0.2   0:00.37 gvfsd-metadata
  42496 root      20   0       0      0      0 I   0.3   0.0   0:01.22 kworker/0:2-events
  42525 am2en     20   0 2805244  70604  55192 S   0.3   1.8   0:01.86 gjs
  42560 root      20   0       0      0      0 I   0.3   0.0   0:00.90 kworker/1:0-events
  42608 root      20   0       0      0      0 I   0.3   0.0   0:00.40 kworker/u256:2-events_unbound
      1 root      20   0  168296  13352   7936 S   0.0   0.3   0:20.55 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.30 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
     11 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_kthread
     12 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_rude_kthread
     13 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_trace_kthread
     14 root      20   0       0      0      0 S   0.0   0.0   0:00.89 ksoftirqd/0
```

5. htop

- **htop** command in Linux system is a command line utility that allows the user to interactively monitor the system's vital resources or server's processes in real time.
- ***htop*** is a newer program compared to top command, and it offers many improvements over top command.
- **htop** supports mouse operation, uses color in its output and gives visual indications about processor, memory and swap usage.

- **htop** also prints full command lines for processes and allows one to scroll both vertically and horizontally for processes and command lines respectively.



=== * ===

07 – Shell Programming

Steps to write and execute a script

- Open the terminal. Go to the directory where you want to create your script.
- Create a file with **.sh** extension use any of the editors like gedit, nano, vi etc.,
- Write the script in the file using an editor.
- Make the script executable with command **chmod u+x** <**fileName**>.
- Run the script using ./<**fileName**>

Script 1: Biggest of Three Numbers using if-elif-fi

```
# Find biggest of Three numbers
#!/bin/bash
echo Enter Three Numbers
read a b c

if (( $a>=$b && $a>=$c ))
then
        big=$a
elif(( $b>=$a && $b>=$c  ))
then
        big=$b
elif(( $c>=$a && $c>=$b  ))
then
        big=$c
fi
echo Biggest among $a $b $c is $big
```

=== * ===

File Handling Scripts:

Script 2: Read the contents of a file and display the same on Screen (Demo cat command)

```
read -p "Enter file name : " filename
while read line
do
        echo $line
done < $filename
```

=== * ===

Script 3: Count number of lines, words, and Characters in given file.

```
echo Enter the filename
read file
c=`cat $file | wc -c`
w=`cat $file | wc -w`
l=`grep -c "." $file`
echo Number of characters in $file is $c
echo Number of words in $file is $w
echo Number of lines in $file is $l
```

=== * ===

Script 4: Write a shell script to perform string operations

```
while :
do
echo " ********************************* "
echo "Menu"
echo "1.to find the length of string "
echo "2.Concatenate Strings "
echo "3. compare two strings"
```

```
echo "4.exit "
echo "Enter your choice: "
read ch
case $ch in
1) echo "enter first string"
read s1
len=`expr length "$s1"`
echo "Length of '$s1' is $len"
;;
2) echo "enter first string"
read s1
echo "enter second string"
read s2
echo "the concated string is" $s1$s2
;;
3) echo "enter first string"
read s1
echo "enter second string"
read s2
if [ $s1 = $s2 ]
then
echo "strings are equal"
else
echo "strings are not equal"
fi
;;
4) echo "Exit..."
exit
```

```
;;
esac
done
```

=== * ===

Script 5: Write a shell script to perform file operations

```
while :
do
echo "Menu"
echo "1.copy a file "
echo "2.rename a file "
echo "3.remove a file "
echo "4.exit "
echo "Enter your choice: "
read ch
case $ch in
1) echo "enter file name to create a copy"
read f1
echo "enter the file name for a copy"
read f2
if [ -f $f1 ]
then
cp $f1 $f2
echo "$f1 is copied to $f2"
else
echo " $f1 does not exit"
fi
;;
2) echo "enter file name to rename"
```

```
read f1
echo "enter the new file name for a file"
read f2
if [ -f $f1 ]
then
mv $f1 $f2
echo "$f1 is renamed to $f2"
else
echo " $f1 does not exit"
fi
;;
3) echo "enter file name to be removed"
read f1
if [ -f $f1 ]
then
rm -i $f1
echo "$f1 is removd"
else
echo " $f1 does not exit"
fi
;;
4) echo "Exit..."
exit
;;
esac
done
```

=== * ===

Script 5: Write a shell script to check memory status

```
#!/bin/bash
echo `date`
#---mem
mem_usage () {
 #MB units
mem_free=`free -m | grep "Mem" | awk '{print $3}'`
 echo "memory space remaining : $mem_free MB"

if [ $mem_free -lt 0 ]
   then
      echo "mem warning!!!"
   else
      echo "mem is free!!!"
fi
}
```

=== * ===

08 – Automation of System Tasks

Cron Command to automate System Tasks

- cron processes will run in background, so to check whether scripts are executed or not, one way to check the log file i.e. /var/log/syslog
- Search for cron in syslog using command: **cat /var/log/syslog/ | grep cron**
- User can see all the entries, which shows, the scripted executed at a scheduled time mentioned in crontab file.

=== * ===

Simple Example:

File Name: task.sh path: **/home/am2en/** (say for example)

Contents:

```
echo  Welcome to Task Scheduler Demo
echo  Try date Command
date
echo  List all file/folders in a home directory
ls
echo  End of Task Scheduler Demo
```

=== * ===

- Change the execution permission to task.sh

  chmod 764 task.sh

- Add the task to /etc/crontab (Task will be executed at 4.13PM, User can Change timings according to the requirements)

  13 16 * * * root sh /home/am2en/task.sh > /home/am2en/output.txt

  Or

  13 16 * * * .task.sh >> output.txt

- Save and Exit. And Wait till 4.13pm

- After that Check the output.txt by the command

  cat /home/am2en/output.txt

- Output of the task.sh is present in output.txt.
- Alternatively, **/var/log/syslog** can be verified about the execution of the Task.
- Similarly you can create any shell script based on your requirement and automate it using **cron**.

=== * ===

09 – Network Management

ifconfig:

- **ifconfig**(interface configuration) is used to configure the kernel-resident network interfaces.
- It is used at the boot time to set up the interfaces as necessary.
- It is used to assign the IP address and netmask to an interface or to enable or disable a given interface.
- **ifconfig -a**

  -a : This option is used to display all the interfaces available, even if they are down.

iwconfig:

- **iwconfig** command in Linux is like **ifconfig** command, in the sense it works with kernel-resident network interface but it is dedicated to wireless networking interfaces only.
- It is used to set the parameters of the network interface that are particular to the wireless operation like SSID, frequency etc.
- *iwconfig* may also be used to display the parameters, and the wireless statistics which are extracted from */proc/net/wireless*.

ethtool:

- The **ethtool** is a command-line tool in Linux for managing network interface devices**.**
- It allows us to modify the parameters of the devices and query the information of those devices.
- If the tool is not installed then you can install it using below command

  $ sudo apt-get install –y ethtool
- The portion of the output is shown below.

