1

INTRODUCTION

Unit Structure :

1.0 OBJECTIVES

After going through the unit, the learner will be able to:

- Demonstrate proficiency with the Linux command line.
- Install Linux server.
- Manage software using RPM and YUM.
- To understand and practice the tools commonly found on most Linux distributions.

1.1 INTRODUCTION

Linux is a family of operating systems that are similar to Unix. Linux is Free and Open-Source Software (FOSS). It began as a hacker's toy many ages ago, but now has matured into a well-known platform for its reliability, performance, and extensibility. The Linux kernel, as well as other system software and libraries, are included in distributions, many of which are given by the GNU Project.

Most hardware platforms now support the Linux operating system. Linux runs on practically all architectures, including i386 and SPARC. Linux may now be found on practically every device, including watches, televisions, mobile phones, servers, desktops, and vending machines.

Linux has made significant gains in the consumer desktop, laptop, tablet, and smartphone industries. This chapter briefly reviews several key underlying concepts of Linux Distributions and different ways of the installation process.

1.2 TECHNICAL SUMMARY OF LINUX DISTRIBUTIONS

As Linux is free, anyone can take the Linux kernel and other supporting programs, repackage them, and resell them. Different parts of Linux are developed by different organizations. These different parts include kernel, shell utilities, X server, system environment, graphical programs, etc. If you wish, you can access the codes of all these parts and assemble them yourself.

Packages are used to group together software and applications, and these package kinds are used to categorise Linux distributions (also called as Distros). Debian (deb), RedHat Package Manager (RPM), Fedora, and Ubuntu are all popular Linux distributions. Red Hat Enterprise Linux and SUSE Linux Enterprise Server are two commercial distributions. A windowing system like X11 or Wayland, as well as a desktop environment like GNOME or KDE Plasma, are included in desktop Linux distributions. Server distributions may or may not include graphics or may include a solution stack such as LAMP.

There are several Linux distributors providing different features. Let's discuss some of the popular Linux distros:

1) Ubuntu

Ubuntu based on Debian and came into existence in 2004. It quickly became popular. It is the most well-known Linux distribution. Ubuntu is a next version of Debian and easy to use for newbies. It comes with a lots of pre-installed apps and easy to use repositories libraries. For package management, Ubuntu uses apt and its graphical fronted Ubuntu Software Center. Ubuntu used GNOME2 desktop environment earlier but now it has developed its own unity desktop environment. It releases every six months and currently working to expand to run on tablets and smartphones.

2) Linux Mint

Mint is based on Ubuntu and uses its repository software, so some packages are common in both. Earlier it was used as an alternative of Ubuntu because media codecs and proprietary software are included in mint which was absent in Ubuntu. But now it has its own popularity and it uses cinnamon and mate desktop instead of Ubuntu's unity desktop environment.

3) Debian

Debian is secure Linux based operating system. has its existence since 1993 by Lan Murdock and releases its versions much slowly then Ubuntu and Mint. This makes it one of the most stable Linux distributors. Debian has developed and maintained GNU/Linux operating system for well. Ubuntu was founded to improve the core bits of Debian more quickly and make it more user friendly.

4) Red Hat Enterprise / CentOS

Red hat is a commercial Linux distributor. Their products red hat enterprise Linux (RHEL) and Fedora are freely available. RHEL is an enterprise Linux operating system which is certified on many clouds and vendors.

Red hat uses trademark law to prevent their software from being redistributed. CentOS is a free version of RHEL and provide a stable platform for a long time. It is a community project that uses red hat enterprise Linux code but removes all its trademark and make it freely available.

5) Fedora

Fedora is a community-supported project that mainly focuses on free software and provides latest version of software. It doesn't make its own desktop environment; it is the upstream source of RHEL and CentOS. By default, it has GNOME3 desktop environment. Different editions of Fedora like Workstation, CoreOS, Silverblue are available. But Fedora is less stable.

6) OpenSuse

OpenSuse started out a German translation of Slackware Linux sponsored by SUSE Linux and other companies, but eventually grew into its own distribution. OpenSuse is known for the KDE desktop and stability. OpenSuse uses zypper and its graphical fronted, the Yast software center for package management.

7) Mageia

Mageia Linux is a new Linux distribution started in 2010 and is based on Mandriva Linux. It is a GNU/Linux-based, Free Software operating system. Mageia is easy to install and easy to use. Mageia utilizes urpmi and drakrpm for package management. There are eight major releases for this stable operating system.

8) Arch Linux

Arch Linux has its own package manager, pacman, and uses pkg.tar.xz packages. Arch doesn't come with a graphical installer, so you'll have to perform everything from the command line. For new Linux users, this can be scary. Arch's core philosophy is KISS (keep it simple, stupid). Some popular beginner-friendly distributions, such as Manjaro Linux, have cloned Arch.

9) Slackware Linux

Slackware is the oldest Linux distribution founded in 1992 by Patrick Volkerding. Slackware does not have a package manager and all the software is compiled by the system administrator or normal users of the system. Packages in Slackware are nothing more than source code. Use Slackware if you truly want to learn a lot about how Linux works.

10) Gentoo Linux

The Gentoo package management system is based on portage. Gentoo can be difficult to install, and it can take many days to complete the procedure. The benefit of this approach is that the software is custom-made for the hardware it will be running on. Portage, like Slackware, leverages application source code. Try Sabayon if you like the idea of Gentoo but want something more beginner-friendly.

1.3 INSTALLING LINUX IN A SERVER CONFIGURATION

Linux has many distributions such as Ubuntu, Fedora, Redhat, Debian but all run on top of Linux server itself. Installation of every distribution is similar. We can install these distributions using following methods:

1. Using CD-ROM or USB Stick
2. Using Virtual Box VMWARE

1. Installation using CD-ROM or USB Stick:

For this, we can download .iso or ISO files from the internet and burn them to a CD-ROM or USB stick after making them bootable with Pen Drive Linux and UNetBootin. Following steps are to be followed:

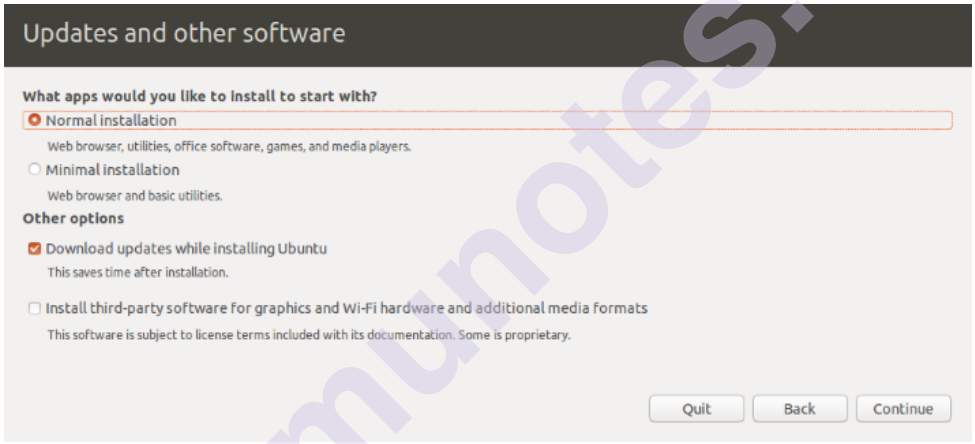
1. **Boot into the USB Stick**

After inserting a CD-ROM or pen drive into your computer, you must restart it. When the computer boots up, press enter to select the CD-ROM or pen drive option to continue the boot process. Hold the F12 key to start the boot process and try a manual boot configuration. Before launching the system, you'll be able to choose from a variety of boot settings. You will be given a selection of possibilities, whether it is USB or CD ROM, or a number of operating systems, from which you must choose one.

When your computer powers up, you'll see a new screen called "GNU GRUB," which is a boot loader for Linux installation. This screen will only appear if there are multiple operating systems installed.

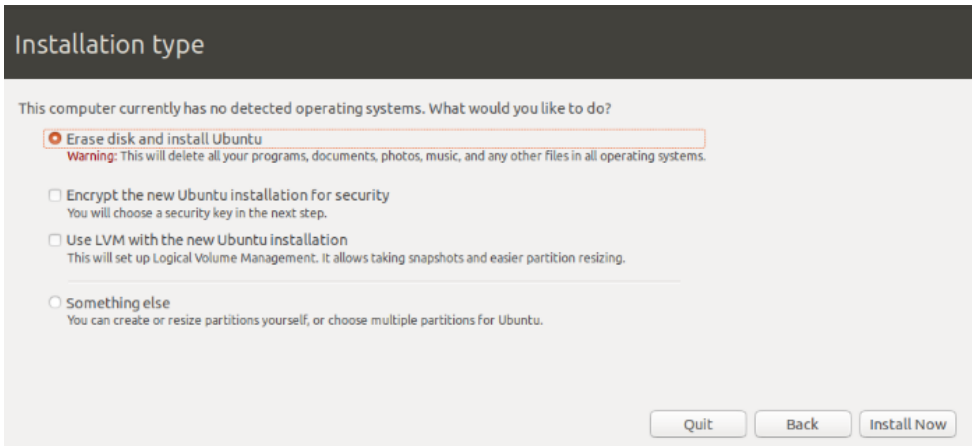
Set the layout of the keyboard.

You will now be questioned. To get started with Linux, what apps would you like to install? 'Normal installation' and 'Minimal installation' are the two alternatives.



2. Derive Selection

Select the drive where the OS will be installed. If you want to replace your current OS, choose "Erase Disk and Install Ubuntu," otherwise choose "Something else" and click INSTALL NOW.

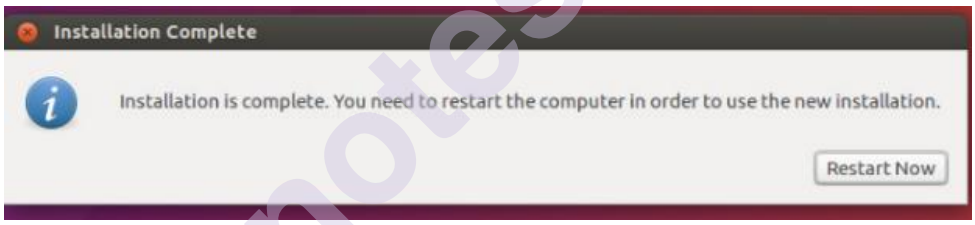


3. Start Installation

A small screen will appear, requesting confirmation. If you don't want to modify any of the information, click Continue. Install Linux in your chosen area on the map. Give the login information.

4. Complete the installation process

When the installation is finished, you'll be prompted to restart your computer.



you can also download drivers of your choice through the System Settings menu.

2. Installation using Virtual Box VMware:

We can install Linux operating system in a virtual machine (VM). A virtual machine is a software environment that mimics the conditions of a physical environment, such as a computer. The environment is based on your physical PC's hardware and is only limited by the components within. On a dual-core processor, for example, you couldn't have a virtual four-core CPU. Virtualization can be accomplished on a variety of platforms, but the results will be significantly superior on PCs with a CPU that supports it. Several virtual machine tools make it simple to set up Linux operating systems. VMware creates the most advanced virtual machine software.

To install Linux using VMware, follow these steps:

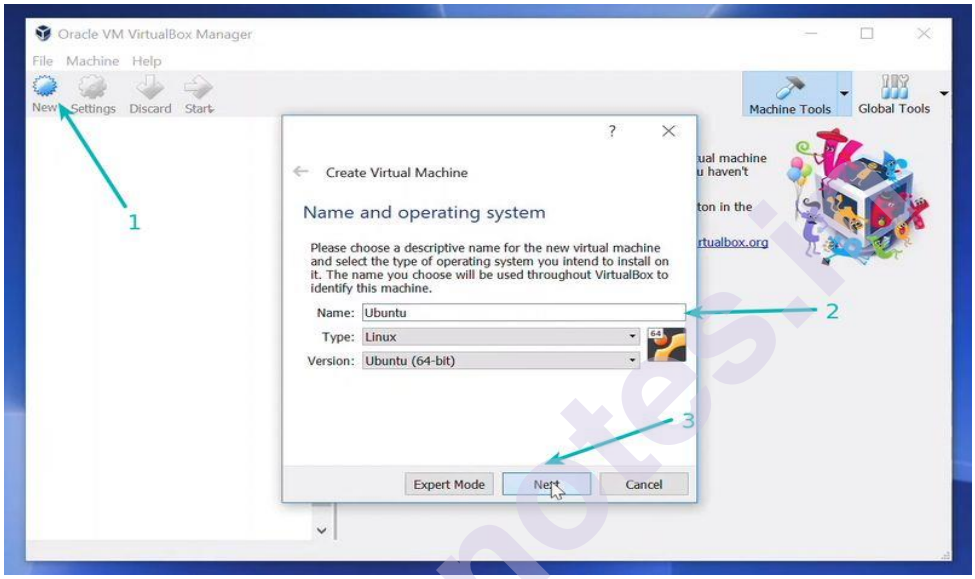
1) Download the free VMware Workstation Player: To start with this, download latest VMware Workstation Player from VMware website. Once the Player is downloaded, launch the installer and follow the installation wizard. All 32-bit and 64-bit Linux distros work in a virtual machine.

2) Install, and restart Windows.

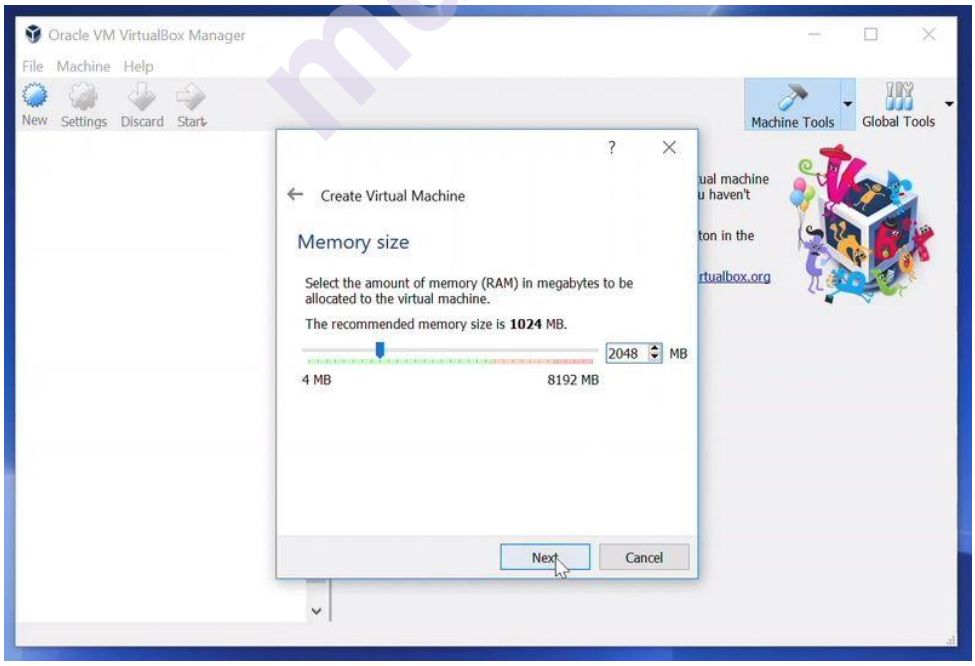
3) Create and configure your virtual machine: Next, you need to download the ISO file of the Linux distribution. You can get this image from the official website of the Linux distribution you are trying to use.

4) Install Linux in the virtual machine: You have installed VirtualBox and you have downloaded the ISO for Linux. You are now set to install Linux in VirtualBox.

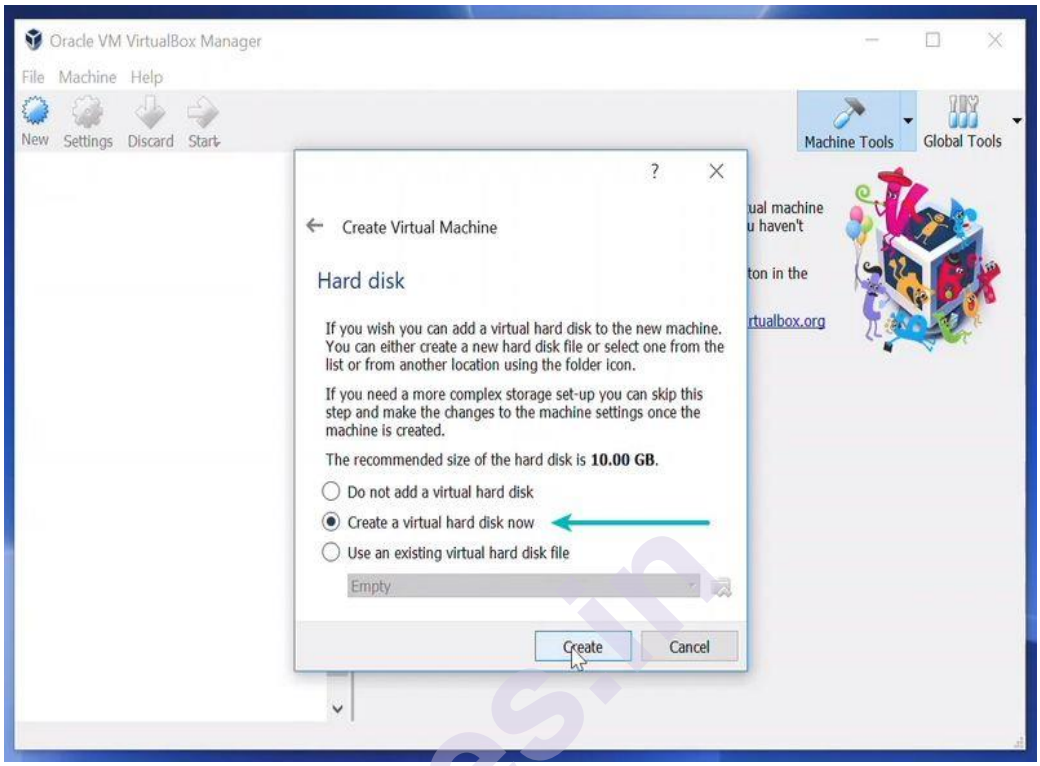
Start VirtualBox, and click on the New symbol. Give the virtual OS a relevant name.