```
am2en@am2en-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.233.128  netmask 255.255.255.0  broadcast 192.168.233.255
        inet6 fe80::1586:70cc:606a:4b20  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:71:db:d4  txqueuelen 1000  (Ethernet)
        RX packets 3618  bytes 3760066 (3.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 982  bytes 107477 (107.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 297  bytes 33150 (33.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 297  bytes 33150 (33.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

am2en@am2en-virtual-machine:~$ sudo ethtool --version
[sudo] password for am2en:
ethtool version 5.16
am2en@am2en-virtual-machine:~$ ethtool ens33
Settings for ens33:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supported pause frame use: No
        Supports auto-negotiation: Yes
        Supported FEC modes: Not reported
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Advertised pause frame use: No
        Advertised auto-negotiation: Yes
        Advertised FEC modes: Not reported
        Speed: 1000Mb/s
        Duplex: Full
```

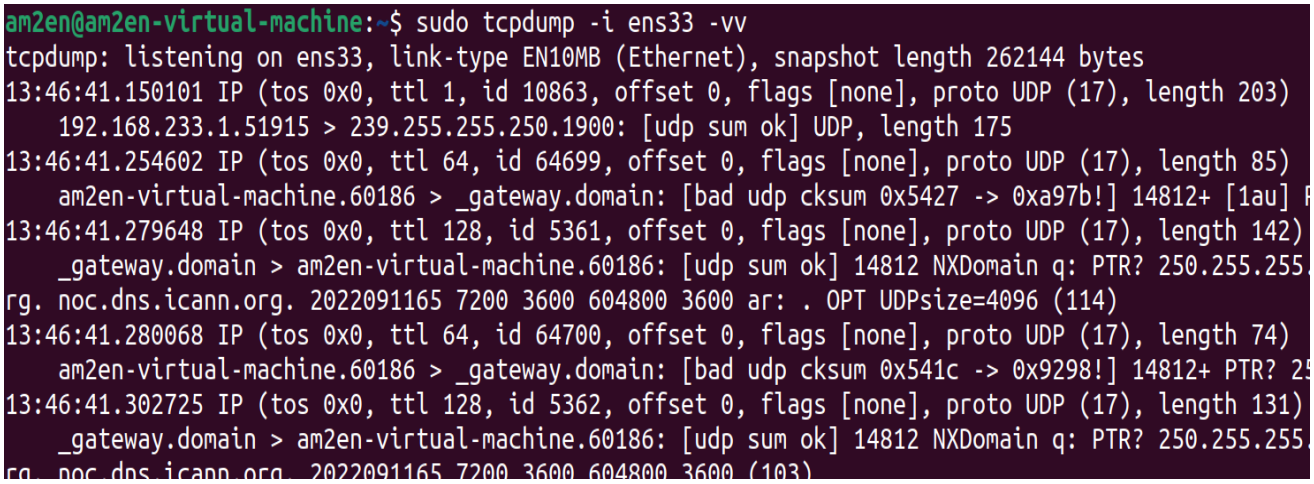
tcpdump:

- The **tcpdump** command captures network traffic on a Linux system.
- It's a versatile command line utility that network administrators often rely on for troubleshooting.
- **tcmpdump** makes our job a little easier by allowing us to isolate only the traffic we're interested in.

- If the tool is not installed then you can install it using below command

  $ sudo apt install tcpdump

- The portion of the output is shown below.



Note: ens33 is the network interface, which may vary across systems. So use the appropriate network interface name in the commands. It can be known from **ifconfig** command as shown above.

=== * ===

10 – User Authentication

Working on user accounts:

useradd

- **useradd** is a command in Linux that is used to add user accounts to the system.
- When we run the '**useradd**' command in the Linux terminal, it performs the following major things:
  - It edits **/etc/passwd**, **/etc/shadow**, **/etc/group** and **/etc/gshadow** files for the newly created user accounts.
  - Creates and populates a home directory for the new user.
  - Sets permissions and ownerships to the home directory.
- **Syntax:**
  - **useradd -d /home/newusername –p passwordstring newusername**
  - This creates user with name newusername
    - -d option will create directory for new user at /home directory
    - -p option allows specifying password while creating user with the value given in place of passwordstring
  - User can change the password later by typing command
    - passwd newusername
    - This asks the user to enter the new password and confirm password

passwd

- The passwd command **changes passwords for user accounts**.
- A normal user may only change the password for their own account, while the superuser may change the password for any account.

- passwd also changes the account or associated password validity period.
- **Important options** used along with passwd:
  - **-d, --delete** Delete a user's password (make it empty). This is a quick way to disable a password for an account. It will set the named account passwordless.
  - **-e, --expire** Immediately expire an account's password. This in effect can force a user to change his/her password at the user's next login.

userdel

- **userdel** command in Linux system is used to delete a user account and related files.
- This command basically modifies the system account files, deleting all the entries which refer to the username LOGIN.
- It is a low-level utility for removing the users.
- **Syntax: sudo userdel -f –r –Z neuser**
- **Important Options** used along with userdel:
  - **-f:** This option forces the removal of the specified user account. It doesn't matter that the user is still logged in. It also forces the *userdel* to remove the user's home directory and mail spool, even if another user is using the same home directory or even if the mail spool is not owned by the specified user.
  - **-r:** Whenever we are deleting a user using this option then the files in the user's home directory will be removed along with the home directory itself and the user's mail spool. All the files located in other file systems will have to be searched for and deleted manually.
  - **-Z :** This option remove any SELinux(Security-Enhanced Linux) user mapping for the user's login.

usermod

- usermod command or modify user is used to change the properties of a user.
- After creating a user we have to sometimes change their attributes like password or login directory etc.
- To change the home directory of a user
  - **usermod -d /home/newfolder existinguser**
- To change the group of a user
  - **usermod -g newgroupname existinguser**
- To change user login name
  - **usermod -l usernewname useroldname**
- To lock a user
  - **usermod -L existinguser** // -U for unlocking user
- To set an unencrypted password for the user
  - **usermod -p newpassword existinguser**

Groups: Groups in Linux refer to the user groups. In Linux, there can be many users of a single system, (normal user can take uid from 1000 to 60000, and one root user (uid 0) and 999 system users (uid 1 to 999)).

groupadd

- **groupadd** command is used to create a new user group.
- **Syntax**: **groupadd newgroupname**
- Every new group created is registered in the file "/etc/group". To verify that the group has been created, enter the command
  - **sudo tail /etc/group**
- Adding user while creating newuser
  - **useradd -d /home/newusername –p passwordstring -g existinggroupname newusername**

- Adding existing user to group // Explained once again in next topic
    - **usermod -g existinggroupname existinguser**

groupmod

- **groupmod** command is used to modify or change the existing group.
- It can be handled by superuser or root user.
- Basically, it modifies a group definition on the system by modifying the right entry in the database of the group.
- **Syntax: groupmod [option] groupname**
    - Change the group *name to new name*.
    - **groupmod -n groupnewname groupoldname**

gpasswd

- *gpasswd* command is used to administer the */etc/group* and */etc/gshadow*.
- *gpasswd* command **assigns** a user to a group with some security criteria.
- *gpasswd* command is called by a group administrator with a group name only which prompts for the new password of the group.
- System administrators can use the *-A* option to define group administrator(s) and *-M* option to define members.
- **Syntax: gpasswd [option] group**
- **Important Options:** Here only -A and -M options can be combined.
    - **-a, –add :** This option is used to add a user to the named group.
    - **-d, –delete :** It is used to remove a user from the named group.
    - **-r, –remove-password :** It is used to remove the password from the named group.
    - **-A, –administrators :** Set the list of administrators for the group.
    - **-M, –members : set the list of members of the group**.

Example:

- add the user to group:
    - **gpasswd -a existinguser existinggroup**
- Deleting the created user from group geeks.
    - **gpasswd -d existinguser existinggroup**

groupdel

- *groupdel* command is used to delete a existing group.
- It will delete all entry that refers to the group, modifies the system account files, and it is handled by superuser or root user.
- **Syntax: groupdel –f existinggroup**
- **Option used: -f –force**: It used to delete a group even if it is the primary group of a user

=== * ===

OpenLDAP Installation

- Lightweight Directory Access Protocol (LDAP) is a standard protocol designed to manage and access hierarchical directory information over a network.
- It can be used to store any kind of information, though it is most often used as a centralized authentication system or for corporate email and phone directories.

Installing and Configuring the LDAP Server

- Install the LDAP server and some associated utilities.
- $ sudo apt-get update
- $ sudo apt-get install slapd ldap-utils
- slapd - *the Stand-alone LDAP Daemon*
- During the installation,
    - Enter administrator password for LDAP.
    - Install and fill the fields asked during configuration:
- **sudo dpkg-reconfigure slapd**

There are quite a few new questions to answer in this process. Accept most of the defaults. Questions are:

- Omit OpenLDAP server configuration? **No**
- DNS domain name?
    - This option will determine the base structure of directory path. Read the message to understand exactly how this will be implemented. This installation use **gptmcse.com**
- Organization name?
    - For this guide, use **gptm** as the name of organization. (user may choose anything which is appropriate.)
- Administrator password? enter a secure password twice

- Database backend? **MDB**
- Remove the database when slapd is purged? **No**
- Move old database? **Yes**
- Allow LDAPv2 protocol? **No**

At this point, LDAP server is configured and running.

Open up the LDAP port on firewall so external clients can connect:

- **sudo ufw allow ldap**

Test LDAP connection with **ldapwhoami**, which should return the username as anonymous:

- **ldapwhoami -x**
  - **anonymous**
  - anonymous is the result expected, since running ldapwhoami without logging in to the LDAP server. This means the server is running and answering queries.
- If everything moves correctly it will display the administar's distinguished name (dn) otherwise it says invalid credentials.
- Portion of the output is shown below.

```
am2en@am2en-virtual-machine:~$ sudo apt install slapd ldap-utils
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ldap-utils is already the newest version (2.5.14+dfsg-0ubuntu0.22.04.2).
slapd is already the newest version (2.5.14+dfsg-0ubuntu0.22.04.2).
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi i965-va-driver intel-media-va-driver liba
  libblas3 libbluray2 libbs2b0 libchromaprint1 libcodec2-1.0 libdav1d5 libflashrom1 libfli
  liblilv-0-0 libllvm13 libmfx1 libmysofa1 libnorm1 libopenmpt0 libpgm-5.3-0 libpostproc55
  libsord-0-0 libsratom-0-0 libsrt1.4-gnutls libssh-gcrypt-4 libswresample3 libswscale5 lib
  libvidstab1.1 libx265-199 libxvidcore4 libzimg2 libzmq5 libzvbi-common libzvbi0 mesa-va-d
  vdpau-driver-all
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 74 not upgraded.
am2en@am2en-virtual-machine:~$ sudo dpkg-reconfigure slapd
  Backing up /etc/ldap/slapd.d in /var/backups/slapd-2.5.14+dfsg-0ubuntu0.22.04.2... done.
  Moving old database directory to /var/backups:
  - directory unknown... done.
  Creating initial configuration... done.
  Creating LDAP directory... done.
am2en@am2en-virtual-machine:~$ nano /etc/ldap/ldap.conf
am2en@am2en-virtual-machine:~$ ldapwhoami -x
anonymous
am2en@am2en-virtual-machine:~$ ldapwhoami -x -D cn=admin,dc=gptmcse,dc=com -W
Enter LDAP Password:
dn:cn=admin,dc=gptmcse,dc=com
am2en@am2en-virtual-machine:~$ ldapwhoami -x -D cn=admin,dc=example,dc=com -W
Enter LDAP Password:
ldap_bind: Invalid credentials (49)
```

=== * ===

11 – System Monitoring

System Monitoring:

df

- Displays the amount of disk space available on the file system containing each file name argument.
- If no file name is passed as an argument with df command then it shows the space available on all currently mounted file systems
- By default, **df** shows the disk space in 1 K blocks.

Syntax:

df [OPTION]...[FILE]...

- OPTION : to the options compatible with df command
- FILE : specific filename in case you want to know the disk space usage of a particular file system only.