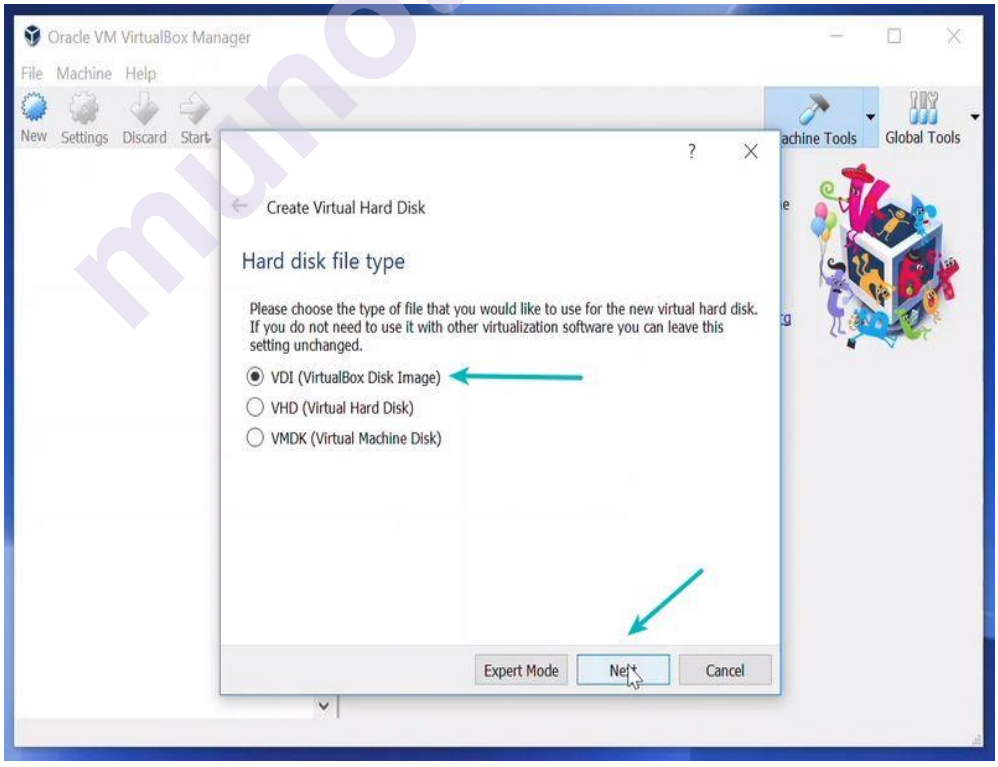
Allocate RAM to the virtual OS.



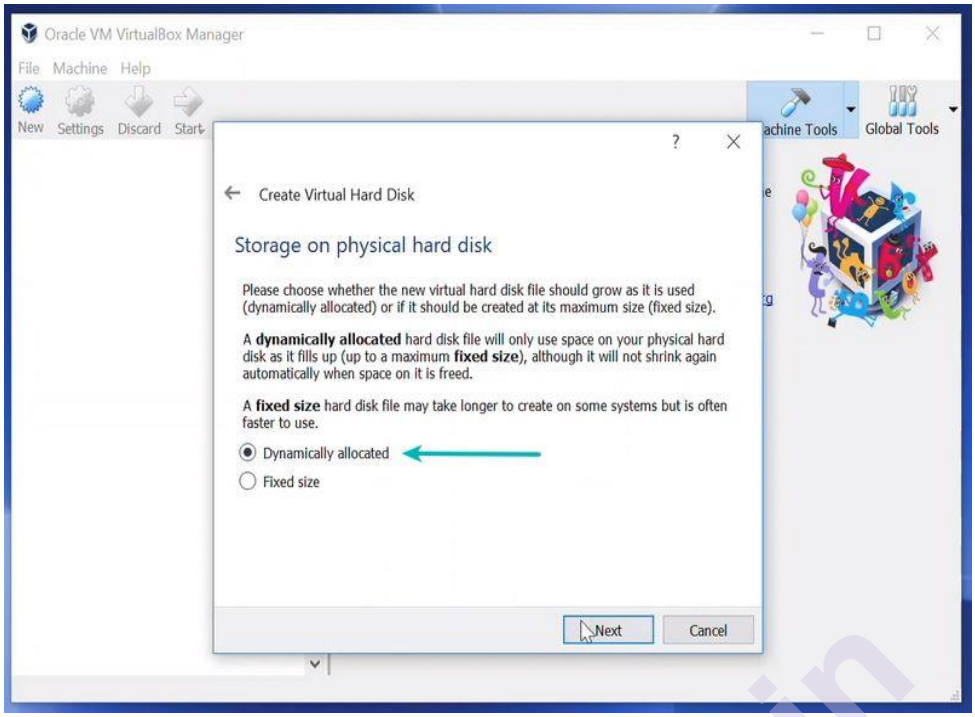
Create a virtual disk. This serves as the hard disk of the virtual Linux system.



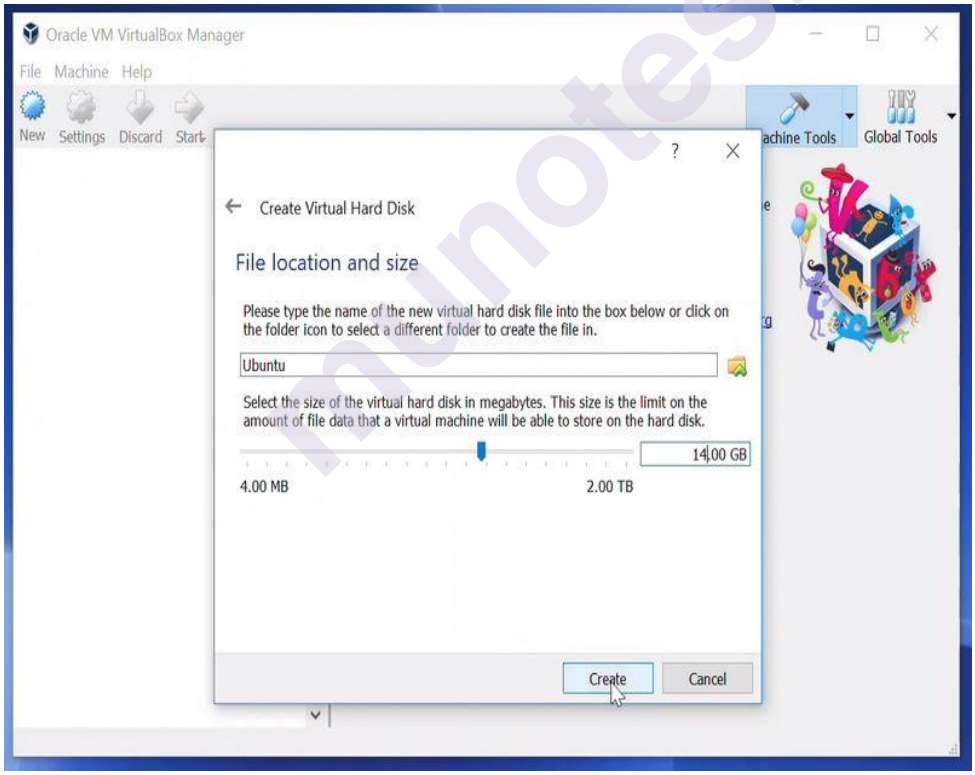
Use the VDI file type.



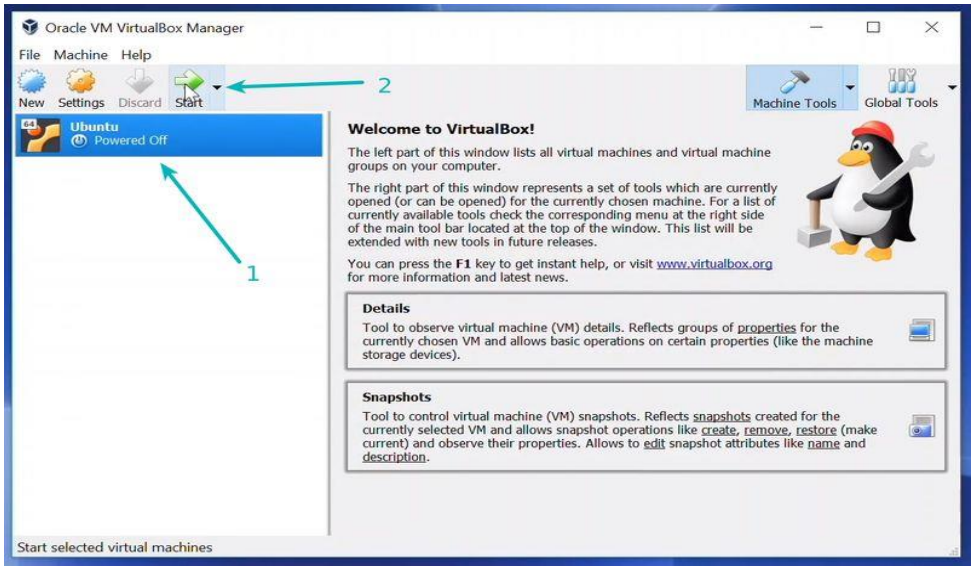
You can choose either the “Dynamically allocated” or the “Fixed size” option for creating the virtual hard disk.



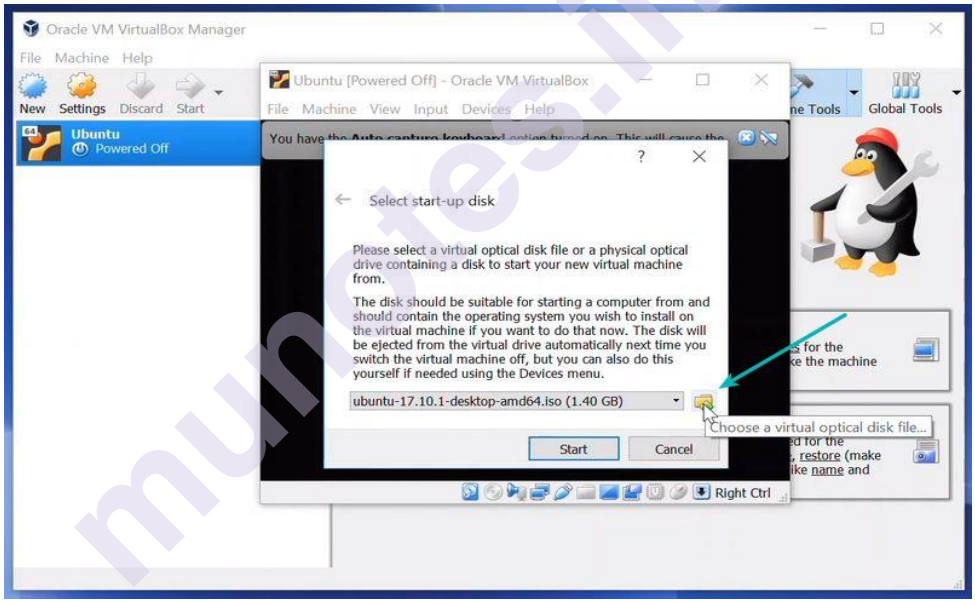
The recommended size is 10 GB. 15-20 GB is preferable.

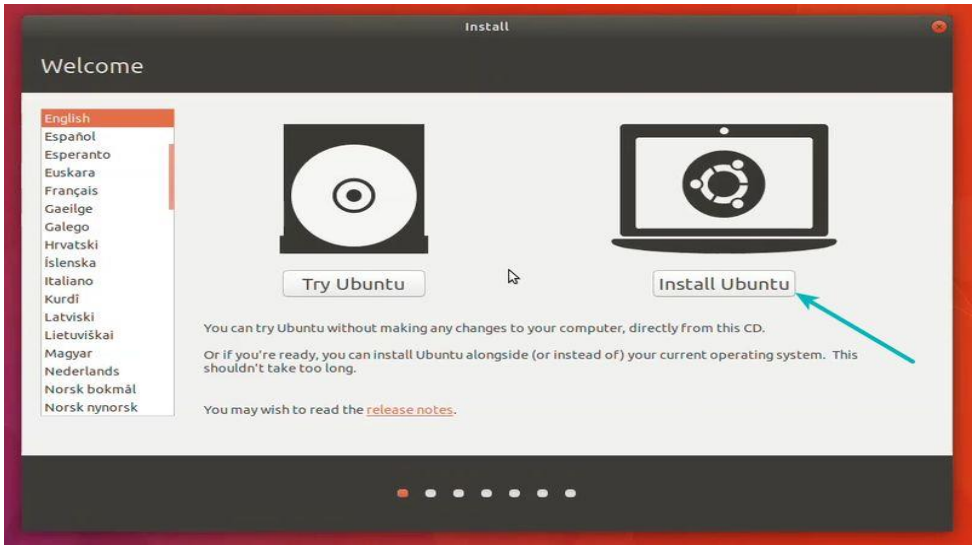


It’s time to boot that ISO and install Linux as a virtual operating system.



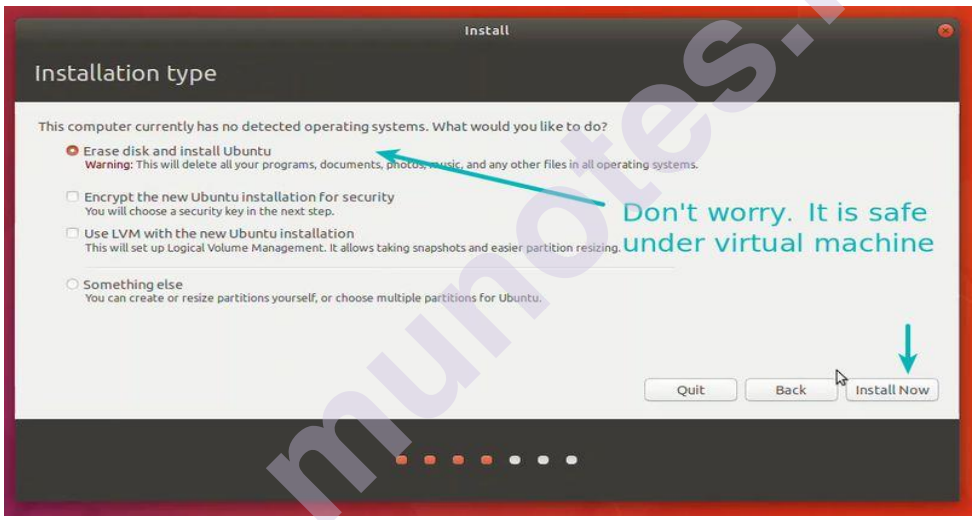
If VirtualBox doesn't detect the Linux ISO, browse to its location by clicking the folder icon as shown in the picture below:



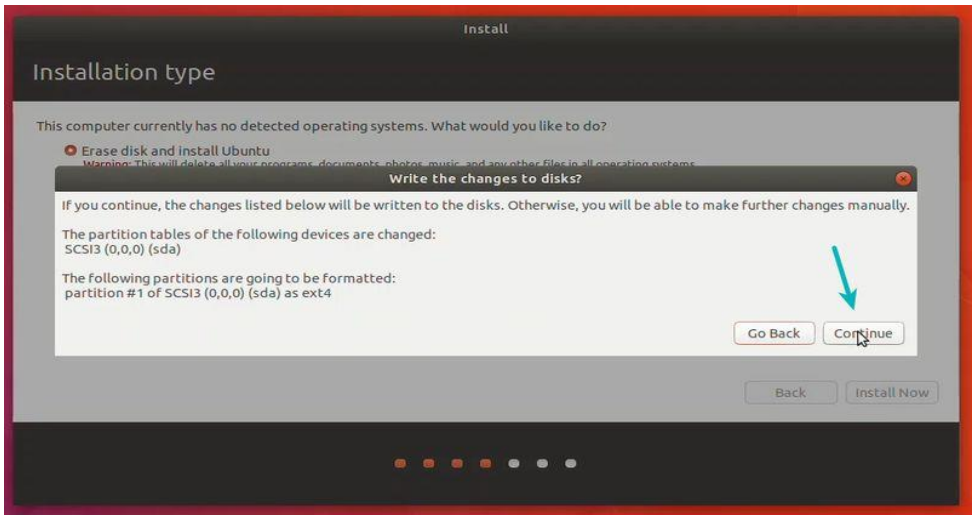


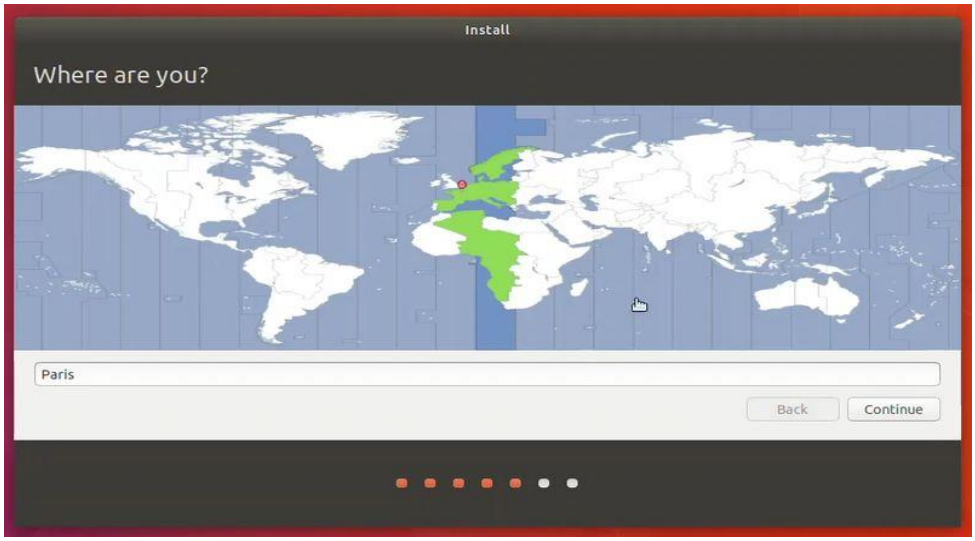
Skip to Continue.