```
am2en@am2en-virtual-machine:~$ df
Filesystem     1K-blocks     Used Available Use% Mounted on
tmpfs             398212     2032    396180   1% /run
/dev/sda3       19946096 12003592   6903964  64% /
tmpfs            1991060        0   1991060   0% /dev/shm
tmpfs               5120        4      5116   1% /run/lock
/dev/sda2         524252     6216    518036   2% /boot/efi
tmpfs             398212      104    398108   1% /run/user/1000
/dev/sr0          129448   129448         0 100% /media/am2en/CDROM
/dev/sr1         3569294  3569294         0 100% /media/am2en/Ubuntu 22.04 LTS amd64
am2en@am2en-virtual-machine:~$ df Killdemo.txt
Filesystem     1K-blocks     Used Available Use% Mounted on
/dev/sda3       19946096 12003592   6903964  64% /
am2en@am2en-virtual-machine:~$ df -m
Filesystem     1M-blocks  Used Available Use% Mounted on
tmpfs                389     2       387   1% /run
/dev/sda3          19479 11723      6743  64% /
tmpfs               1945     0      1945   0% /dev/shm
tmpfs                  5     1         5   1% /run/lock
/dev/sda2            512     7       506   2% /boot/efi
tmpfs                389     1       389   1% /run/user/1000
/dev/sr0             127   127         0 100% /media/am2en/CDROM
/dev/sr1            3486  3486         0 100% /media/am2en/Ubuntu 22.04 LTS amd64
am2en@am2en-virtual-machine:~$
```

=== * ===

cat /proc/cpuinfo

- The simplest way to determine what type of CPU you have is by displaying the contents of the /proc/cpuinfo virtual file.

The portion of the output is as shown below;

```
am2en@am2en-virtual-machine:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 140
model name      : 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz
stepping        : 1
microcode       : 0xffffffff
cpu MHz         : 2803.209
cache size      : 12288 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
```

cat /proc/meminfo

- The */proc/meminfo* file inside the */proc* pseudo-filesystem provides a usage report about memory on the system.
- When we want to find out statistics like used and available memory, swap space, or cache and buffers, we can analyze this file's contents.

The portion of the output is as shown below;

```
am2en@am2en-virtual-machine:~$ cat /proc/meminfo
MemTotal:        3982120 kB
MemFree:          213516 kB
MemAvailable:    1694728 kB
Buffers:           54120 kB
Cached:          1640176 kB
SwapCached:          320 kB
Active:           916056 kB
Inactive:        1328812 kB
Active(anon):       2912 kB
Inactive(anon):   597072 kB
Active(file):     913144 kB
Inactive(file):   731740 kB
```

=== * ===

System Log Files:

ls -l /var/log

- All Linux systems create and store information log files for boot processes, applications, and other events.
- These files can be a helpful resource for troubleshooting system issues.
- Most Linux log files are stored in a plain ASCII text file and are in the **/var/log** directory and subdirectory.
- Logs are generated by the Linux system daemon log, **syslogd** or **rsyslogd**.

The portion of the output is as shown below;

```
am2en@am2en-virtual-machine:~$ ls -l /var/log
total 6852
-rw-r--r--  1 root             root                0 Jun  6 20:18 alternatives.log
-rw-r--r--  1 root             root             3182 Jun  4 08:56 alternatives.log.1
-rw-r--r--  1 root             root             2940 Mar 19 23:15 alternatives.log.2.gz
drwxr-x---  2 root             adm              4096 Jun  8 21:11 apache2
-rw-r-----  1 root             adm                 0 Apr  1 16:16 apport.log
-rw-r-----  1 root             adm               106 Mar 19 23:24 apport.log.1
drwxr-xr-x  2 root             root             4096 Jun  4 09:00 apt
-rw-r-----  1 syslog           adm             74572 Jun  8 21:33 auth.log
-rw-r-----  1 syslog           adm             56471 Jun  3 23:52 auth.log.1
-rw-r-----  1 syslog           adm              1383 Jun  2 10:57 auth.log.2.gz
-rw-r-----  1 syslog           adm              1888 Apr 28 11:28 auth.log.3.gz
-rw-r-----  1 syslog           adm              1107 Apr  2 11:32 auth.log.4.gz
-rw-------  1 root             root              955 Jun  8 21:11 boot.log
-rw-------  1 root             root            12098 Jun  8 21:11 boot.log.1
-rw-------  1 root             root            23360 Jun  6 20:18 boot.log.2
```

=== * ===

System Maintenance:

shutdown

- The **shutdown** command in Linux is used to shutdown the system in a safe way.
- You can shutdown the machine immediately, or schedule a shutdown using 24 hour format.
- When the shutdown is initiated, all logged-in users and processes are notified that the system is going down, and no further logins are allowed.

- Only root user can execute shutdown command.

Syntax

shutdown [OPTIONS] [TIME] [MESSAGE]

- OPTIONS – Shutdown options such as halt, power-off (the default option) or reboot the system.
- TIME – The time argument specifies when to perform the shutdown process.
- MESSAGE – The message argument specifies a message which will be broadcast to all users

- The following example will schedule a system shutdown at 05 A.M:

  sudo shutdown 05:00

reboot

- **reboot** command is used to restart or reboot the system.
- In a Linux system administration, there comes a need to restart the server after the completion of some network and other major updates.
- Rebooting is needed so that the changes that the user has made can affect the server.

Syntax

reboot [OPTIONS] [TIME] [MESSAGE]

- Force immediate reboot in Linux system:

  sudo reboot -f

=== * ===

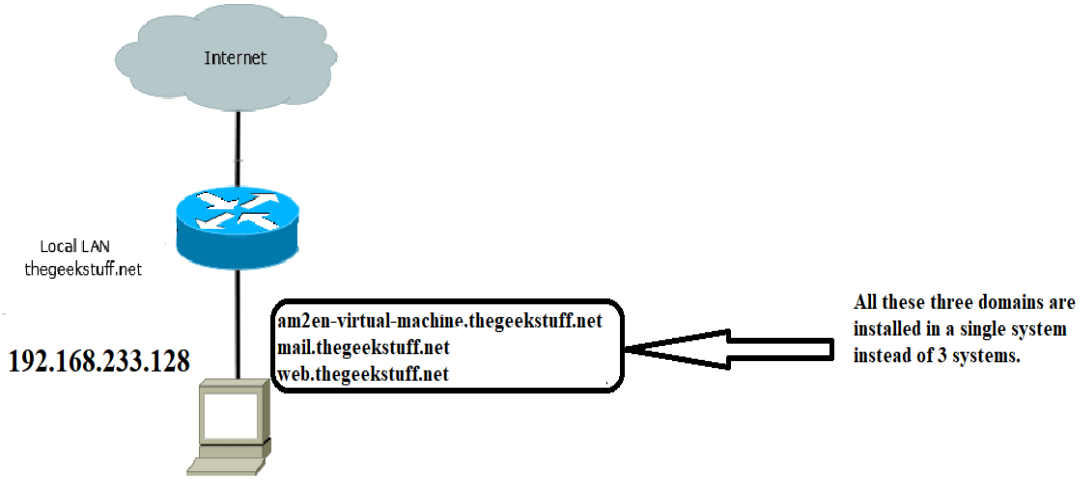
12 – Server Setup

DNS

- Domain Name Service (DNS) is an internet service that maps IP addresses to fully qualified domain names (FQDN) and vice versa.
- BIND stands for Berkley Internet Naming Daemon.
- BIND is the most common program used for maintaining a name server on Linux.

Steps to Install and Configure DNS

Prerequisites:



- We will use "thegeekstuff.net" domain as an example for this DNS installation. "mail", "web", "am2en-virtual-machine" are the hosts that resides within this domain as shown in the above diagram.
- Before proceeding further, know the ip address and hostname of your PC by using the below commands:

  $hostname

  $hostname –I

  In this case, hostname is am2en-virtual-machine and host ip address is 192.168.233.128

1. **Install Bind**

   $sudo apt-get install bind9

   1. All the DNS configurations are stored under /etc/bind directory.

2. **Configure the Cache Name Server**

- The job of a DNS caching server is to query other DNS servers and cache the response.
- Next time when the same query is given, it will provide the response from the cache. The cache will be updated periodically.
- Configure a Cache NameServer by adding your ISP (Internet Service Provider)'s DNS to the file **/etc/bind/named.conf.options**.

  $sudo nano /etc/bind/named.conf.options

  Add the below lines to the file;

```
forwarders {
        8.8.8.8;
        8.8.4.4;
};
```

- Restart the DNS Server

  $sudo service bind9 restart

3. **Test the Cache Name Server**

   $dig ubuntu.com

- The query time decreases if you dig for the second time.

```
am2en@am2en-virtual-machine:~$ sudo service bind9 restart
[sudo] password for am2en:
am2en@am2en-virtual-machine:~$ dig ubuntu.com

; <<>> DiG 9.18.12-0ubuntu0.22.04.1-Ubuntu <<>> ubuntu.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49294
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 3, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;ubuntu.com.                    IN      A

;; ANSWER SECTION:
ubuntu.com.             5       IN      A       185.125.190.20
ubuntu.com.             5       IN      A       185.125.190.29
ubuntu.com.             5       IN      A       185.125.190.21

;; AUTHORITY SECTION:
ubuntu.com.             5       IN      NS      ns2.canonical.com.
ubuntu.com.             5       IN      NS      ns3.canonical.com.
ubuntu.com.             5       IN      NS      ns1.canonical.com.

;; Query time: 32 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Fri Jun 09 20:27:21 IST 2023
;; MSG SIZE  rcvd: 151

am2en@am2en-virtual-machine:~$
```

4. Configure Primary/Master Nameserver

- Add forward and reverse resolutions to bind9 as show below;

  $sudo nano /etc/bind/named.conf.local

```
 GNU nano 6.2                                  /etc/bind/named.conf.local *
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "thegeekstuff.net" {
    type master;
    file "/etc/bind/db.thegeekstuff.net";
};
zone "233.168.192.in-addr.arpa" {
        type master;
        notify no;
        file "/etc/bind/db.10";
};
```

Note: Use first 3 octets of host ip address in reverse order.

5. Build the Forward Resolution for Primary/Master Nameserver

- Now, add the details which is necessary for forward resolution into /etc/bind/db.thegeekstuff.net.
- First, copy /etc/bind/db.local to /etc/bind/db.thegeekstuff.net

  $sudo cp /etc/bind/db.local /etc/bind/db.thegeekstuff.net
- After editing the file should look like;

```
 GNU nano 6.2                                                    /etc/bind/db.thegeekstuff.net
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     am2en-virtual-machine.thegeekstuff.net. root.localhost. (
                              3         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@                       IN      NS      am2en-virtual-machine.thegeekstuff.net.
thegeekstuff.net.       IN      MX      192                                     mail.thegeekstuff.net.
am2en-virtual-machine   IN      A       192.168.233.128
web                     IN      A       192.168.233.128
mail                    IN      A       192.168.233.128
```

6. Build the Reverse Resolution for Primary/Master Nameserver

- Add the details, which are necessary for reverse resolution to the file /etc/bind/db.10.
- Copy the file /etc/bind/db.127 to /etc/bind/db.10

  $sudo cp /etc/bind/db.127 /etc/bind/db.10
- After editing the file should look like;

Note: Everytime you edit /etc/bind/db.thegeekstuff.net and /etc/bind/db.10, update the serial number by 1.