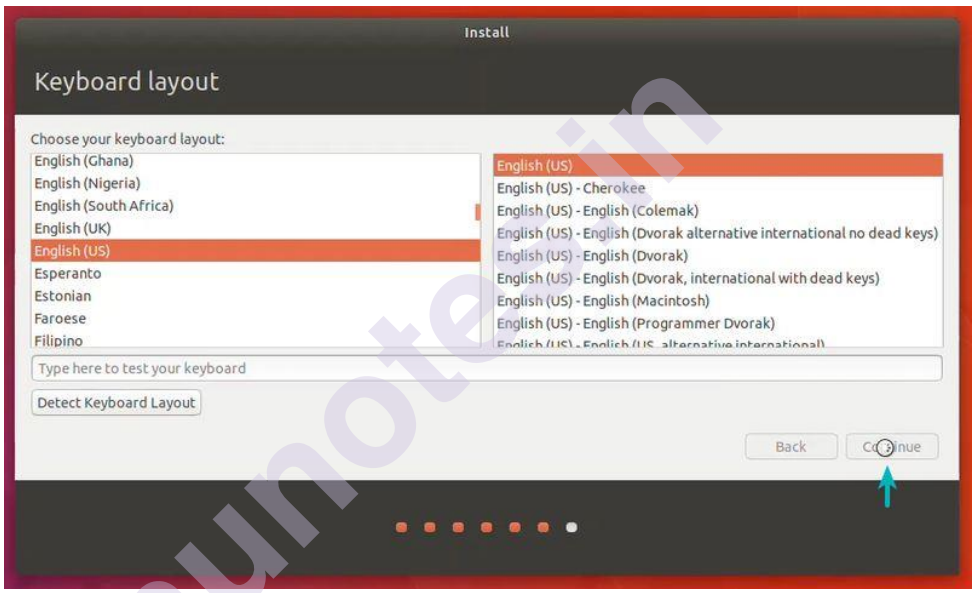
Select 'Erase disk and install Ubuntu'. It won't delete anything on your Windows operating system. You are using the virtual disk space of 15-20GB that we created in previous steps. It won't impact the real operating system.



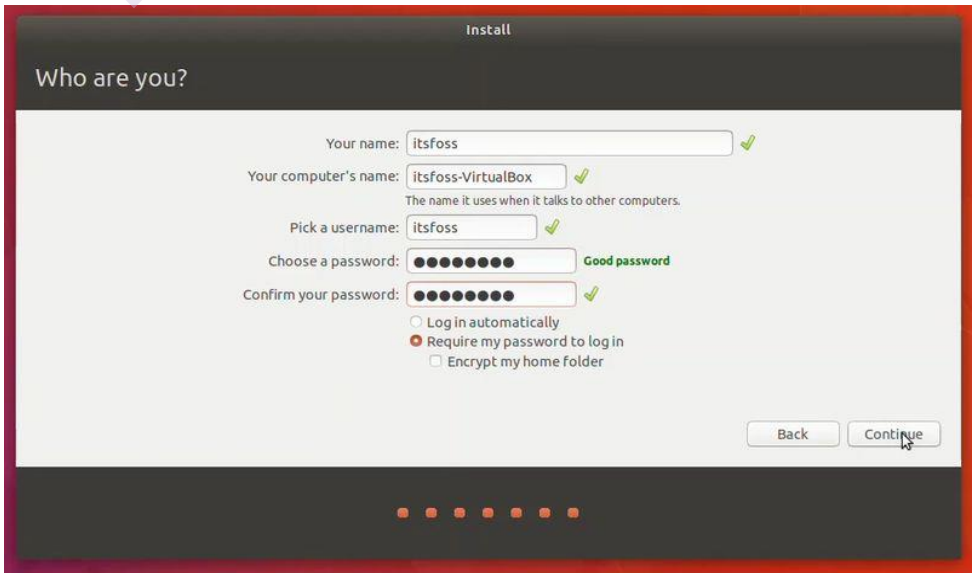
Click on Continue.





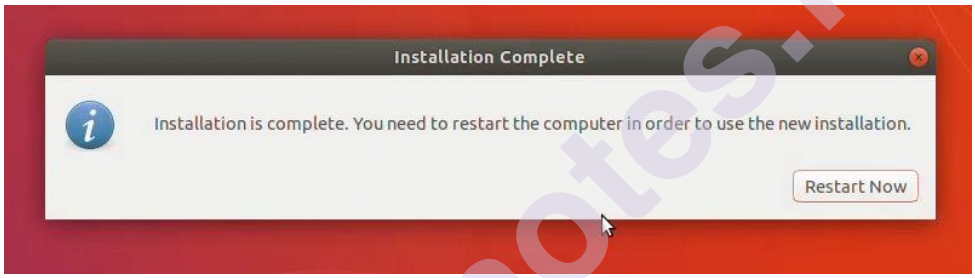


Try to choose a password that you can remember. Password can be reset if needed.



It may take 10-15 minutes to complete the installation after this screen.

Once the installation finishes, restart the virtual system.



If it gets stuck on the screen below, you may close the VirtualBox.

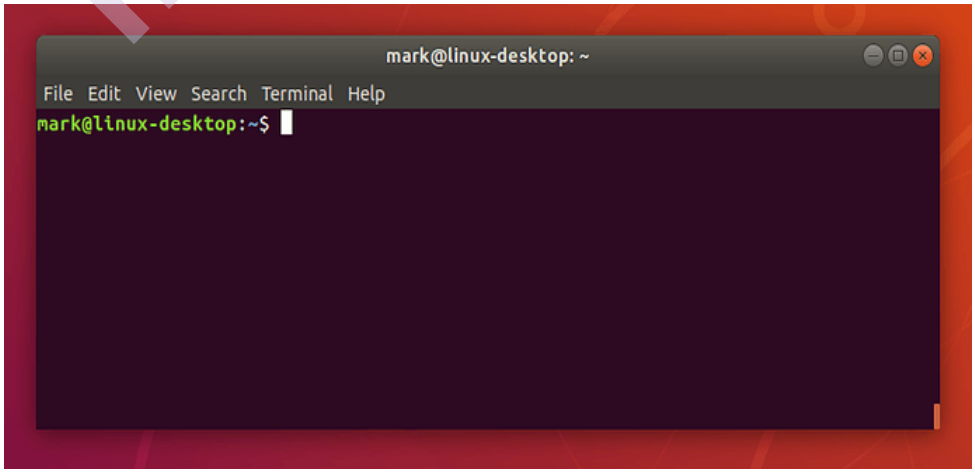
Linux is installed on your system.

1.4 THE COMMAND LINE

The Linux command line is a text-based computer interface. It can be referred to as the shell, terminal, console, prompt, or a variety of other names, and it can appear to be difficult to use. However, because of the ability to copy and paste commands from a website, as well as the command line's power and versatility, it may be necessary to use it while trying to follow instructions online.

Generally on an Ubuntu system, type the first few letters of "terminal," "command," "prompt," or "shell" into the Activities box at the top left of the screen to locate a terminal launcher. The launcher has been set up with all of the most prevalent synonyms, so you should have no trouble finding it.

A terminal launcher is normally found in the same location as your other application launchers in other versions of Linux or Ubuntu. Most Linux systems use the same default keyboard shortcut to start the terminal: Ctrl-Alt-T. If you can't locate a launcher, or if you just want a faster way to bring up the terminal, use Ctrl-Alt-T. The colours may differ depending on your Linux system, and the text may say something different, but the general arrangement of a window with a wide text area should be comparable.



You can run various commands here. But, you have logged in as a Guest (Normal user) or Super user makes difference here. The superuser (use command sudo) is a user with superpowers, as the name implies. In previous

systems, you could enter in as a legitimate user with a real username (usually invariably "root") as if you knew the password. Root has the ability to alter or delete any file in any directory on the system, regardless of who owns it; root can rewrite firewall rules or launch network services that might possibly expose the computer to an attack; and root can shut down the machine even if others are still using it. In summary, root can do almost anything, easily avoiding the restrictions that are normally put in place to prevent users from going too far.

Of course, a person logged in as root can make mistakes just like anyone else. A mistyped command can wipe out an entire file system or terminate a critical server, according to the annals of computing history. Then there's the risk of a hostile attack: if a user is signed in as root and leaves their desk, it's not difficult for a disgruntled coworker to gain access to their machine and cause havoc. Regardless, human nature being what it is, many administrators have been guilty of utilising root as their primary or only account over the years.

1.5 MANAGING SOFTWARE

One of the most common activities that any system admin has to perform is installing, patching, and deleting software packages on Linux workstations. Here's how to get started managing Linux packages in Red Hat-based Linux distributions (distros).

Installing, updating, deleting, and keeping track of software updates from specified repositories (repos) on the Linux system is known as package management. Different package management tools are used by different Linux distributions. RPM (RPM Package Manager) and YUM/DNF (Yellow Dog Updater, Modified/Dandified YUM) are used in Red Hat-based distributions.

Managing Packages using Yellow Dog Updater, Modified (YUM)

YUM is Red Hat Enterprise Linux's principal package management tool for installing, updating, uninstalling, and managing software packages. When installing, updating, or uninstalling software packages, YUM resolves dependencies. YUM can handle packages from the system's installed repositories or from.rpm packages. /etc/yum.conf is the main configuration file for YUM, while /etc/yum.repos.d contains all of the repositories.

For managing packages in Linux with YUM, At the command line, you need to enter:

yum -option command

There are various commands available to use with YUM. As well as different options are available with YUM. Some commonly used commands for YUM are listed below:

Table 1.1 Commands with YUM

Command	Use/Purpose
yum install	Installs the specified packages
remove	Removes the specified packages
search	Searches package metadata for keywords
info	Lists description
update	Updates each package to the latest version
repolist	Lists repositories
history	Displays what has happened in past transactions

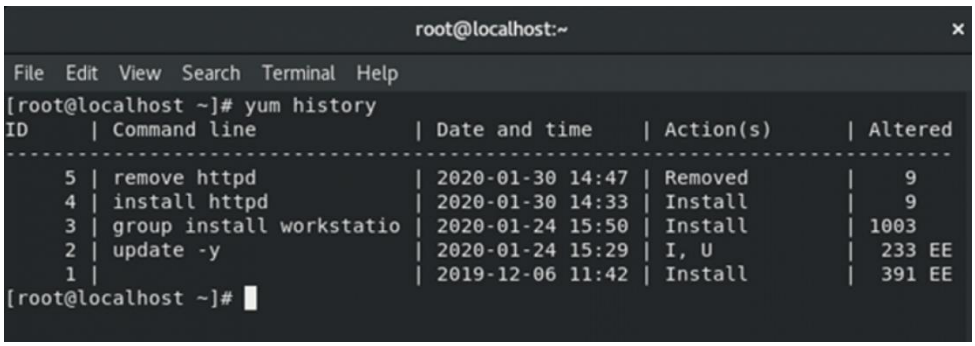
YUM provides many options for package management. For detailed option information, look at man yum and yum –help. Here is a list of some commonly used options with YUM:

Table 1.2 Options with YUM

Options	Use/Purpose
-C	Runs from system cache
--security	Includes packages that provide a fix for a security issue
-y	Answers yes to all questions
--skip-broken	Skips packages causing problems
-v	Verbose

Let's see how history option works with YUM. The history option provides you with a summary of previous transactions. This gives some valuable information, such as the transaction's date and the command that was executed. Use the following command:

yum history



You can undo or redo certain transactions using the history command using the following syntax:

yum history undo <id>

Managing Packages with RPM Package Manager (RPM)

In Red Hat Enterprise Linux-based distributions, RPM is a popular package management tool. Individual software packages can be installed, uninstalled, and queried using RPM. It still can't handle dependency resolution as well as YUM. RPM does produce helpful information, such as a list of required packages. A RPM package is made up of a collection of files and metadata. Helper scripts, file properties, and package information are all examples of metadata.

RPM keeps track of installed packages in a database, allowing for sophisticated and quick queries. The RPM database file is designated __db* and is located in /var/lib.

There are several basic modes in RPM: query, verify, install, upgrade, delete, show querytags, and show configuration. To perform package management tasks, at least one of these modes must be selected. Each mode offers a unique set of choices. Install mode I for example, offers its own set of installation settings. The modes' options can be found in the RPM man pages at man rpm.
Here is a list of some commonly used modes and options:

Table 1.3 Modes with RPM

Mode	Description
-i	Installs a package
-U	Upgrades a package
-e	Erases a package
-V	Verifies a package
-q	Queries a package

Table 1.4 Options with RPM

Options	Use/Purpose
-? | --help	Prints help
--version	Prints version number
-v	Prints verbose output

To install or upgrade an .rpm package using RPM, you can use the following command:

rpm -i package-file

rpm -U package-file

rpm -ivh package-file

The flag -i is for install, U is for upgrade, v for verbose, h for hash. In this example, v and h are optional flags.

To query for a package using RPM you can use following command:

rpm -q query-options package

rpm -qa vim-enhanced

Option a queries all installed packages on the system.

To erase a package, you can use the following command:

rpm -e erase-options package-name

rpm -evh vim-enhanced

1.6 SUMMARY

This chapter discussed Linux's command-line interface, different commands required for Linux installation. Package management is a common task for every system. YUM and RPM provide efficient ways to install, upgrade, remove, and track software packages on Red Hat Enterprise Linux systems.

1.7 REFERENCES

1. Linux Administration: A Beginner's Guide, Wale Soyinka, Seventh Edition, McGraw-Hill Education, 2016.
2. Ubuntu Server Guide, Ubuntu Documentation Team, 2016.
3. Mastering Ubuntu Server, Jay LaCroix, PACKT Publisher, 2016.
4. https://access.redhat.com/documentation
5. https://www.linux.com/

❄❄❄❄❄❄❄

2

SINGLE-HOST ADMINISTRATION

Unit Structure :

2.0 OBJECTIVES

After going through the unit, the learner will be able to:

- Demonstrate proficiency with the Linux command line for managing Users and Groups.
- Explain booting and shutting down process in Linx.
- Understand directory and file system organization.
- Illustrate various System services in Linux.
- To understand and practice the tools commonly found on most Linux distributions.
- State the subsystems of Linux kernel and it’s configuration.

2.1 INTRODUCTION

Linux system administration revolves around the management of users and groups. Because Linux is a multi-user operating system, multiple individuals can be logged in and actively working on a single machine. Allowing users to share the same account's credentials is never a good idea from a security standpoint. In fact, best practises recommend creating as many user accounts as the number of persons that need access to the machine. It is reasonable to suppose that two or more users will need to share access to specific system resources, such as directories and files, at the same time. In Linux, we may achieve both goals by using user and group administration. This chapter covers complex subjects including building group directories and discusses how to add, manage, and delete users and groups using the graphical user interface and the command line.

2.2 MANAGING USERS AND GROUPS

Users can be either persons (accounts related to physical users) or accounts that exist for certain applications to use. Groups are logical manifestations of organisation, connecting users together for a shared goal. Users in a group have the same permissions to access, write, and execute files that the group owns.

A user ID (UID) is a unique numerical identification number assigned to each user. Similarly, each group has its own unique ID (GID). The owner and group owner of a file are the same user who created it. The owner, the group, and everyone else have distinct read, write, and execute permissions on the file.

Linux checks the overall effective permissions on a resource to see if a user or group has access to files, programs, or other resources on the system. In Linux, the typical permissions paradigm is straightforward: there are four basic access types as well as two special access permissions. The following permissions and access types are available:

- Read permission (r)
- Write permission (w)
- Execute permission (x)
- No permission or no access (-)
- SetUID or SetGID permission (s) (special permission)
- Sticky bit (t) (special permission; useful for preventing users from deleting other
- users' files or directories)

These permissions can be applied to three classes of users:

- Owner-The owner of the file or application.
- Group-The group that owns the file or application.
- Everyone-All users.

Only root can alter the file owner, and both the root user and the file owner can change access rights. Adding a new user necessitates working with a different account than your own, which necessitates superuser (also known as root) access. Other user or group management activities, such as deleting an account, updating accounts, and creating and removing groups, follow the same pattern.

The table below shows the commands for these operations:

Table 2.1 Commands for Managing Users and Groups

Command	Operation
id	Displays user and group IDs.
useradd	Add a user to the system.
userdel	Delete a user account and related files.
usermod	Modify the user account.
groupadd	Add a group to the system.
groupdel	Remove a group from the system.
groupmod	Modify the group.
gpasswd	Utility primarily used for modification of group password in the /etc/gshadow file which is used by the newgrp command.
change	Change user password expiry information.
sudo	Run one or more commands as another user (typically with superuser permissions).

Relevant files for these utilities are **/etc/passwd** (user information), **/etc/shadow** (encrypted passwords), **/etc/group** (group information) and **/etc/sudoers** (configuration for sudo).

The su command or sudo can be used to change to the root user and get superuser permissions. You can check if sudo is installed on your machine by typing sudo in the command prompt.

which sudo

The package is installed if this command provides the absolute path to the related file (usually /usr/bin/sudo). Otherwise, you can install it with the following command:

apt-get install sudo

2.2.1 Adding a new user

To add a new user to the system, type the following command at a shell prompt as root:

useradd options username

The useradd command creates a locked user account by default. To unlock the account, run the following command to assign a password:

passwd username

The options for useradd commands are described in the following table:

Table 2.2 Options for useradd command

Option	Purpose
-c '*comment*'	*comment* can be replaced with any string. This option is generally used to specify the full name of a user.
-d *home_directory*	Home directory to be used instead of default /home/*username*/.
-e *date*	Date for the account to be disabled in the format YYYY-MM-DD.
-f *days*	Number of days after the password expires until the account is disabled. If 0 is specified, the account is disabled immediately after the password expires. If -1 is specified, the account is not disabled after the password expires.
-g *group_name*	Group name or group number for the user's default (primary) group. The group must exist prior to being specified here.
-G *group_list*	List of additional (supplementary, other than default) group names or group numbers, separated by commas, of which the user is a member. The groups must exist prior to being specified here.
-m	Create the home directory if it does not exist.
-M	Do not create the home directory.
-N	Do not create a user private group for the user.
-p *password*	The password encrypted with crypt.
-r	Create a system account with a UID less than 1000 and without a home directory.
-s	User's login shell, which defaults to /bin/bash.
-u *uid*	User ID for the user, which must be unique and greater than 999.