```
 GNU nano 6.2                                                    /etc/bind/db.10
;
; BIND reverse data file for local loopback interface
;
$TTL    604800
@       IN      SOA     am2en-virtual-machine.thegeekstuff.net. root.localhost. (
                              4         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      am2en-virtual-machine.
128     IN      PTR     am2en-virtual-machine.thegeekstuff.net.
128     IN      PTR     mail.thegeekstuff.net.
128     IN      PTR     web.thegeekstuff.net.
```

- Finally, restart the bind9 service

  $sudo service bind9 restart

7. Test the DNS Server

- Now open the file /etc/resolv.conf and add the below line

  nameserver 192.168.233.128

- Test the server by pinging to mail.thegeekstuff.net

  $ping mail.thegeekstuff.net

```
am2en@am2en-virtual-machine:~$ ping mail.thegeekstuff.net
PING mail.thegeekstuff.net (192.168.233.128) 56(84) bytes of data.
64 bytes from am2en-virtual-machine.thegeekstuff.net (192.168.233.128): icmp_seq=1 ttl=64 time=0.041 ms
64 bytes from mail.thegeekstuff.net (192.168.233.128): icmp_seq=2 ttl=64 time=0.067 ms
64 bytes from web.thegeekstuff.net (192.168.233.128): icmp_seq=3 ttl=64 time=0.079 ms
64 bytes from am2en-virtual-machine.thegeekstuff.net (192.168.233.128): icmp_seq=4 ttl=64 time=0.069 ms
64 bytes from am2en-virtual-machine.thegeekstuff.net (192.168.233.128): icmp_seq=5 ttl=64 time=0.061 ms
64 bytes from web.thegeekstuff.net (192.168.233.128): icmp_seq=6 ttl=64 time=0.062 ms
```

- If everything is configured correctly then ping command will result in correct reply from the DNS server with proper name resolution as show above.
- It shows the ip address of the machine where DNS is installed.

=== * ===

FTP

- **FTP (file transfer protocol)** is an internet protocol that is used for transferring files between client and server over the internet or a computer network.
- **FTP server** enables the functionality of transferring files between server and client.
- A **client** connects to the **server** with credentials and depending upon the permissions it has, it can either read files or upload files to the server as well.
- A client makes two types of connection with the server, one for giving commands and one for transferring data.
- The client issues the command to the FTP server on port 21, which is the command port for FTP. For transferring data, a data port is used.

There are two types of connection modes for transferring data:

- **Active mode:** In Active mode, the client opens a port and waits for the server to connect to it to transfer data. The server uses its port 20 to connect to the client for data transfer.
- **Passive mode:** In this, when a client requests a file from the server, the server opens a random port and tells the client to connect to that port.

Steps to Install and transfer files via FTP

1. **Install FTP Server**

   $sudo apt install vsftpd

2. Once successfully installed, check the status of ftp by executing the below command;

   $sudo systemctl status vsftpd

3. If FTP Server is running, then it will be highlighted with green dot and shows active (running)

```
am2en@am2en-virtual-machine:~$ sudo systemctl status vsftpd
[sudo] password for am2en:
● vsftpd.service - vsftpd FTP server
     Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-06-09 19:41:56 IST; 3h 4min ago
    Process: 991 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/empty (code=exited, status=0/SUCCESS)
   Main PID: 998 (vsftpd)
      Tasks: 1 (limit: 4573)
     Memory: 2.2M
        CPU: 65ms
     CGroup: /system.slice/vsftpd.service
             └─998 /usr/sbin/vsftpd /etc/vsftpd.conf