Example

sudo useradd -m olive

We have now created the user olive. If you look in the /home directory, you'll find the user's home (because we used the -m option, which creates a home directory).

Now, each user must have a password. To add password into the mix, we need to issue the following command:

sudo passwd olive

When you run this command, you will be prompted to enter (and verify) a new password for the user. And our user is created.

2.2.2 Adding a new group

To add a new group to the system, type the following command at a shell prompt as root:

groupadd options group_name

The options for groupadd commands are described in the following table:

Table 2.2 Options for groupadd command

Option	Purpose
-f, --force	When used with -g *gid* and *gid* already exists, groupadd will choose another unique *gid* for the group.
-g *gid*	Group ID for the group, which must be unique and greater than 999.
-K, --key *key=value*	Override /etc/login.defs defaults.
-o, --non-unique	Allows creating groups with duplicate GID.
-p, --password *password*	Use this encrypted password for the new group.
-r	Create a system group with a GID less than 1000.

Example

addgroup students

This command will create the group students. Now, if you issue the command less /etc/group, you can see the newly created group listed there.

With our group created, we need to add users. We'll add user Manthan to group students with the command:

sudo usermod -a -G students Manthan

Let's add the user olive to the group students with the command:

sudo usermod -a -G students olive

2.2.3 Adding an existing user to an existing group

We can use the usermod command to add an already existing user to an already existing group.

Various options of usermod command have different impact on user's primary group and on supplementary groups.

To override user's primary group, use the following command:

usermod -g group_name user_name

To override user's supplementary groups, use the following command:

usermod -G group_name1,group_name2,... user_name

2.2.4 Giving groups permissions to directories

System administrators frequently form a group for each significant project and assign employees to it when they need access to the project's files. File administration is tough with this traditional method; when someone creates a file, it is associated with the major group to which they belong. It's tough to associate the proper files with the right group when a single worker is working on many projects. The UPG system, on the other hand, assigns groups to files produced within a directory with the setgid flag set.

Let's assume that we have the directory /STUDENTS and we need to allow all members of the students group access to that directory. First, change the group of the folder with the command:

sudo chown -R :students /STUDENTS

Then, remove write permission from the group with the command:

sudo chmod -R g-w /STUDENTS

Now we remove the others x bit from the /STUDENTS directory (to prevent any user not in the readers group from accessing any file within) with the command:

sudo chmod -R o-x /STUDENTS

At this point, only the owner of the directory (root) and the members of the readers group can access any file within /STUDENTS.

Let's assume that we have the directory /TEACHERS and we need to give members of this group read and write permission to its contents. To do that, use the following command:

sudo chown -R :teachers /TEACHERS

sudo chmod -R g+w /TEACHERS

sudo chmod -R o-x /TEACHERS

At this point, any member of the teachers group can access and modify files within. All others (minus root) have no access to the files and folders within /TEACHERS.

Note: With this method, you can only add one group to a directory at a time. This is where access control lists come in picture.

2.2.5 Using Access Control Lists

Let's assume that we have a single folder—/DATA—and we want to give members of the students group read permission and members of the group teachers read/write permissions. For doing this, we can use **setfacl** command. The setfacl command sets file access control lists for files and folders.

The syntax of this command looks like:

setfacl OPTION X:NAME:Y /DIRECTORY

Where OPTION is the available options, X is either u (for user) or g (for group), NAME is the name of the user or group, and DIRECTORY is the directory to be used. We can use the option -m for modify. So, our command to add the group student for read access to the /DATA directory would look like this:

sudo setfacl -m g:students:rx -R /DATA

Now any member of the students group can read the files contained within /DATA, but they cannot modify them.

To give members of the teachers group read/write permissions (while retaining read permissions for the students group), we'd issue the command;

sudo setfacl -m g:teachers:rwx -R /DATA

The above command would give any member of the teachers group both read and write permission, while retaining the read-only permissions to the stduents group.

2.3 BOOTING AND SHUTTING DOWN PROCESSES

One of the most powerful features of Linux is its open approach of starting and terminating the operating system, which allows you to load specific programs based on their specifications, change those variables to manage the boot process, and shut down in a gentle and orderly manner. Beyond the topic of controlling the boot or shutdown process, Linux's open nature makes it much easier to pinpoint the actual root of most startup and shutdown issues. Anyone who uses a Linux system will benefit from having a basic understanding of this process.

For booting operating systems, many Linux systems use lilo, the LInux LOader.

2.3.1 The Boot process

When an x86 computer boots, the processor looks for the BIOS (Basic I/O System) at the end of the system memory and runs it.

The BIOS program is stored permanently in read-only memory and is constantly accessible. The BIOS regulates the first phase of the boot process

and offers the lowest level interface to peripheral devices. The BIOS runs tests on the system, looks for and checks peripherals, and then looks for a bootable drive. It usually scans for bootable media on the floppy drive (or CD-ROM drive) first, and then the hard drive. The sequence in which the drives are booted is normally determined by a BIOS setting on the computer.

This MBR offers instructions on how to use a pre-selected operating system to load the GRUB (or LILO) boot-loader. The boot-loader is then loaded by the MBR, which takes over the operation (if the boot-loader is installed in the MBR). GRUB uses the MBR settings to present boot options in a menu in the basic Red Hat Linux configuration. When GRUB receives the correct instructions for the operating system to start, whether from the command line or from a configuration file, it locates the required boot file and transfers control of the machine over to that operating system.

2.3.2 The Boot Loader

When a computer having Linux is turned on, a special program known as a boot loader loads the operating system into memory.

A boot loader is a program that runs on the system's primary hard drive (or other media device) and is responsible for loading the Linux kernel and its needed files, as well as other operating systems.

Each architecture capable of running Linux uses a different boot loader (GRUB/LILO).

There are at least two stages to a boot loader for the x86 architecture.

On the MBR, the first stage is a tiny machine code binary. Its main purpose is to identify and load the initial component of the second stage boot loader into memory. GRUB has the advantage of reading ext2 and ext3 partitions and loading its configuration file — **/boot/grub/grub.conf** — at boot time.

When the second stage boot loader is loaded into memory, it displays a graphical panel with the various operating systems or kernels that can be booted. On this page, a user can select the operating system or kernel they want to boot using the arrow keys and then hit Enter.

The boot loader loads the default choices after a configurable amount of time has passed if no key is touched.

The second stage boot loader locates the relevant kernel binary in the /boot/ directory once it has determined which kernel to boot. **/boot/vmlinuz-kernel-version>** file (where kernel-version> corresponds to the kernel version given in the boot loader's parameters) is the name of the kernel binary.

After that, the boot loader loads one or more initramfs images into memory.

The kernel then uses cpio to decompress these images from memory and write them to **/boot/**, a RAM-based virtual file system. The kernel loads drivers and modules required to boot the system from the initramfs.

The boot loader gives control of the boot process to the kernel once the kernel and initramfs image(s) are loaded into memory.

2.3.2.1 GRUB

The GNU GRand Unified Boot loader (GRUB) is a programme that allows you to choose which operating system or kernel to load when your computer starts up. It also allows the user to give the kernel arguments.

Because instructions are utilised to directly load the operating system, there is no intermediary code between the boot-loaders and the operating system's major files (such as the kernel), this boot approach is known as direct loading.

Other OS systems' boot processes may change slightly from the one described above. When Microsoft's DOS and Windows operating systems are installed, for example, they fully wipe anything on the MBR without adopting any of the present MBR's configuration. Other operating systems, such as Linux, will lose any data stored in the MBR as a result of this.

GRUB supports both boot techniques, allowing you to use it with almost any operating system, most popular file systems, and nearly any hard disc that your BIOS can detect.

Features of GRUB:

1. On x86 platforms, GRUB provides a complete command-based pre-OS environment, allowing maximal flexibility in loading operating systems with specific settings or obtaining system information.

2. Logical Block Addressing (LBA) mode is supported by GRUB, which is required to access many IDE and all SCSI hard drives. Prior to LBA, hard drives might reach a 1024-cylinder limit, after which the BIOS would be unable to locate a file.

3. Every time the system boots, GRUB's configuration file is read from disc, saving you from having to write over the MBR every time you modify the boot choices.

2.3.2.2 LILO

LILO (LInux LOader) is a boot loader for the Linux operating system. Most new computers come pre-installed with boot loaders for Microsoft Windows or Mac OS. A unique boot loader must be installed on a computer before it can be used with Linux. LILO is the most widely used boot loader among Linux users that use it as their primary or sole operating system.

When a computer is turned on or restarted with LILO installed, the basic input/output system (BIOS) runs several preliminary tests before transferring control to the Master Boot Record (MBR), which contains LILO. The main advantage of LILO is that when put in the MBR, it enables for quick booting of Linux. Its biggest disadvantage is that not all PCs will allow you to change the MBR. There are alternative techniques of employing LILO in similar scenarios, but they take longer.

Other boot loaders, such as LOADLIN (LOAD LINux) and GRUB, can be used to boot Linux into a computer's memory instead than LILO (GRand Unified Bootloader).

2.3.2 The INIT process

Once loaded, the kernel looks for **init** in **sbin** and runs it.

When **init** starts, it becomes the parent or grandparent of all the processes on your Linux system that start up automatically.

The first thing init does is read **/etc/inittab**, which is its initialization file. This tells init to read the environment's first setup script, which establishes the path, starts swapping, and checks the file systems, among other things. Essentially, this step handles everything that needs to be done during system initialization, such as setting the clock and initialising serial ports.

Then **init** reads the **/etc/inittab** file, which specifies how the system should be configured in each run level and establishes the default run level. A process configuration is referred to as a run level. All UNIX-like systems can be configured to run in various process configurations, such as single user mode (also known as run level 1 or run level S) (or s). Only the system administrator can connect to the system in this mode. It is used to do maintenance operations without endangering the system or the data of the users. We don't need to provide user services in this configuration, thus they'll all be disabled.

The reboot run level, also known as run level 6, shuts down all running services and then restarts the system according to the required procedures.

To find out what your current run level is, use the who command:

who -r

Init looks in the appropriate rc directory for that run level after determining the default run level for your system. Init then launches all of the background processes required for the system to execute. Each of the kill scripts (whose file names begin with a K) is run with a stop parameter by init. It then runs all of the start scripts in the proper run level directory (their file names begin with a S) to ensure that all services and apps are launched correctly. In fact, after the system has finished booting, you can manually run these scripts with a command like **/etc/init.d/httpd stop** or **service httpd stop** logged in as root, which will stop the web server in this situation.

The **/etc/inittab** script forks a getty process for each virtual console (login prompt in text mode) after init has progressed through the run levels to reach the default run level.

getty opens tty lines, configures their modes, prints the login prompt, retrieves the user's name, and then begins the login procedure for that user. This allows users to log in and use the system after authenticating themselves.

Most systems come with six virtual consoles by default, but as you can see from the **inittab** file, this can be changed.

Init can also be told how to handle a user hitting **Ctrl+Alt+Delete** at the console using **/etc/inittab**. **Init** is told to run the command **/sbin/shutdown -t3 -r now**, for example, when a user presses certain keys, because the system should be properly shut down and restarted rather than power-cycled. In addition, if your system includes a UPS unit, **/etc/inittab** specifies what **init** should perform in the event of a power outage.

The graphical login screen is launched on run level 5 on most RPM-based systems, where **/etc/inittab** runs a script named **/etc/X11/prefdm**. The prefdm script uses the contents of the **/etc/sysconfig/desktop** directory to execute the preferred X display manager. If you're using GNOME, this is usually gdm, and if you're using KDE, it's kdm, although they can be combined, and there's also the xdm that comes with a regular X installation.

2.3.3 Shutdown in LINUX

Although UNIX was not designed to be shut down, the **shutdown** command can be used if necessary. The **-h** option will stop the system when it has completed the shutdown routine, whereas the **-r** option will reboot it.

Only root user can execute shutdown command. If run when the system is in run levels 1-5, the reboot and halt commands can now invoke shutdown, ensuring correct shutdown of the system. However, this is a hazardous habit to get into, as not all UNIX/Linux versions have this feature.

Syntax:

shutdown [OPTIONS] [TIME] [MESSAGE]

- **options** – 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.

Options

- -r : Requests that the system be rebooted after it has been brought down.
- -h : Requests that the system be either halted or powered off after it

has been brought down, with the choice as to which left up to the system.

- -H : Requests that the system be halted after it has been brought down.
- -P : Requests that the system be powered off after it has been brought down.
- -c : Cancels a running shutdown. TIME is not specified with this option, the first argument is MESSAGE.
- -k : Only send out the warning messages and disable logins, do not actually bring the system down.

Example

sudo shutdown 05:00

The above command will schedule a system shutdown at 5 A.M.

If your computer does not shut down automatically, wait until you get a notification indicating that the system is stopped or completed shutting down before turning it off to give the system time to unmount all partitions. Impatience can lead to data loss.

2.4 FILE SYSTEMS

Every general-purpose computer requires a hard disc drive (HDD) or its equivalent, such as a USB memory stick, to store various sorts of data. This is due to a number of factors. First, when the computer is turned off, the contents of RAM are lost. The second argument for storing data on hard drives is that even normal RAM is more expensive than disc space.

A **file system** is a logical grouping of files on a disc or partition. A **partition** is a data container that can span the full hard drive if necessary.

Your hard disc can have multiple partitions, each of which normally only contains one file system, such as one for the /file system and another for the /home file system. Different file systems can be logically maintained and managed using one file system per partition.

In Unix/Linux, everything including physical media like DVD-ROMs, USB devices, and floppy drives, is considered a file.

2.4.1 Linux File System Structure

The Linux file system contains the following parts:

- The root directory (/) which contains other files and directories.
- A specific data storage format (EXT3, EXT4, BTRFS, XFS and so on).
- A partition or logical volume having a particular file system.

Because it has a root directory and its subdirectories, the Linux file system has a hierarchical file structure. The root directory provides access to all other directories. Normally, a partition has just one file system, although it might have many file systems.

A file system is constructed in such a way that it can manage and store non-volatile data. A namespace, which is a naming and organizational mechanism, is essential for all file systems. The namespace specifies the naming procedure, file name length, or a subset of characters that can be used in the file name. It also specifies the logical structure of files on a memory segment, such as the use of directories for file organization. Once a namespace has been formed, a Metadata description for that specific file must be defined.

A hierarchical directory structure must be supported by the data structure. This data structure is used to indicate the available and utilized disc space for a specific block. It also contains information about the files, such as file size, creation date and time, update, and last edited. It also saves sophisticated data about the disk's section, such as partitions and volumes. The advanced data and structures it depicts contain information about the file system that is stored on the drive; it is separate and independent from the file system metadata.

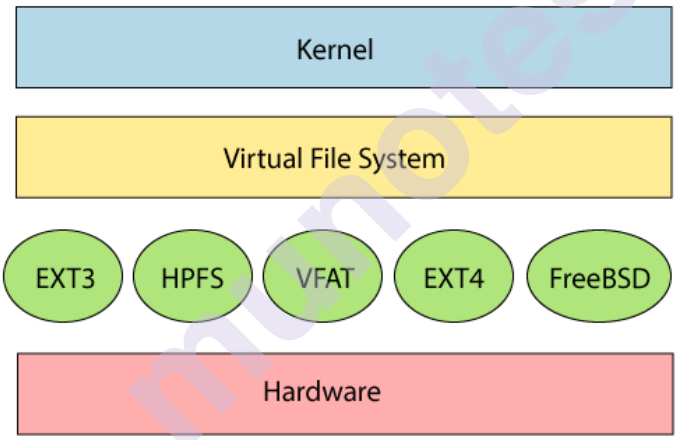


Fig 2.1 Linux File system structure

As shown in the Fig 2.1, Linux file system contains two-part file system software implementation architecture.

To communicate with file system components such as files and directories, an API (application programming interface) is required. Creating, deleting, and copying files are all made easier with API. It facilitates the implementation of an algorithm that defines the file system's file organization. A Linux virtual file system is made up of the first two components of a given file system. It gives the kernel and developers a single set of commands to access the file system. This virtual file system necessitates the use of a specialized system driver to provide a file system interface.

2.4.2 Types of Linux File System

Ext, Ext2, Ext3, Ext4, JFS, ReiserFS, XFS, btrfs, and swap are among the file systems available when installing the Linux operating system.

Ext stands for Extended File System. The Ext file system is an earlier version that is no longer utilized due to several restrictions. It was designed primarily for MINIX OS.

Ext2 is the first Linux file system to support data storage of up to two terabytes. Ext3 is based on Ext2, and it is an enhanced version of Ext2 with backward compatibility. The main disadvantage of Ext3 is that it does not support servers since it lacks file recovery and disc snapshot capabilities. Among all the Ext file systems, the Ext4 file system is the fastest. It is the default file system in Linux distributions and is a very compatible solution for SSD (solid-state drive) devices.

JFS stands for Journaled File System and was created by IBM for the AIX Unix operating system. It's a replacement for the Ext file system. It can also be used instead of Ext4 when a high level of stability is required with limited resources. When CPU power is restricted, it is a useful file system.

ReiserFS is a file system that replaces the Ext3 file system. It includes more advanced features and increased performance. ReiserFS was previously the default file system in SUSE Linux, however it was replaced with Ext3 after the company changed its rules. Although this file system dynamically supports the file extension, it has some performance issues.

Fig 2.2 Linux File system types

The **XFS** file system is a high-speed JFS that was designed for parallel I/O processing. With its high storage server, NASA continues to use this file system.

The B tree file system is abbreviated as **Btrfs**. It serves as a fault tolerance system, a repair system, a fun administration system, a large storage configuration system, and more. It is unsuitable for the manufacturing system.

During system hibernation, the **swap** file system is utilised for memory paging in the Linux operating system. A system that does not go into hibernation must have swap space equal to its RAM size.

2.4.3 The Directory Structure

All physical hard drives and partitions are combined into a single directory structure by the Linux filesystem. Everything begins at the top, with the root (/) directory. The single Linux root directory contains all other directories and their subdirectories. This means that searching for files and programmes is limited to a single directory tree.

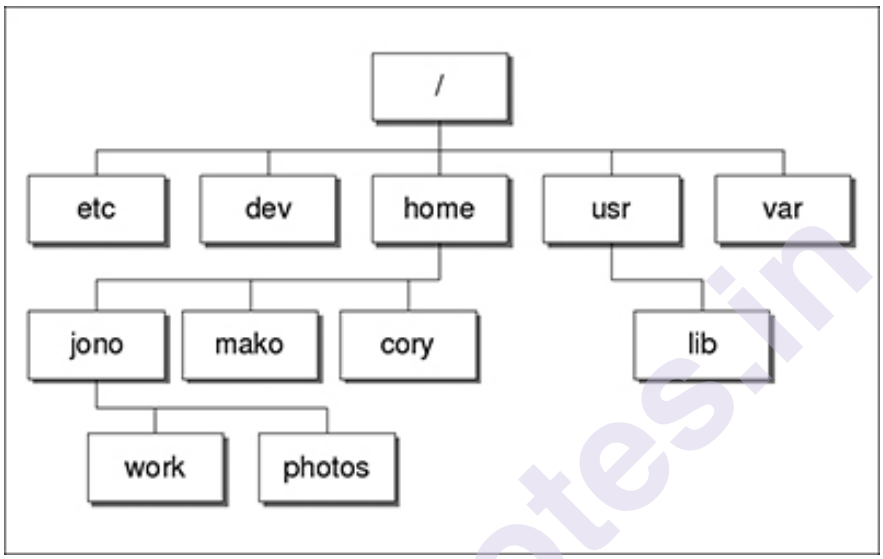


Fig 2.3 Linux Directory Structure

Each file or directory is identifiable by its name, the directory it resides in, and a unique identifier, commonly referred to as an **inode**. The inode number of the root directory is 2, and the inode number of the lost+found directory is 3. The numerals 0 and 1 are not utilized in inodes. The -i option to the ls command can be used to view file inode numbers. Between one filesystem and another, there are no dependencies.

Every directory serve a specific purpose and typically contain the same types of information to make file searching easier. The directories that exist on the major Linux versions are shown below:

Table 2.3 Directories in Linux

Sr.No.	Directory	Description
1	/	This is the root directory which should contain only the directories needed at the top level of the file structure
2	**/bin**	This is where the executable files are located. These files are available to all users
3	**/dev**	These are device drivers

4	**/etc**	Supervisor directory commands, configuration files, disk configuration files, valid user lists, groups, ethernet, hosts, where to send critical messages
5	**/lib**	Contains shared library files and sometimes other kernel-related files
6	**/boot**	Contains files for booting the system
7	**/home**	Contains the home directory for users and other accounts
8	**/mnt**	Used to mount other temporary file systems, such as **cdrom** and **floppy** for the **CD-ROM** drive and **floppy diskette drive**, respectively
9	**/proc**	Contains all processes marked as a file by **process number** or other information that is dynamic to the system
10	**/tmp**	Holds temporary files used between system boots
11	**/usr**	Used for miscellaneous purposes, and can be used by many users. Includes administrative commands, shared files, library files, and others
12	**/var**	Typically contains variable-length files such as log and print files and any other type of file that may contain a variable amount of data
13	**/sbin**	Contains binary (executable) files, usually for system administration. For example, ***fdisk*** and ***ifconfig*** utlities
14	**/kernel**	Contains kernel files

2.4.4 Mounting the File system

In Linux, the term "to mount" a filesystem harkens back to the days when a tape or detachable disc pack had to be physically mounted on a suitable drive device. The operating system would logically mount the filesystem on the disc pack once it was physically placed on the drive, making the contents available for access by the OS, application applications, and users.

A mount point is simply a directory established as part of the root filesystem, just like any other. The home filesystem, for example, is

mounted on the directory /home. Mount points on other non-root filesystems can be used to mount filesystems, but this is less frequent.

Early in the boot procedure, the Linux root filesystem is mounted on the root directory (/). Other filesystems are mounted later by the Linux startup programmes, which are either rc in SystemV or systemd in newer Linux releases. The /etc/fstab configuration file controls how filesystems are mounted during the startup process. The acronym fstab stands for "file system table," and it is a list of filesystems to be mounted, their specified mount points, and any parameters that might be required for individual filesystems.

The **mount** command is used to mount filesystems on an existing directory/mount point.

mount -t file_system_type device_to_mount directory_to_mount_to

Example

mount -t iso9660 /dev/cdrom /mnt/cdrom

This command will mount a CD-ROM to the directory /mnt/cdrom, This assumes that your CD-ROM device is called /dev/cdrom.

The mount command enables you to access your file systems, but on most modern Unix systems, the automount function makes this process invisible to the user and requires no intervention.

2.4.4 Unmounting the File system

To unmount/remove the file system from your system, you can use the **umount** command after identifying the mount point or device.

Example

umount /dev/cdrom

This command will unmount cdrom.

2.5 CORE SYSTEM SERVICES

Every Linux-based system comes with some essential services, regardless of distribution, network setup, or overall system design. Systemd, init, the logging daemon, cron, and other services are among them. Although the functions performed by these services appear to be basic, they are quite important. A lot of Linux's power and utility would be lost without them.

To start and run these services, use the following command:

service "service name" start/stop/status/restart

And to check if this service is running, use the command:

service “service name” status

2.5.1 Init.d

All of these services rely on scripts, which are kept in the /etc/init.d directory. In a Linux system, Init.d is a deamon, which is the first process that the kernel runs when a system boots up. It's a daemon process that keeps running until the system is turned off. As a result, it is the mother of all processes.

To begin, init reads the script stored in the /etc/inittab file. The command init reads the initial configuration script, which takes care of everything a system has to perform during startup, such as setting the clock and initialising the serial port.

Thus, Init starts other processes, services, daemons, and threats. So, init.d is the init process's configuration database. A daemon script contains functions such as start, stop, status, and restart.

Init decides how the system should be set up in each run level and sets the default runlevel by reading this file. Init initiates all background processes required to run the system after identifying the system's default runlevel. It starts by passing a stop argument to each of the kill scripts (their file names begin with a K). Then it launches all services and applications by running their start scripts (their file names begin with a S).

/sbin/telinit is linked up with **/sbin/init**. **Telinit** command takes up an argument and signals **init** to perform the respective function. List of possible arguments is given below:

Table 2.4 Arguments for Tellinit

Argument	Function
0,1,2,3,4,5,6	Switch to specified runlevel
a,b,c	Processes only file entries from **/etc/inittab** having runlevel a,b,c.
Q or q	Re-examine file **/etc/inittab**
S or s	Switch to single user mode
U or u	init re-execute itself. No re-examine occurs, runlevel should be from S,s,1,2,3,4,5 otherwise request would be ignored silently.

Numbers are used to identify runlevels. There are eight different runlevels in which init can be found. A privileged user can alter it by running **telinit**, which sends suitable signals to **init** to adjust the runlevel.

Table 2.5 Runlevels

Runlevel	Function
0 (Reserved)	Halt the system
1 (Reserved)	Single user mode
2	Multiuser mode without networking
3	Multiuser mode with networking
4	Not used
5	Multiuser with networking and X windows
6 (Reserved)	Reboot the system
S/s	Not used directly
7-9	Same as runlevels S or s (aliased)

2.5.2 Systemd

For Linux operating systems, Systemd is a system and service manager. It includes capabilities such as parallel launch of system services at boot time, on-demand activation of daemons, and dependency-based service control logic, and is designed to be backwards compatible with SysV init scripts. The default init system in Linux is systemd, which replaces Upstart.

Control Groups (cgroups)

cgroups is a kernel feature that allows processes to be organised hierarchically and named separately. Every process that systemd begins is placed in a control group named after the service. This helps systemd to maintain track of processes and get a better understanding and control of a service over its lifetime. Systemd, for example, may securely terminate or destroy a process, as well as any child processes it may have created.

Socket Activation

The advantages/advantages of systemd arise from its proper and intrinsic grasp of the interconnectedness of system services—that is, it understands what diverse system services require of one another. Most startup services or daemons, it turns out, simply require the sockets offered by specific services, not the high-level services themselves. Because systemd is aware of this, it ensures that any required sockets are available as soon as the system boots up. It eliminates the need to first start a service that provides both a service and a socket.

Unit

The concept of systemd units is introduced by Systemd. Units are the building blocks of 37ystem, and they are the objects that 37ystem manages. Services or daemons, devices, file system entities like mount points, and so

on are examples of these objects. Units are named after the configuration files they use, which are usually found in the /etc/38ystem/system/ directory. The /lib/38ystem/system directory contains standard unit configuration files. For actual use, any required files must be copied to the working /etc/38ystem/system/ folder. The /run/38ystem/system/ directory, which is itself a temporary file system structure, stores transient or ephemeral units created during system runtime.

These units are represented by unit configuration files located in one of the directories listed in table below:

Table 2.6 Systemd unit file locations

Directory	Description
/usr/lib/38ystem/system/	Systemd unit files distributed with installed RPM packages.
/run/38ystem/system/	Systemd unit files created at run time. This directory takes precedence over the directory with installed service unit files.
/etc/38ystem/system/	Systemd unit files created by systemctl enable as well as unit files added for extending a service. This directory takes precedence over the directory with runtime unit files.

List of available 38ystem unit types is given in the table below:

Table 2.7 Systemd unit types

Unit Type	File Extension	Description
Service unit	.service	A system service.
Target unit	.target	A group of 38ystem units.
Automount unit	.automount	A file system automount point.
Device unit	.device	A device file recognized by the kernel.
Mount unit	.mount	A file system mount point.
Path unit	.path	A file or directory in a file system.
Scope unit	.scope	An externally created process.
Slice unit	.slice	A group of hierarchically organized units that manage system processes.

Snapshot unit	.snapshot	A saved state of the 39ystem manager.
Socket unit	.socket	An inter-process communication socket.
Swap unit	.swap	A swap device or a swap file.
Timer unit	.timer	A 39ystem timer.

2.5.3 xinetd and inetd

The **xinetd** daemon is super service which controls access to a subset of popular network services like FTP, IMAP, and Telnet.Sservice-specific configuration options for access control, enhanced logging, binding, redirection, and resource utilization control is also provided by xinetd.

When a client attempts to connect to a network service controlled by xinetd, the service receives the request and checks for any TCP Wrappers access control rules.

When access is allowed, xinetd verifies that the connection is allowed under its own access rules for that service. It also checks that the service can have more resources assigned to it and that it is not in violation of any defined rules.

The service xinetd then starts an instance of the requested service and passes control of the connection to it. After the connection has been established, xinetd takes no further part in the communication between the client and the server.

The configuration files for xinetd are:

/etc/xinetd.conf -The global xinetd configuration file. The /etc/xinetd.conf file contains general configuration settings which affect every service under xinetd's control. It is read when the xinetd service is first started, so for configuration changes to take effect, you need to restart the xinetd service.

/etc/xinetd.d/ - The directory containing all service-specific files. The configuration files for each service managed by xinetd and the names of the files correlated to the service are contained within the /etc/xinetd.d/ directory. Similar to xinetd.conf, this directory is read only when the xinetd service is started. For any changes to take effect, the administrator must restart the xinetd service.

Xinetd is successor of inetd. The service **inetd** maintains a list of passive sockets for various services configured to run on that particular machine on Unix-like operating systems. The inetd will start a program to handle the connection based on the configuration files whenever the client tries to connect to one of the services. Thus, inetd will run the server programs as they are needed by creating multiple processes to service multiple network connections.

Once a list of services is known, inetd watches those services' ports and protocols for requests. Whenever an activity occurs, inetd maps the incoming request to standard input (stdin), standout output (stdout), and standard error (stderr) and launches the proper daemon. The inetd service processes the data and terminates. It keeps resource consumption to a minimum and makes daemons easier to write.

2.5.4 The Logging Daemon

Unix frameworks have a truly adaptable and incredible logging framework, which empowers you to record nearly anything you can envision and afterward control the logs to recover the data you require.

Numerous renditions of Unix give a broadly useful logging office called **syslog**. Individual projects that need to have data logged, send the data to syslog.

Unix syslog is a host-configurable, uniform framework logging facility. The framework utilizes an incorporated framework logging measure that runs the program **/etc/syslogd** or **/etc/syslog**.

The activity of the framework lumberjack is very clear. Projects send their log passages to **syslogd**, which counsels the configuration file **/etc/syslogd.conf** or **/etc/syslo**g and when a match is found, composes the log message to the ideal log record.

Table 2.8 Syslog terms

Sr. No.	Term	Description
1	**Facility**	The identifier used to describe the application or process that submitted the log message. For example, mail, kernel, and ftp.
2	**Priority**	An indicator of the importance of the message. Levels are defined within syslog as guidelines, from debugging information to critical events.
3	**Selector**	A combination of one or more facilities and levels. When an incoming event matches a selector, an action is performed.
4	**Action**	What happens to an incoming message that matches a selector — Actions can write the message to a log file, echo the message to a console or other device, write the message to a logged in user, or send the message along to another syslog server.

The mix of facilities and levels empowers you to be knowing with regards to what is logged and where that data goes.

As each program sends its messages obediently to the framework logger, the logger settles on choices on what to monitor and what to dispose of dependent on the levels characterized in the selector.

At the point when you determine a level, the framework will monitor everything at that level and higher. Following is the list of available facilities for the selector:

Table 2.9 Syslog facilities

Sr. No.	Facility	Description
1	**auth**	Activity related to requesting name and password (getty, su, login)
2	**authpriv**	Same as auth but logged to a file that can only be read by selected users
3	**console**	Used to capture messages that are generally directed to the system console
4	**cron**	Messages from the cron system scheduler
5	**daemon**	System daemon catch-all
6	**ftp**	Messages relating to the ftp daemon
7	**kern**	Kernel messages
8	**local0.local7**	Local facilities defined per site
9	**lpr**	Messages from the line printing system
10	**mail**	Messages relating to the mail system
11	**mark**	Pseudo-event used to generate timestamps in log files
12	**news**	Messages relating to network news protocol (nntp)
13	**ntp**	Messages relating to network time protocol
14	**user**	Regular user processes
15	**uucp**	UUCP subsystem

The syslog priorities are summarized in the following table:

Table 2.10 Syslog priorities

Sr. No.	Priority	Description
1	**emerg**	Emergency condition, such as an imminent system crash, usually broadcast to all users
2	**alert**	Condition that should be corrected immediately, such as a corrupted system database
3	**crit**	Critical condition, such as a hardware error
4	**err**	Ordinary error
5	**Warning**	Warning
6	**notice**	Condition that is not an error, but possibly should be handled in a special way
7	**info**	Informational message
8	**debug**	Messages that are used when debugging programs
9	**none**	Pseudo level used to specify not to log messages

The **/etc/syslog.conf** file controls where messages are logged. Each line of the file contains a message selector that specifies which kind of messages to log and an action field that says what should be done with the message.

To deal with system logging Unix provides the **logger** command, which is an extremely useful command. This command sends logging messages to the **syslogd** daemon, and consequently provokes system logging. The logger command provides a method for adding one-line entries to the system log file from the command line.

The syntax of the command is

logger [-i] [-f file] [-p priority] [-t tag] [message]...

Here is a list of parameters for the logger command:

Table 2.11 Parameters for logger command

Sr. No.	Option	Description
1	**-f filename**	Uses the contents of file filename as the message to log.
2	**-i**	Logs the process ID of the logger process with each line.
3	**-p priority**	Enters the message with the specified priority (specified selector entry); the message priority can be specified numerically, or as a facility.priority pair. The default priority is user.notice.
4	**-t tag**	Marks each line added to the log with the specified tag.
5	**message**	The string arguments whose contents are concatenated together in the specified order, separated by the space.

2.5.5 systemd-journald (journald)

The **journal** is a part of systemd. It's a concentrated area for all messages logged by various parts in a systemd-empowered Linux framework. This incorporates bit and boot messages, messages coming from syslog, or various administrations.

During the boot-up process of conventional Linux, separate subsystems of the OS, or application daemons, would log all of their messages in various text files around the system. Each subsystem would keep track of its communications in different ways. When troubleshooting, an administrator would frequently have to look through messages from numerous files and correlate the items from different time frames. Journaling solves this problem by storing both OS and application level messages in a single location.

The systemd-journald daemon is in charge of the journal. It gathers data from several sources and enters the messages into the journal.

The systemd journal is a relatively small text file. It's a binary file that the daemon keeps track of. As a result, a text editor will not open it. The location

and size of this binary file are controlled by the daemon's configuration file, as we'll learn later. It also doesn't have to be persistent; using configuration parameters, an administrator can disable journaling entirely or keep it in memory, making it volatile. Systemd creates journal files in the /run/log/journal directory when using in-memory journaling. If the directory does not exist, it is created.

The journal is created under the /var/log/journal directory with persistent storage; the directory is created by systemd if necessary. If this directory is removed for whatever reason, systemd-journald will not automatically recreate it; instead, it will write the logs in a non-persistent manner to /run/log/journal. When the daemon is restarted, it will recreate the directory.

2.5.6 The cron Program

Cron is a Linux-like operating system's software application that automates a scheduled task at a defined time. It is a daemon process that runs in the background and conducts the specified tasks at a predetermined time when a specific event or condition occurs without the need for user participation. Dealing with a repetitive task on a regular basis might be scary for a system administrator, so he can use cron to create a list of commands that will execute automatically in the background at regular intervals.

It allows users to perform planned tasks invisibly on a regular basis, such as backing up files every day at midnight, scheduling updates on a weekly basis, and synchronising files at regular intervals. Cron checks for the planned task on a regular basis, and the scheduled commands are executed when the scheduled time fields match the current time fields. When you enter multi-user run levels, it starts immediately from **/etc/init.d**.

Syntax for the command:

cron [-f] [-l] [-L loglevel]

Options:

-f : Used to stay in foreground mode, and don’t daemonize.

-l : This will enable the LSB compliant names for /etc/cron.d files.

-n : Used to add the FQDN in the subject when sending mails.