Jun 09 19:41:56 am2en-virtual-machine systemd[1]: Starting vsftpd FTP server...
Jun 09 19:41:56 am2en-virtual-machine systemd[1]: Started vsftpd FTP server.
am2en@am2en-virtual-machine:~$
```

4. If FTP server is not running, then execute the below command and check once again the status as mentioned above.

   $sudo systemctl enable --now vsftpd

2. Configure Firewall

$sudo ufw allow 20/tcp

$sudo ufw allow 21/tcp

$sudo ufw allow 990/tcp

For passive FTP Connection, use ports between 5000 to 10000

$sudo ufw allow 5000:10000/tcp

3. Configure Users

5. Let us create a public user account for FTP

   $sudo adduser ftpuser

6. Create a directory for the above user and set the permission to access this directory from my user account.

   $sudo mkdir /ftp

   $whoami

Note: In this case, the user is am2en. You need to use your username

$sudo chown am2en /ftp

4. Configure and Secure vsftpd

7. Open the vsftpd configuration file and do the following changes;

   $sudo nano /etc/vsftpd.conf

8. Uncomment the below lines;

   anonymous_enable = NO

   local_enable = YES

   write_enable = YES

   chroot_local_user = YES

   chroot_list_enable = YES

   chroot_list_file = /etc/vsftpd.chroot_list

9. Add the below lines;

   pasv_min_port = 5000

   pasv_max_port = 10000

   local_root = /ftp

   allow_writeable_chroot = YES

   local_umask = 0002

Finally press ctrl + x then y and then enter to save the file.

10. Now, create the chroot_list file and add the below usernames line by line who can use FTP

    $sudo touch /etc/vsftpd.chroot_list

    $sudo nano /etc/vsftpd.chroot_list

root

am2en

11. Save the file and restart the vsftpd server

    $sudo systemctl restart --now vsftpd

5. Connecting to our FTP Server

12. Visit the website https://filezilla-project.org and download Filezilla client software.
13. Open the FTP client application present in the bin directory by untaring the downloaded package.
14. Enter the below details in the application window and click on Quickconnect;

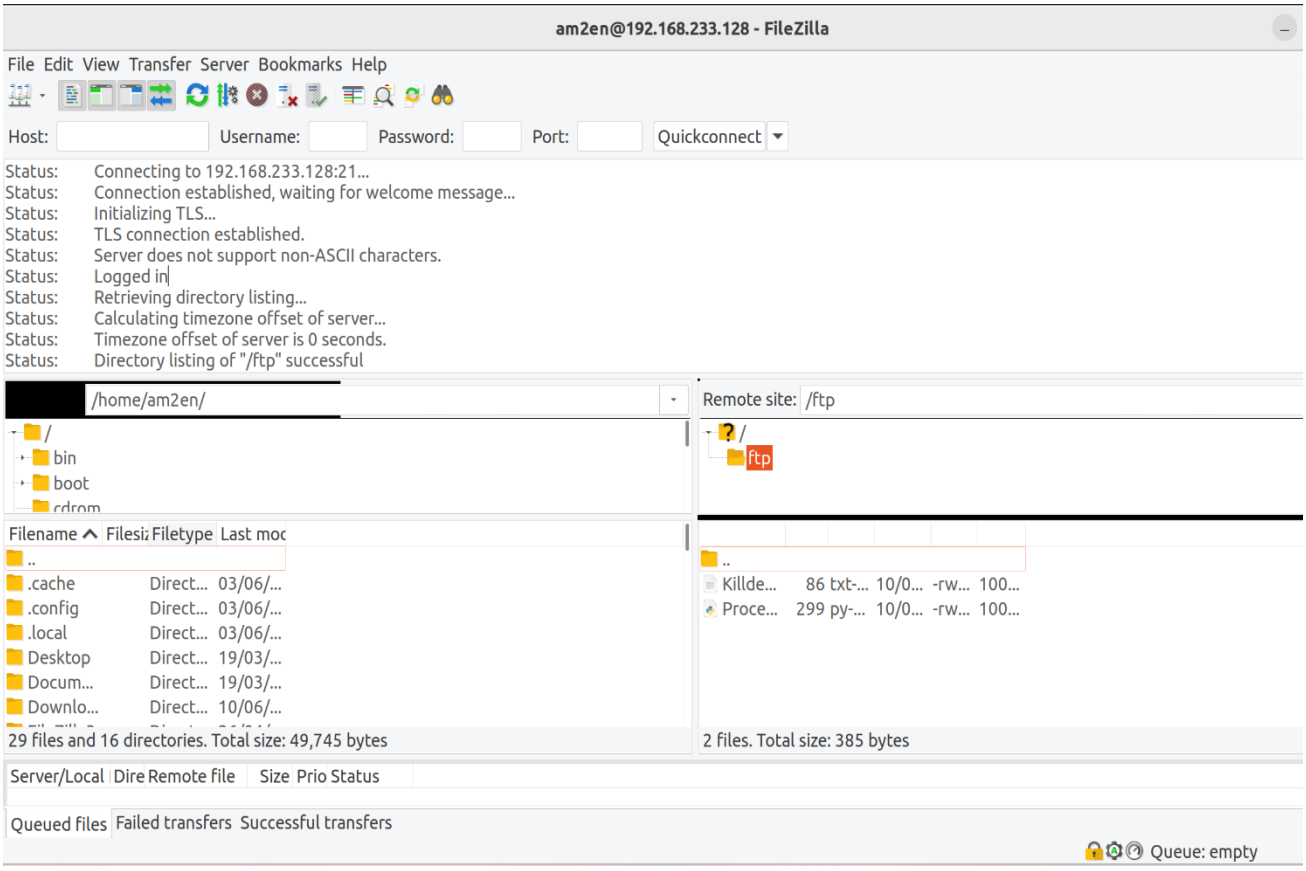
    Host: 192.168.233.128 // Here use your machine ip address

    Username: am2en // Here use the logged in user name to which ftp access is given

    Password: ******** // User Login password

    Port: 21
15. If everything moves correctly, then it will login and displays the ftp server home directory '/ftp'

Now, try to upload and download the files from client and server respectively and respective status of operation will be displayed in the application.



=== * ===

Apache Web Server

- Apache is free and open-source software of web server that is used by approx 40% of websites all over the world.
- Apache HTTP Server is its official name and is developed and maintained by the Apache Software Foundation.
- Apache permits the owners of the websites for serving content over the web. It is the reason why it is known as a "web server."

Steps to Install Apache Webserver

1. **Installing Apache**

   $sudo apt update

   $sudo apt install apache2

2. **Adjusting the Firewall**

   $sudo ufw app list

   16.It displays the list of apps allowed from Firewall

   $sudo ufw allow 'Apache'

3. **Checking Your Webserver**

- Check the status of apache webserver by executing the below command;

  $sudo systemctl status apache2

```
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-06-10 18:36:45 IST; 1h 25min ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 961 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 997 (apache2)
      Tasks: 55 (limit: 4573)
     Memory: 7.5M
        CPU: 584ms
     CGroup: /system.slice/apache2.service
             ├─ 997 /usr/sbin/apache2 -k start
             ├─1000 /usr/sbin/apache2 -k start
             └─1001 /usr/sbin/apache2 -k start