-L loglevel : This option will tell the cron what to log about the jobs with the following values:

1 - It will log the start of all cron jobs.

2 - It will log the end of all cron jobs.

4 - It will log all the failed jobs. Here the exit status will not equal to zero.

8 - It will log the process number of all the cron jobs.

The **crontab** (short for "cron table") is a set of instructions that allows you to run scheduled actions at a given time. It allows the user to add, remove, or change the tasks that are scheduled. The crontab command syntax contains six fields separated by spaces, the first five of which reflect the task time and the final of which is for the command.

1. Minute (holds a value between 0-59)
2. Hour (holds value between 0-23)
3. Day of Month (holds value between 1-31)
4. Month of the year (holds a value between 1-12 or Jan-Dec, the first three letters of the month's name shall be used)
5. Day of the week (holds a value between 0-6 or Sun-Sat, here also first three letters of the day shall be used)
6. Command

2.6 LINUX KERNEL

The Linux kernel is the core interface between a computer's hardware and its programmes and is the most important component of a Linux operating system (OS). It communicates between the two, allowing for the most efficient use of resources.

The kernel is so named because it exists within the OS and handles all of the key functions of the hardware, whether it's a phone, laptop, server, or any other type of computer, much like a seed inside a hard shell.

The following are the jobs of a kernel:

- Memory management: Keep track of how much memory is used to store what, and where.
- Process management: Determine which processes can use the central processing unit (CPU), when, and for how long.
- Device drivers: Act as mediator/interpreter between the hardware and processes.
- System calls and security: Receive requests for service from the processes.

If built correctly, the kernel is completely invisible to the user, operating in its own small universe known as kernel space, where it allocates memory and maintains track of where everything is stored. The user space refers to what the user sees, such as web browsers and files. A system call interface is used by these apps to communicate with the kernel (SCI).

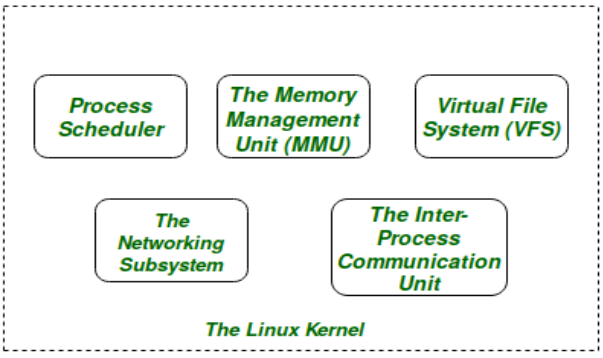


Fig 2.4 Linux Kernel

The Linux Kernel consists of following subsystems:

1. The Process Scheduler
2. The Memory Management Unit (MMU)
3. The Virtual File System (VFS)
4. The Networking Unit
5. Inter-Process Communication Unit

The Process Scheduler:

This kernel subsystem is in charge of distributing CPU time evenly among all programmes operating on the system at the same time.

The Memory Management Unit:

This kernel sub-unit is in charge of allocating memory resources appropriately among the numerous programmes executing on the system. The MMU does more than merely provide different virtual address spaces for each activity.

The Virtual File System:

This subsystem is in charge of providing a unified interface for accessing data stored across multiple filesystems and physical storage media.

The system's code runs in one of two modes on CPUs: kernel mode or user mode. In kernel mode, code has full access to the hardware, but in user mode, the SCI has limited access to the CPU and memory. Memory is divided in a similar way (kernel space and user space). These two modest features serve as the foundation for more complex activities such as security privilege separation, container construction, and virtual machine creation.

2.7 PROCESS OF CONFIGURING, COMPILING

The configuration of the current kernel is stored in the file /proc/config.gz. To modify kernel configuration, go to the directory /usr/src/linux as root and type the following commands:

zcat /proc/config.gz > .config

make oldconfig

The command build oldconfig creates a template for the current kernel configuration from the file /usr/src/linux/.config. Your current kernel sources will be queried for any new options. The default configuration contained in the kernel sources will be utilised if the file.config does not exist.

Change to /usr/src/linux and run the make config command to configure the kernel. Choose the features you want the kernel to support. There are usually two or three options: y, n, or m are the possible answers. This device

will be loaded as a module rather than being built directly into the kernel. With y, drivers required for system booting must be incorporated within the kernel. To validate the default settings retrieved from the.config file, press Enter. To display a quick help text regarding a certain option, press any other key.

2.8 SUMMARY

This chapter gives us details about managing Users and Groups in Linux system. Also, the file system of Linux was seen in detail. The many aspects of starting and shutting down a standard Linux system were covered in this chapter. We began our investigation with the all-powerful boot loader. Because it is the boot loader of choice among popular Linux distributions, we looked at GRUB in particular as an example boot loader/manager.

In this chapter, we looked at some of the most significant system services that most Linux systems have. These services are stand-alone (i.e., they can operate whether or not the system is in multiuser mode) and thus do not require network support. Their implementation may differ slightly from host to host or distribution to distro, but the essential goals and problems they seek to answer are the same. The process of setting and constructing the Linux kernel was covered in this chapter. Compiling the kernel is a rather simple procedure. The Linux community has created wonderful tools to make the procedure as painless as possible.

2.9 REFERENCES

1. Linux Administration: A Beginner's Guide, Wale Soyinka, Seventh Edition, McGraw-Hill

Education, 2016.

2. Ubuntu Server Guide, Ubuntu Documentation Team, 2016.
3. Mastering Ubuntu Server, Jay LaCroix, PACKT Publisher, 2016.
4. https://access.redhat.com/documentation
5. https://www.linux.com/

❄❄❄❄❄❄❄

3

NETWORKING AND SECURITY

Unit Structure :

3.0 OBJECTIVES

- After going through the unit, the learner will be able to:
- Gain the basic knowledge about TCP/IP and OSI model.
- Impart understanding of protocol headers used in Transport layer.
- Elaborate commands related to basic Network Configuration.
- Understand working of Netfilter.
- Gain the knowledge about network security and tools used.

3.1 INTRODUCTION

This chapter provides an introduction to the insight of the Transmission Control Protocol / Internet Protocol (TCP/IP). We start with a discussion of TCP/IP and OSI model. We will learn different protocol headers starting with Ethernet header. We will alsosee how Linux Netfilter works. This chapter also addresses the fundamentals of keeping your system secure against security attacks.

3.2 TCP/IP FOR SYSTEM ADMINISTRATION

3.2.1 TCP/IP and OSI model

The **OSI (Open System Interconnection) standard** was created in the 1980s. It's a type of conceptual network communication model. It is not fully executed, yet it is still referred to today. This OSI model has seven layers, each of which is related to the others. Each layer of the OSI model adds more information to the data as it passes down the model. The data descends until it reaches the OSI model's last layer. The data is transported via the network after it is received at the last layer of the OSI architecture. The process will be reversed after the data has been received on the opposite side.

TCP/IP model consists of Transmission Control Protocol (TCP) and Internet Protocol (IP). The TCP/IP paradigm may encompass several protocols that make up the internet. Nowadays, we don't hear much about the TCP/IP model; instead, we hear about IPv4 or IPv6, but it's still valid. There are four layers in this model.

3.2.1.1 Layers of OSI model

- **Physical Layer**

  The lowest layer of the OSI Model is concerned with electrically or optically transmitting raw unstructured data bits across the network from the physical layer of the sending device to the physical layer of the receiving device. It can include specifications such as voltages, pin layout, cabling, and radio frequencies. At the physical layer, one might find “physical” resources such as network hubs, cabling, repeaters, network adapters or modems.

- **Data Link Layer**

  At the data link layer, directly connected nodes are used to perform node-to-node data transfer where data is packaged into frames. The data link layer also corrects errors that may have occurred at the physical layer.

  The data link layer encompasses two sub-layers of its own. The first, media access control (MAC), provides flow control and multiplexing for device transmissions over a network. The second, the logical link control (LLC), provides flow and error control over the physical medium as well as identifies line protocols.

- **Network Layer**

  The network layer is responsible for receiving frames from the data link layer, and delivering them to their intended destinations among based on the addresses contained inside the frame. The network layer finds the destination by using logical addresses, such as IP (internet protocol). At this layer, routers are a crucial component used to quite literally route information where it needs to go between networks.

- **Transport Layer**

  The transport layer manages the delivery and error checking of data packets. It regulates the size, sequencing, and ultimately the transfer of data between systems and hosts. One of the most common examples of the transport layer is TCP or the Transmission Control Protocol.

- **Session Layer**

  The session layer controls the conversations between different computers. A session or connection between machines is set up,

managed, and termined at layer 5. Session layer services also include authentication and reconnections.

- **Presentation Layer**

  The presentation layer formats or translates data for the application layer based on the syntax or semantics that the application accepts. Because of this, it at times also called the syntax layer. This layer can also handle the encryption and decryption required by the application layer.

- **Application Layer**

  At this layer, both the end user and the application layer interact directly with the software application. This layer sees network services provided to end-user applications such as a web browser or Office 365. The application layer identifies communication partners, resource availability, and synchronizes communication.

As shown in the figure below, on the surface, it appears like the four layers of the TCP/IP model perfectly match the seven layers of the OSI model, but this is not the case. The application layer of the TCP/IP model corresponds to the OSI model's first three layers: application, session, and presentation. The TCP's transport layer corresponds to the OSI model's transport layer. The TCP/IP model's internet layer corresponds to the OSI model's network layer. The OSI model's last two levels correspond to the TCP/IP model's network layer. In comparison to the OSI paradigm, TCP/IP is the most extensively utilized model for delivering communication between computers over the internet.

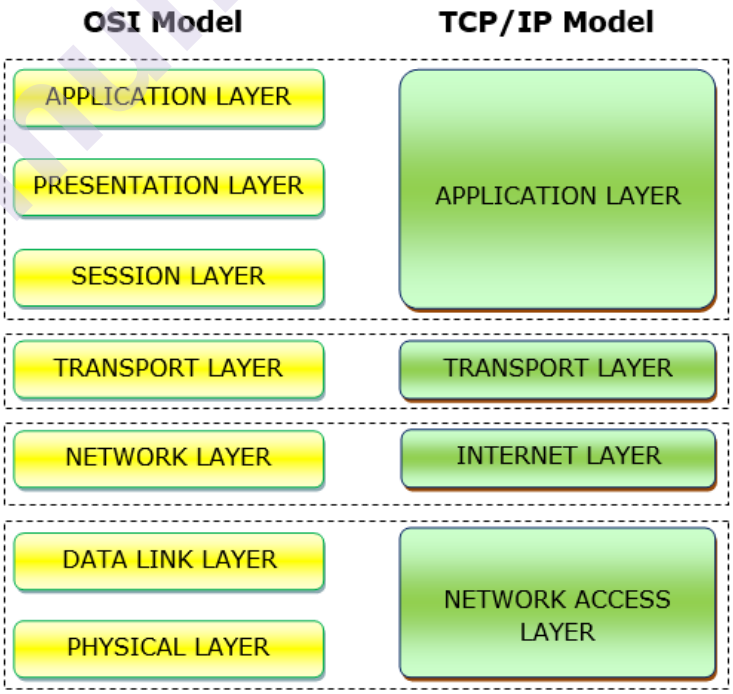


Fig 3.1 OSI model and TCP/IP model

Similarities between TCP/IP and OSI model:

- The layered architecture is used in both reference models.
- The models' layers are compared to one another. The OSI model's physical layer and data link layer correlate to the TCP/IP model's link layer. In both models, the network layers and transport layers are the same. The application layer of the TCP/IP model is made up of the session layer, presentation layer, and application layer of the OSI model.
- Protocols are defined on a layer-by-layer basis in both paradigms.
- Data is separated into packets in both models, and each packet can take its own path from the source to the destination.

Differences between TCP/IP and OSI model:

- TCP/IP has four layers, but the OSI has seven.
- The OSI model is a generic model based on the functions of each layer. TCP/IP model is a protocol-oriented standard.
- The OSI model separates three concepts: services, interfaces, and protocols. There is no apparent differentiation between these three in TCP/IP.
- The OSI model specifies how communication should be carried out, whereas TCP/IP protocols provide the standards upon which the Internet was built. TCP/IP is therefore a more practical model.
- In OSI, the model was created first, followed by the protocols for each layer. The protocols were created first, followed by the model in the TCP/IP suite.

3.2.1.2 Packets

The lowest unit of data that networks want to deal with is **packets**, which are at the bottom of the layering structure. Packets contain the data we wish to send between our systems, as well as some control information that helps networking equipment figure out where the packet should go.

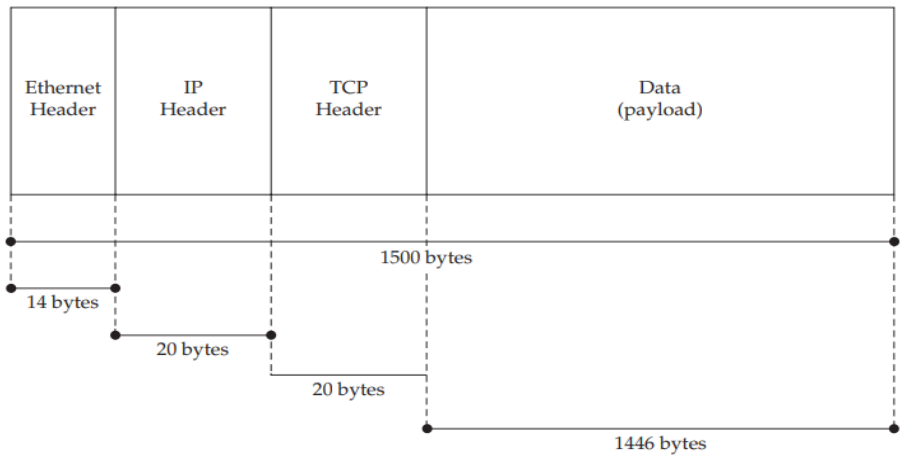


Fig 3.2 A TCP/IP packet

As shown in the fig. 3.2, packets are layered based on protocol, with the lowest layers being applied first. A header is used by each protocol to describe the information required to transfer data from one host to the next. TCP, IP, and Ethernet headers, in their simplest and most usual combined form, take up only 54 bytes of the packet. The remaining 1446 bytes of the packet are dedicated to data.

The relevant headers are removed when each tier examines the message. The Ethernet headers will be stripped, the IP headers will be stripped, and the TCP headers will be stripped in the event of a TCP/IP packet over Ethernet. Only the data that needs to be given to the relevant application will be left.

3.2.2 Ethernet header

The IEEE 802.3 standard defines the basic frame format, **Ethernet**, which is necessary for all MAC implementations. Several extra formats are used to expand the protocol's fundamental functionality.

The destination address, source address, and packet protocol type are all listed in the Ethernet header. Ethernet addresses, also known as Media Access Control (MAC) addresses, are 48-bit (6-byte) integers that identify any Ethernet device on the planet. Although changing an interface's MAC address is possible, it is not encouraged because the default MAC address is guaranteed to be unique, and all MAC addresses on a LAN segment should be unique.

Preamble and SFD, both at the physical layer, begin an Ethernet frame. The Ethernet header contains both the Source and Destination MAC addresses, followed by the packet payload. The CRC field, which is used to identify errors, is the final field. Let's look at each field of the basic frame format now.

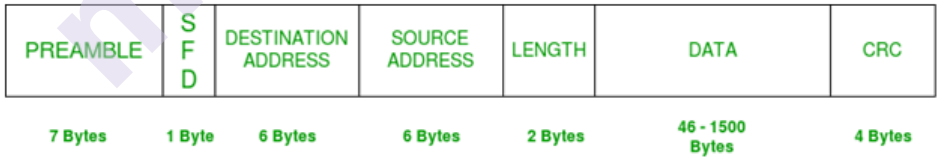


Fig 3.3 Ethernet (IEEE 802.3) Frame Format

- PREAMBLE (7 Bytes) – Ethernet frame starts with Preamble (PRE). This is a pattern of alternative 0's and 1's which indicates starting of the frame and allow sender and receiver to establish bit synchronization. Initially, PRE was introduced to allow for the loss of a few bits due to signal delays. But today's high-speed Ethernet don't need Preamble to protect the frame bits. PRE indicates the receiver that frame is coming and allow the receiver to lock onto the data stream before the actual frame begins.

- Start of frame delimiter (SFD) (1 Byte) – This field is always set to 10101011. SFD indicates that upcoming bits are starting of the frame,

which is the destination address. Sometimes SFD is considered the part of PRE, this is the reason Preamble is described as 8 Bytes in many places. The SFD warns station or stations that this is the last chance for synchronization.

- Destination Address (6 Bytes) – This field contains the MAC address of machine for which data is destined.
- Source Address (6 Bytes) – This field contains the MAC address of source machine. As Source Address is always an individual address (Unicast), the least significant bit of first byte is always 0.
- Length (2 Bytes) – Length field indicates the length of entire Ethernet frame. This field can hold the length value between 0 to 65534, but length cannot be larger than 1500 because of some own limitations of Ethernet.
- Data – This is the place where actual data is inserted, also known as Payload. Both IP header and data will be inserted here if Internet Protocol is used over Ethernet. The maximum data present may be as long as 1500 Bytes. In case data length is less than minimum length i.e. 46 bytes, then padding 0's is added to meet the minimum possible length.
- Cyclic Redundancy Check (CRC) (4 Bytes) – This field contains a 32-bits hash code of data, which is generated over the Destination Address, Source Address, Length, and Data field. If the checksum computed by destination is not the same as sent checksum value, data received is corrupted.

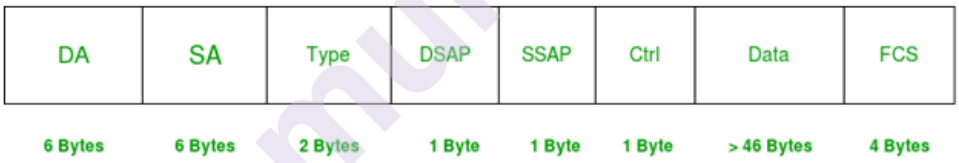


Fig 3.4 Ethernet II Frame

In 802.3 packets, the destination and source MAC addresses remain in place; however, the next 2 bytes represent the length of the packet. Any ethernet header where the protocol type field value is less than 1500, is really an 802.3 packet.

3.2.3 Internet Protocol (IPv4 and IPv6)

For packet-switched networks, **IPv4**, a connectionless protocol is used. It uses a best-effort delivery paradigm, which means that neither delivery nor appropriate sequencing or avoidance of duplicate distribution are guaranteed. IPv4 (Internet Technology Version 4) is the fourth iteration of the Internet Protocol and a widely used data transfer protocol over various networks. In packet-switched layer networks, such as Ethernet, IPv4 is a

connectionless protocol. It establishes a logical connection between network devices by allowing each device to be identified.

An IPv4 address is a 32-bit address that is usually represented in dotted decimal notation, with a decimal value representing each of the four octets (bytes) that make up the address.

For Ethernet communication, IPv4 uses five classes of 32-bit addresses: A, B, C, D, and E. The bit length for addressing the network host differs between Classes A, B, and C. Class D addresses are for military use only, while class E addresses are for future usage only.

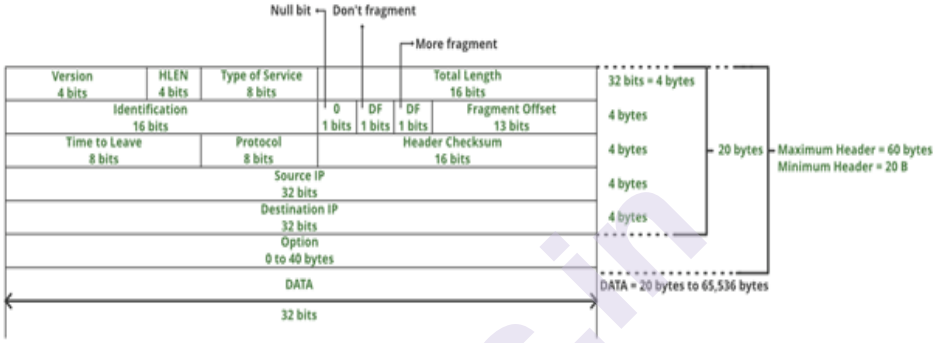


Fig 3.5 IPv4 Datagram Header

The size of IPv4 header is 20-60 bytes. Different fields in IPv4 header are:

VERSION (4 bits) - This field indicates version of the IP protocol, which is 4 for IPv4.

HLEN (IP header length) (4 bits) – This is the number of 32-bit words in the header. The minimum value for this field is 5 and the maximum is 15.

Type of service (8 bits) - Low Delay, High Throughput, Reliability.

Total Length (Length of header + Data) (16 bits) – This field has a minimum value 20 bytes and the maximum is 65,535 bytes.

Identification (16 bits) – This is a Unique Packet Id for identifying the group of fragments of a single IP datagram.

Flags – This field contains 3 flags of 1 bit each : reserved bit (must be zero), do not fragment flag, more fragments flag (same order).

Fragment Offset - Represents the number of Data Bytes ahead of the particular fragment in the particular Datagram. Specified in terms of number of 8 bytes, which has the maximum value of 65,528 bytes.

Time to live (8 bits) - Indicates Datagram's lifetime. It prevents the datagram to loop through the network by restricting the number of Hops taken by a Packet before delivering to the Destination.

Protocol (8 bits) - Name of the protocol to which the data is to be passed.

Header Checksum (16 bits) – This field is used for checking errors in the datagram header.

IP version 6 is a new version of the Internet Protocol that is far more sophisticated and efficient than IP version 4. Let's have a look at the IPv6 header and see how it differs from the IPv4 header.

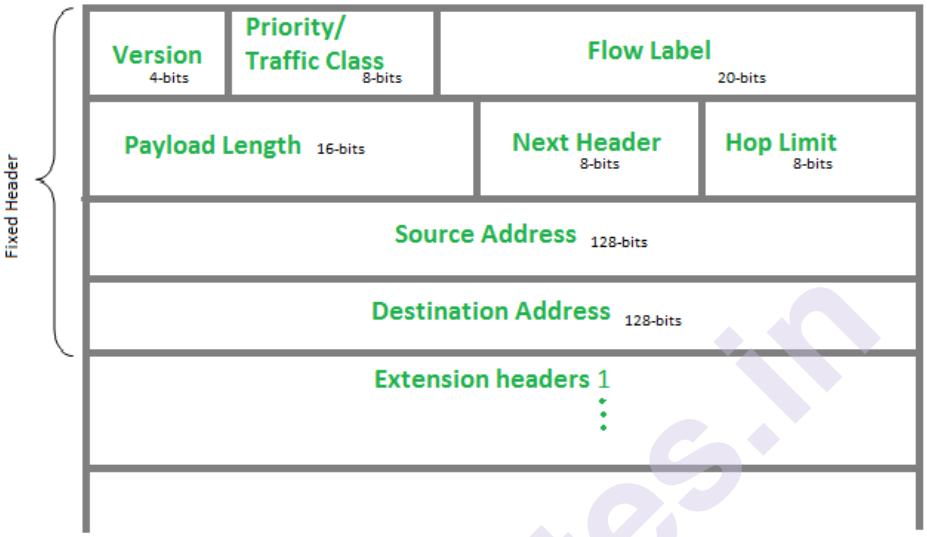


Fig 3.6 IPv6 Header

Version (4-bits) - Indicates version of Internet Protocol which contains bit sequence 0110.

Traffic Class (8-bits) - The Traffic Class field indicates class or priority of IPv6 packet which is similar to Service Field in IPv4 packet. It helps routers to handle the traffic based on priority of the packet. If congestion occurs on router then packets with least priority will be discarded.

Flow Label (20-bits) - Flow Label field is used by source to label the packets belonging to the same flow in order to request special handling by intermediate IPv6 routers, such as non-default quality of service or real time service. In order to distinguish the flow, intermediate router can use source address, destination address and flow label of the packets. Between a source and destination multiple flows may exist because many processes might be running at the same time. Routers or Host that do not support the functionality of flow label field and for default router handling, flow label field is set to 0. While setting up the flow label, source is also supposed to specify the lifetime of flow.

Payload Length (16-bits) - It indicates total size of the payload which tells routers about amount of information a particular packet contains in its payload. Payload Length field includes extension headers(if any) and upper layer packet. In case length of payload is greater than 65,535 bytes (payload up to 65,535 bytes can be indicated with 16-bits), then the payload length

field will be set to 0 and jumbo payload option is used in the Hop-by-Hop options extension header.

Next Header (8-bits) - Next Header indicates type of extension header(if present) immediately following the IPv6 header. Whereas In some cases it indicates the protocols contained within upper-layer packet, such as TCP, UDP.

Hop Limit (8-bits) - Hop Limit field is same as TTL in IPv4 packets. It indicates the maximum number of intermediate nodes IPv6 packet is allowed to travel. Its value gets decremented by one, by each node that forwards the packet and packet is discarded if value decrements to 0. This is used to discard the packets that are stuck in infinite loop because of some routing error.

Source Address (128-bits) - Source Address is 128-bit IPv6 address of the original source of the packet.

Destination Address (128-bits) - Destination Address field indicates the IPv6 address of the final destination. All the intermediate nodes can use this information in order to correctly route the packet.

Extension Headers - In order to rectify the limitations of IPv4 Option Field, Extension Headers are introduced in IPv6. The extension header mechanism is very important part of the IPv6 architecture. Next Header field of IPv6 fixed header points to the first Extension Header and this first extension header points to the second extension header and so on.

3.2.4 TCP (Transmission Control Protocol)

TCP is a Transport layer protocol which provides acknowledgement of the received packets and is also reliable as it resends the lost packets. It is better than UDP but due to these features it has an additional overhead. It is used by application protocols like HTTP and FTP.

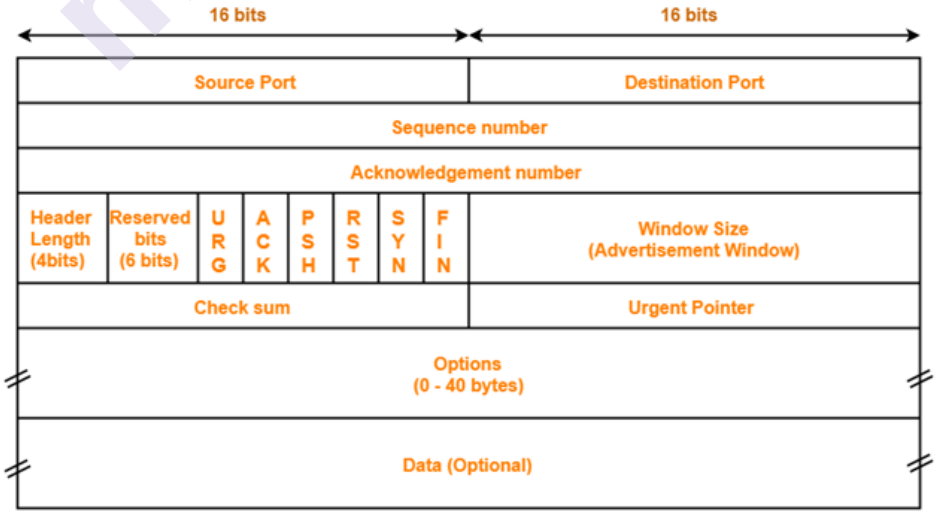


Fig 3.7 TCP Header

1. Source Port- Source Port is a 16-bit field. It identifies the port of the sending application.
2. Destination Port- Destination Port is a 16-bit field. It identifies the port of the receiving application.
3. Sequence Number- Sequence number is a 32-bit field. TCP assigns a unique sequence number to each byte of data contained in the TCP segment. This field contains the sequence number of the first data byte.
4. Acknowledgement Number- Acknowledgment number is a 32-bit field. It contains sequence number of the data byte that receiver expects to receive next from the sender. It is always sequence number of the last received data byte incremented by 1.
5. Header Length- Header length is a 4-bit field. It contains the length of TCP header. It helps in knowing from where the actual data begins.
6. Reserved Bits- The 6 bits are reserved. These bits are not used.
7. URG Bit- URG bit is used to treat certain data on an urgent basis.

   When URG bit is set to 1, it indicates the receiver that certain amount of data within the current segment is urgent. Urgent data is pointed out by evaluating the urgent pointer field. The urgent data has be prioritized. Receiver forwards urgent data to the receiving application on a separate channel.
8. ACK Bit- ACK bit indicates whether acknowledgement number field is valid or not. When ACK bit is set to 1, it indicates that acknowledgement number contained in the TCP header is valid. For all TCP segments except request segment, ACK bit is set to 1. Request segment is sent for connection establishment during Three Way Handshake.
9. PSH Bit- PSH bit is used to push the entire buffer immediately to the receiving application. When PSH bit is set to 1, all the segments in the buffer are immediately pushed to the receiving application. No wait is done for filling the entire buffer. This makes the entire buffer to free up immediately.
10. RST Bit- RST bit is used to reset the TCP connection. When RST bit is set to 1, It indicates the receiver to terminate the connection immediately. It causes both the sides to release the connection and all its resources abnormally. The transfer of data ceases in both the directions. It may result in the loss of data that is in transit.

    This is used only when-

    - There are unrecoverable errors.
    - There is no chance of terminating the TCP connection normally.
11. SYN Bit- SYN bit is used to synchronize the sequence numbers.

    When SYN bit is set to 1,

It indicates the receiver that the sequence number contained in the TCP header is the initial sequence number. Request segment sent for connection establishment during Three way handshake contains SYN bit set to 1.

12. FIN Bit- FIN bit is used to terminate the TCP connection. When FIN bit is set to 1, It indicates the receiver that the sender wants to terminate the connection. FIN segment sent for TCP Connection Termination contains FIN bit set to 1.

13. Window Size- Window size is a 16-bit field. It contains the size of the receiving window of the sender. It advertises how much data (in bytes) the sender can receive without acknowledgement. Thus, window size is used for Flow Control.

14. Checksum-Checksum is a 16-bit field used for error control. It verifies the integrity of data in the TCP payload. Sender adds CRC checksum to the checksum field before sending the data.

Receiver rejects the data that fails the CRC check.

15. Urgent Pointer- Urgent pointer is a 16-bit field. It indicates how much data in the current segment counting from the first data byte is urgent. Urgent pointer added to the sequence number indicates the end of urgent data byte. This field is considered valid and evaluated only if the URG bit is set to 1.

16. Options- Options field is used for several purposes. The size of options field vary from 0 bytes to 40 bytes. Options field is generally used for -Time stamp,Window size extension, Parameter negotiation, Padding.

3.2.5 UDP (User Datagram Protocol)

UDP is a part of the Internet Protocol suite, referred to as UDP/IP suite. UDP is a connectionless Transport Layer protocol. Unlike TCP, it is an unreliable protocol. So, there is no need to establish a connection prior to data transfer.

Despite the fact that TCP is the most widely used transport layer protocol for most Internet services, it provides guaranteed delivery, reliability, and much more, all of these benefits come at a cost in terms of overhead and delay. UDP enters the scene at this point. UDP is required for real-time applications such as computer gaming, phone or video communication, and live conferencing. Because great throughput is required, UDP allows packets to be dropped rather than processed. Because UDP does not do error checking, it saves bandwidth.

In terms of latency and bandwidth, the User Datagram Protocol (UDP) is more efficient.

The UDP header is an 8-byte fixed and simple header, whereas the TCP header can range from 20 to 60 bytes. The first 8 bytes contain the necessary

header information, while the remainder is data. Because each UDP port number field is 16 bits long, the range for port numbers is 0 to 65535; port number 0 is reserved. Port numbers are used to differentiate between different user requests or processes.

The following are the fields in UDP header:

- Source Port (2 Bytes) : Source Port field is used to identify port number of source.
- Destination Port (2 Bytes) : It is used to identify the port of destined packet.
- Length (16 bits) : This field indicates the length of UDP including header and the data.
- Checksum (2 Bytes) : Checksum is the 16-bit one's complement of the one's complement sum of the UDP header, pseudo header of information from the IP header and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

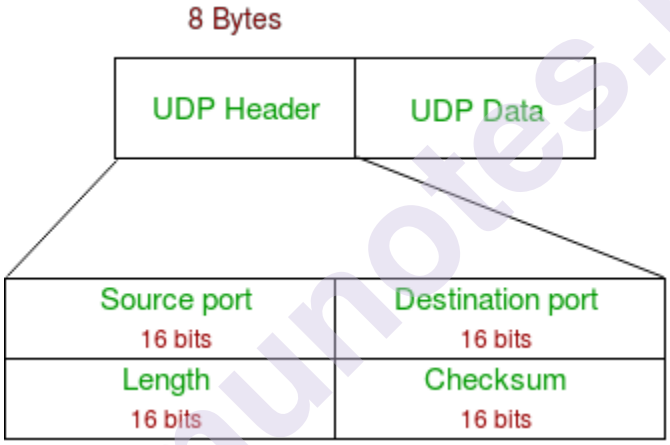


Fig 3.7 UDP Header

3.2.6 TCP Connection

3.2.6.1 Connection establishment

TCP hosts must establish a connection-oriented session with one another in order for transport services to be reliable. The three-way handshake procedure is used to establish the connection. By allowing both ends of a network to agree on original sequence numbers, a **three-way handshake** synchronizes both ends of the network. The process for three-way handshake is shown in the fig 3.8.

This approach also ensures that both parties are prepared to send data and are aware that the other is open to communicate. This is necessary to prevent packets from being shared or retransmitted during session setup or termination. Each host chooses a sequence number at random to keep track of bytes in the stream it is sending and receiving.

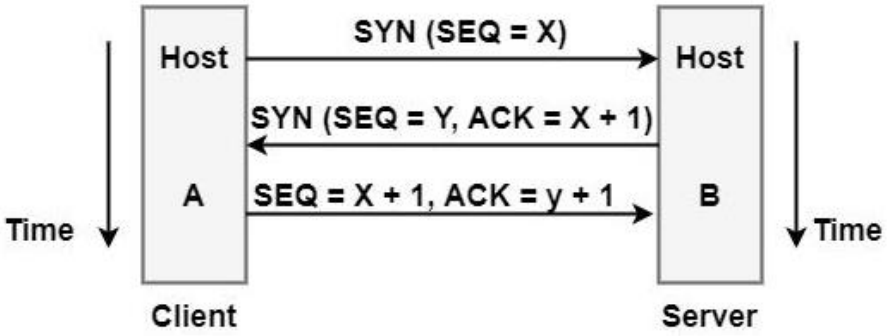


Fig 3.8 Three way Handshake

As shown in figure, the requesting end (Host A) sends a SYN segment with the server's port number and beginning sequence number for the client to connect to (x). Host B acknowledges its own SYN segment, which includes the server's beginning sequence number (y). The server additionally accepts the sender's SYN plus one (X + 1) in response to the client SYN. One sequence number is consumed by a SYN. Accepting the server's SEQ plus one (SEQ = x + 1, ACK = y + 1), the client should acknowledge this SYN from the server. A TCP connection is established in this manner.

3.2.6.2 Connection release

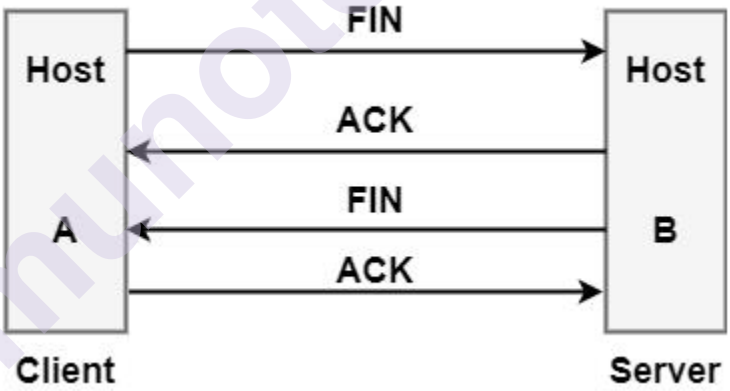


Fig 3.9 Connection release

While three segments are required to establish a connection, four segments are required to terminate one. Each direction of a TCP connection should be shut down independently while it is full-duplex i.e. data flows in each direction independently of the other direction.