Jun 10 18:36:44 am2en-virtual-machine systemd[1]: Starting The Apache HTTP Server...
Jun 10 18:36:45 am2en-virtual-machine apachectl[976]: AH00558: apache2: Could not reliably
Jun 10 18:36:45 am2en-virtual-machine systemd[1]: Started The Apache HTTP Server.
~
```

- If the Active status is not running as shown above, then you need to start/restart the service by executing the below commands;

  $sudo systemctl start apache2

  $sudo systemctl restart apache2

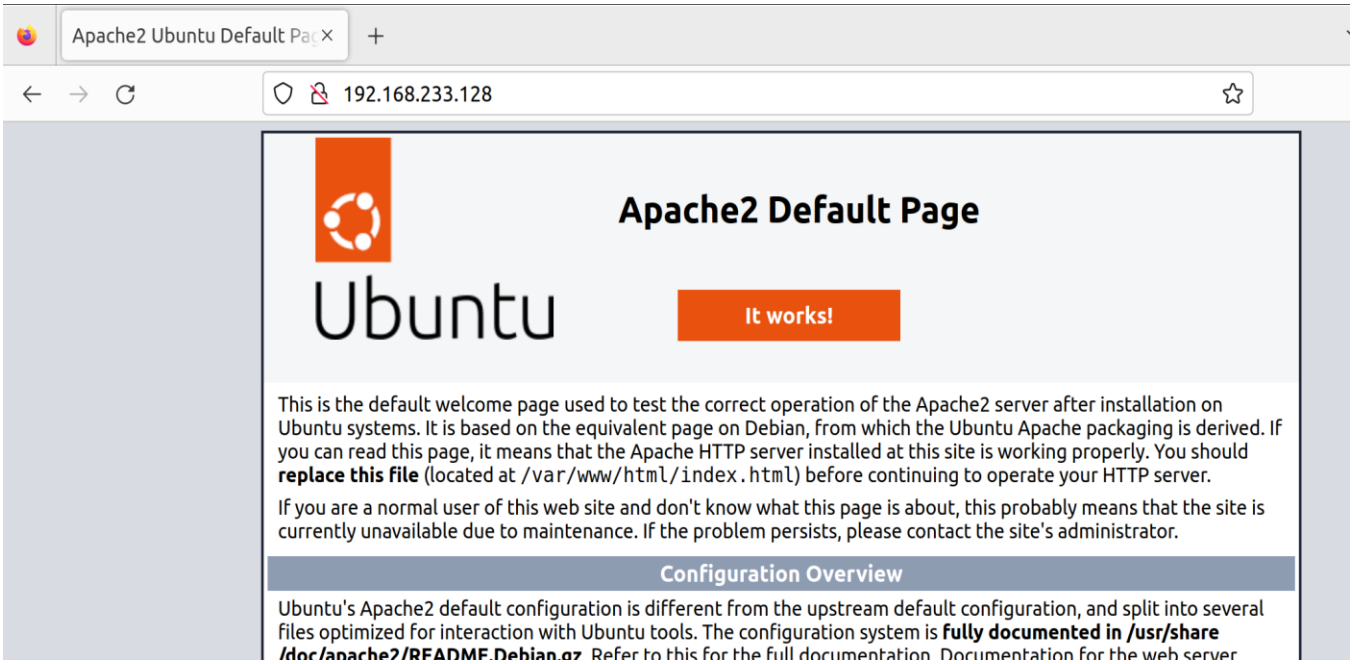
- Know the ip address of the system by using any of the below commands;

  $ifconfig

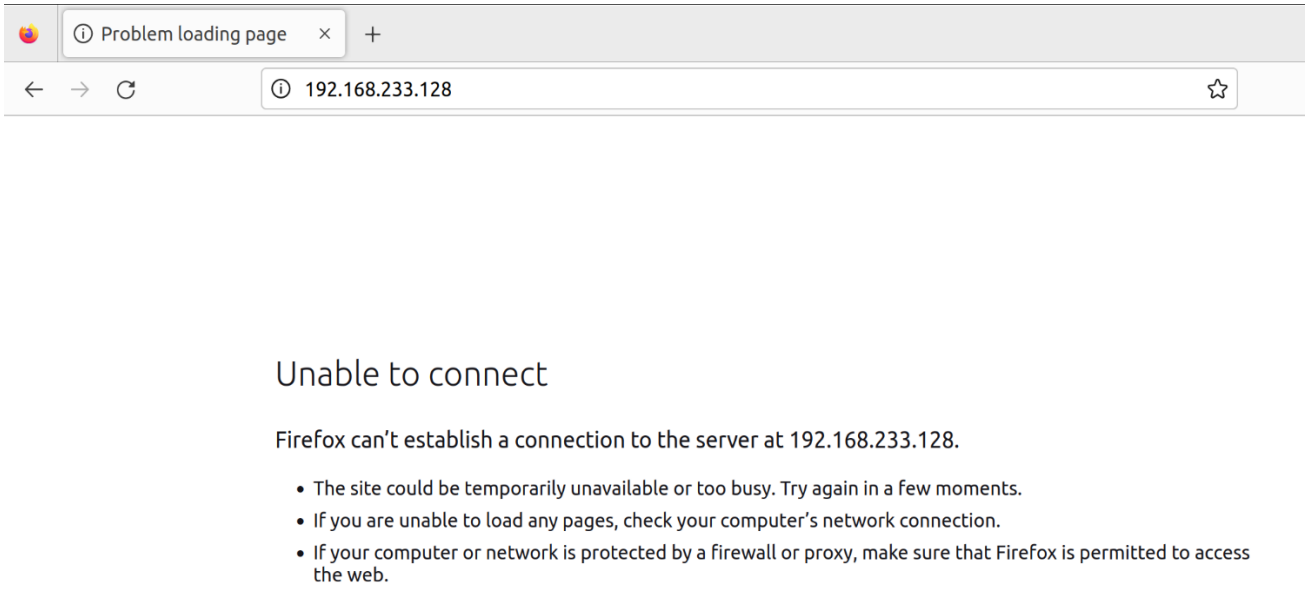
$hostname –I

In this case, it is 192.168.233.128

- Now, open the browser say Mozilla Firefox web browser and in search bar type http://your_server_ip_address i.e., http://192.168.233.128 and below output will be displayed.



- If there is any problem in connecting to the server, problem loading page or unable to connect will be shown as below;



- Finally, stop the apache server by executing the below command and also check the status once as shown below;

  $sudo systemctl stop apache2

  $sudo systemctl status apache2

```
am2en@am2en-virtual-machine:~$ sudo systemctl stop apache2
am2en@am2en-virtual-machine:~$ sudo systemctl status apache2
○ apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: inactive (dead) since Sat 2023-06-10 20:13:59 IST; 4s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 961 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
    Process: 4277 ExecStop=/usr/sbin/apachectl graceful-stop (code=exited, status=0/SUCCESS)
   Main PID: 997 (code=exited, status=0/SUCCESS)
        CPU: 736ms

Jun 10 18:36:44 am2en-virtual-machine systemd[1]: Starting The Apache HTTP Server...
Jun 10 18:36:45 am2en-virtual-machine apachectl[976]: AH00558: apache2: Could not reliably determine
Jun 10 18:36:45 am2en-virtual-machine systemd[1]: Started The Apache HTTP Server.
Jun 10 20:13:58 am2en-virtual-machine systemd[1]: Stopping The Apache HTTP Server...
Jun 10 20:13:58 am2en-virtual-machine apachectl[4279]: AH00558: apache2: Could not reliably determine
Jun 10 20:13:59 am2en-virtual-machine systemd[1]: apache2.service: Deactivated successfully.
Jun 10 20:13:59 am2en-virtual-machine systemd[1]: Stopped The Apache HTTP Server.
lines 1-16/16 (END)
```

=== * ===

REFERENCE:

- The content in this manual/journal is taken from various websites and modified to suit the students' requirements as per C-20 curriculum.

Users Note:

- This manual is prepared by MBA KHAN, Lecturer C.S Dept. Government Polytechnic Mudhol – 587313.
- This manual is prepared by referring online resources and other manuals.
- The experiments included in this manual are tested under Virtual Machine [VM Ware V 17, Ubuntu 22.04.2 LTS] installed on top of Windows 11.
- If any improvements or corrections you can feel free to give your suggestions via email : ameen.gpt2013@gmail.com
- You are free to copy, modify and circulate as per your needs by deleting the primary author of this document.

=== * ===

=== * ===

=== * ===