The fig 3.9 depicts the termination procedure for each host. When one end has done sending data, the other end might share a FIN. When a TCP receives a FIN, it should inform the application that the other end of the data flow direction has been terminated. The application issuing a close usually results in the sending of a FIN.

The receipt of a FIN just indicates that data will no longer flow in that direction. After getting a FIN, a TCP can send data. The active close is executed by the end that sends the first FIN (for example, transmit the first FIN). The passive end is managed by the opposite end (which receives this FIN).

3.2.7 ARP

A basic message format is used by the address resolution protocol (ARP), which contains either an address resolution request or an address resolution answer. The size of an ARP message is determined by the link layer and network layer address sizes. The network type and address size used at each layer are described in the message header. The operation code, which is 1 for the request and 2 for the response, completes the message header.

The payload of the packet has following four addresses:

- Hardware address of the sender hosts
- Hardware address of the receiver hosts
- Protocol address of the sender hosts
- Protocol address of the receiver hosts

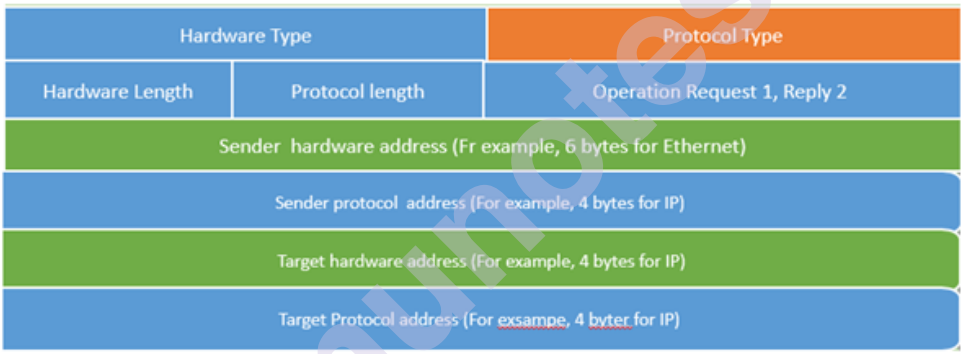


Fig 3.10 ARP header

As shown in fig 3.10, following fields exists in ARP header:

- Hardware Type – It is 1 for Ethernet.
- Protocol Type – It is a protocol used in the network layer.
- Hardware Address Length – It is the length in bytes so that it would be 6 for Ethernet.
- Protocol Address Length – Its value is 4 bytes.
- Operation Code - indicates that the packet is an ARP Request (1) or an ARP Response (2).
- Senders Hardware Address – It is a hardware address of the source node.
- Senders Protocol Address - It is a layer 3 address of the source node.

- Target Hardware Address – It is used in a RARP request, which response impact both the destination's hardware and layer 3 addresses.
- Target Protocol Address – It is used in an ARP request when the response carries both layer 3 addresses and the destination's hardware.

3.3 Basic Network Configuration

Like most modern operating systems, Linux distributions include a comprehensive collection of graphical tools for managing the majority of the system's networking-related operations. Understanding how Linux distributions handle network configuration is crucial and can be useful in a variety of situations. Now, we will tackle an overview of network interface drivers, the tools

necessary for performing command-line administration of your network interface(s).

3.3.1 Module and Network Interfaces

Under Linux, network devices violate the convention of accessing all devices through the file abstraction layer. There is no mechanism for anything to access the interface until the network driver initializes it and registers itself with the kernel. Ethernet devices are typically identified by names like ethX, emX, ensX, or pXpX, where X is the device or index number.

The device drivers for your network interface cards (NICs) may have been compiled as a module, depending on how your kernel was compiled. This is the default approach in most distributions because it makes it much easier to probe for new hardware.

If you have auto-loading modules enabled and the driver is setup as a module, you may need to inform the kernel the mapping between device names and the module to load, or you may just need to supply some extra options to the module. Interface cards (NICs) may have been compiled as a module, and this can be done by using (or creating) an appropriate configuration file inside the /etc/modprobe.d directory. This can be accomplished by modifying (or creating) a configuration file in the /etc/modprobe.d directory.

3.3.2 Network Device Configuration Utilities

To wrap around the different command-line tools that we cover in the next sections, a variety of graphical and menu-driven utilities have been built. Many of these wrapper GUI programs come pre-installed on Linux distributions, particularly those that run in full desktop graphical mode.

As an administrator, you should become acquainted with and be able to setup network devices directly using any of the underlying command-line programs. Knowing how to accomplish this is quite useful because the

command-line interface exposes many more choices not available in GUIs (CLI).

Let’s have a look at some command-line utilities to manipulate network settings.

3.3.2.1 Ip

The ip utility is a strong tool for managing network devices in Linux. The ip utility is included in the software packages iproute and iproute2. The package includes networking utilities (such as ip) that are designed to take advantage of and control the Linux kernel's advanced networking capabilities. The ip utility's syntax is a little more terse and tolerant than the ifconfig utility's. The ip command, on the other hand, is far more powerful.

3.3.2.2 Ipconfig

The ifconfig application is one of various options for configuring network interface cards. Because its native format lacks menus or a graphical interface, all of its operations can be carried out via command-line arguments. Despite its popularity, several Linux distributions are gradually deprecating the usage of ifconfig, and users are advised to instead utilise the utilities that come with the iproute package (for example, ip).

Microsoft developed some CLI networking utilities that replicate functional portions of their UNIX counterparts, and administrators who have used the Windows ipconfig application may notice some similarities.

3.3.2.3 nmcli

The NetworkManager package's suite of utilities is the newest kid on the block when it comes to controlling network devices and connections in the Linux environment. The command-line nmcli utility, for example, can be used to operate the NetworkManager daemon. The concept of connections is a unique concept in the NetworkManager environment (or profiles). A connection, also known as a profile, is a set of connected data that explains how to set up or connect to a network. All network settings (MAC addresses, IP addresses, DNS, gateways, and so on) is saved as a connection in NetworkManager. A real (or virtual) network device can configure itself using the information in a connection assigned to it. To manage varied contexts, you can define as many connections as you desire.

3.3.2 Managing Routes

You may need a router or gateway device to communicate with other hosts if your host is connected to a network with several subnets. This device lies in the middle of networks, redirecting packets to their intended destination. When a host is unsure where to send a packet, it falls back on its default route. This channel leads to a router that, in theory, understands where the packet should go or, at the very least, knows of another router that can make better decisions.

A typical single-homed Linux host is familiar with a number of basic routes. The loopback route, which merely points to the loopback device, is one of the conventional routes. Another is the route to the local area network (LAN), which allows packets to be forwarded directly to hosts within the same LAN. The default path is yet another standard route. This route is used to send packets to networks outside of the LAN. The link-local route is another route that you can encounter in a standard Linux routing table (169.254.0.0).

3.3.2.1 Route configuration

You may need to alter your routes manually in some cases. When many network interfaces are accessible on the same host, and each NIC is connected to a distinct network, this is usually required (multi-homed). You should be able to add a route so that packets for a specific destination address are sent to the correct network.

You can connect with subnets and other networks using the route command, and you can even prohibit traffic between networks or devices by altering the routing table.

Because the net-tools package has been superseded by iproute2 and isn't included by default in all Linux editions, if you don't have it on Debian or derived systems like Ubuntu or Mint, type the following command:

 # apt install net-tools

Once installed you'll be able to use route, ifconfig, iwconfig and other commands included in the package.

To print the routing table on Linux, you can type:

 # sudo route

you will see the default gateway, that is the router, the first hop through which the traffic passes before going to the next hop or final node.

You can create and delete routes using ip command too.

3.3.2.2 Displaying routes

Using route command is one of the easiest ways to display your routing table. Just run route

without any parameters. Here is a complete run, along with the output:

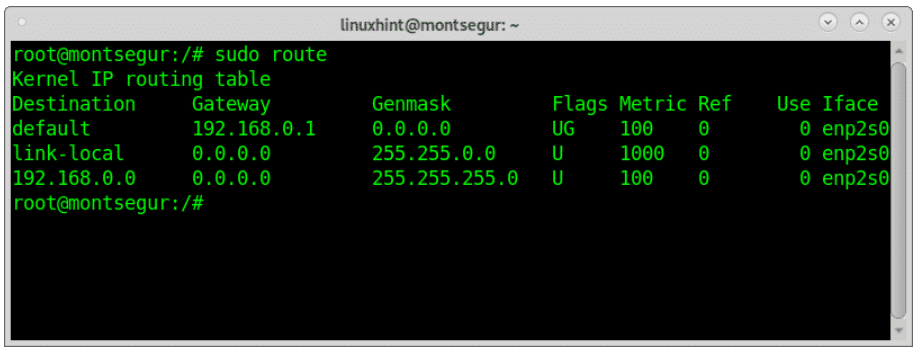


Fig 3.11 Output of route command

It's worth noting that route presented hostnames for all IP addresses it could find and resolve. While this is pleasant to read, it becomes problematic when network failures occur and DNS or Network Information Service (NIS) servers become inaccessible. The route command will hang while attempting to resolve hostnames and waiting for the servers to respond. This will continue for several minutes, or until the request is completed.

To get around this, use the -n option with route to display the same information, but route will not attempt to resolve the IP addresses' hostnames.

 # sudo route -n

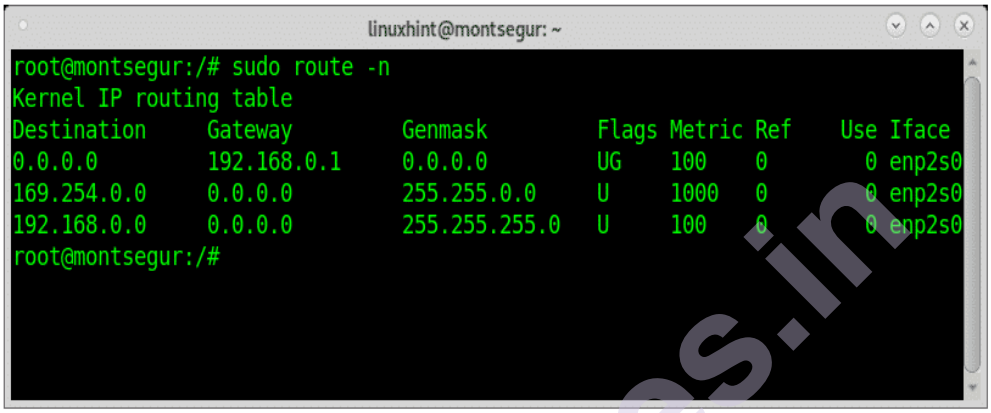


Fig 3.12 Output of route command with -n option

The **network statistics(netstat)** tool is typically used to display the status of all network connections on a computer. It can, however, display the kernel routing table if you use the -r option. This technique of examining routes is supported by most other UNIX-based operating systems. To list all listening ports, using both TCP and UDP, use netstat -a.

Advanced IP routing and network device configuration features are provided by the iproute package. On a Linux computer, the ip command can also be used to alter the routing table. The route object is used in conjunction with the ip command to accomplish this.

A Linux-based system, like most commercial carrier-grade routing devices, can maintain and use many routing tables at the same time. The route command you just saw was actually only displaying and managing one of the system's default routing tables—the main table.

To view the contents of table main, you need type:

ip route show table main

To view the contents of all the routing tables on the system, type the following command:

ip route show table all

3.3.3 A simple Linux Router

Linux offers a lot of networking capabilities, including the ability to operate as a full-fledged router. A typical PC running Linux with a few network adapters can suffice in situations when a powerful, low-cost router is required. A Linux router may realistically move thousands of data packets per second, depending on the speed of the PC, the CPU cache, the type of NIC, PCI or other interfaces, and the system bus speeds. In fact, there are several commercial routers on the market that run a stripped-down and optimized Linux kernel with a good GUI administrative front-end.

3.3.3.1 Routing with static routes

By using the ip command, you can set up and view static route. For example, to display the current routing table you can type the command:

 # ip route show

Output (Sample):

```
192.168.2.0/24 dev eth1 proto kernel  scope link  src 192.168.2.1
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.2
default via 192.168.1.254 dev eth0
```

You can add static route using following command:

ip route add {NETWORK} via {IP} dev {DEVICE}

For example, network 192.168.55.0/24 available via 192.168.1.254:

 # ip route add 192.168.55.0/24 via 192.168.1.254 dev eth1

Alternatively, you can use old good route command too:

 # route add -net 192.168.55.0 netmask 255.255.255.0 gw 192.168.1.254 dev eth1

3.3.3.2 How Linux chooses an IP address

Correct communication between hosts with various IP addresses depends on selecting the correct source address. If a host communicates with a public Internet host using a private network address, the return portion of the message is likely to never arrive.

An outbound packet's first source address is determined using the following set of rules. The application can request a specific IP address, in which case the kernel will use the src hint from the selected route path, or, if this hint is missing, the kernel will choose the first address configured on the interface that is in the same network as the destination address or the nexthop router.

3.3.3.3 Hostname configuration

On a network, a system's hostname is the friendly name by which other systems or applications can address it. As a result, configuring a system's hostname is a crucial network configuration activity. During the OS installation, you would have been prompted to create or select a hostname for your machine. It's possible that a hostname was allocated to your machine automatically if you didn't provide one during the OS installation.

Running the command hostname, without any parameters, will return the current hostname of your Linux system.

$ hostname

If you want to change or set hostname of your Linux system, you need to type:

$ hostname NEW_HOSTNAME

Of course, you will need to replace “NEW_HOSTNAME” with the actual hostname that you wish to set. This will change the hostname of your system immediately, but the original hostname will be restored upon next reboot.

Recent Linux distros (such as Fedora, RHEL) that have fully incorporated systemd as the service manager now support the notion of a single host having multiple hostnames for different purposes. The different supported hostname classes are:

- Static hostname: This is the same as the traditional hostname used to initialize the kernel at boot. It is stored in the /etc/hostname file.
- Transient hostname: This is a dynamic type of hostname that is subject to change during the system runtime. It can be changed, for example, via DHCP. It initially starts off being set to the same value as the static hostname.
- Pretty hostname: This is a pretty, free-form version of the hostname. It can support the inclusion of special characters that the other hostname forms do not support.

3.4 LINUX FIREWALL (NETFILTER)

The Linux firewall and packet filtering/mangling system has come a long way, from an initial implementation based on Berkeley Software Distribution (BSD), through several major rewrites/updates (kernels 2.0, 2.2, 2.4, 2.6, 3.0, and 4.0) and four user-level interfaces (ipfwadm, ipchains, iptables, and nftables), and four user-level interfaces. Netfilter refers to the present Linux packet filter and firewall infrastructure (including kernel and user tools).

3.4.1 Working

he netfilter project is a community-driven collaborative FOSS project that provides packet filtering software for the Linux 2.4.x and later kernel series.

The netfilter project is commonly associated with iptables and its successor nftables. The netfilter project enables packet filtering, network address [and port] translation (NA[P]T), packet logging, userspace packet queueing and other packet mangling.

The netfilter hooks are a framework inside the Linux kernel that allows kernel modules to register callback functions at different locations of the Linux network stack. The registered callback function is then called back for every packet that traverses the respective hook within the Linux network stack.

iptables is a generic firewalling software that allows you to define rulesets. Each rule within an IP table consists of a number of classifiers (iptables matches) and one connected action (iptables target).

nftables is the successor of iptables, it allows for much more flexible, scalable and performance packet classification. This is where all the fancy new features are developed.

Main Features

- stateless packet filtering (IPv4 and IPv6).
- stateful packet filtering (IPv4 and IPv6).
- all kinds of network address and port translation, e.g. NAT/NAPT (IPv4 and IPv6).
- flexible and extensible infrastructure.
- multiple layers of API's for 3rd party extensions.

With Netfilter, we can

- build internet firewalls based on stateless and stateful packet filtering.
- deploy highly available stateless and stateful firewall clusters.
- use NAT and masquerading for sharing internet access if you don't have enough public IP addresses.
- use NAT to implement transparent proxies.
- aid the tc and iproute2 systems used to build sophisticated QoS and policy routers.
- do further packet manipulation (mangling) like altering the TOS/DSCP/ECN bits of the IP header.

3.4.2 Installing and configuring iptables

Netfilter, a packet filtering program included into the Linux kernel, is a built-in feature. The iptables command is used to define the rules for what happens to packets depending on their header IP addresses and network connection type. When you don't have a dedicated firewall between your Linux system and the Internet, such as when you use a DSL or cable modem to access to the Internet, the built-in packet filtering capabilities comes in

handy. In essence, a packet filtering firewall can be installed between the kernel and the programs on your Linux system.

Most Linux distributions, such as Fedora and SUSE, now include GUI tools to turn on a packet filtering firewall and simplify the configuration experience for the user.

In some distributions, you need to install ufw (an acronym for Uncomplicated Firewall), which lets you manage a net-filter firewall and simplify configuration. ufw serves as a front end to iptables, which allows you to enter commands in a terminal window directly through it. The following command turns the firewall on:

sudo ufw enable

The command below displays the information as the following:

sudo ufw status verbose

```
Status: active

Logging: on (low)

Default: deny (incoming), allow (outgoing), disabled (routed)

New profiles: skip
```

The default settings are exactly what you're looking for in most cases for a client machine: allowing outgoing traffic and denying incoming traffic.

You can allow incoming packets meant for specific Internet services such as SSH, Telnet, and FTP. If you select a network interface such as eth0 (the first Ethernet card) as trusted, all network traffic over that interface is allowed without any filtering.

Using the **iptables** command is somewhat complex. The command uses the concept of a chain, which is a sequence of rules. Each rule says what to do with a packet if the header contains certain information, such as the source or destination IP address. If a rule doesn't apply, iptables consults the next rule in the chain. By default, there are three chains:

- INPUT chain: Contains the first set of rules against which packets are tested. The packets continue to the next chain only if the INPUT chain doesn't specify DROP or REJECT.
- FORWARD chain: Contains the rules that apply to packets attempting to pass through this system to another system (when you use your Linux system as a router between your LAN and the Internet, for example).
- OUTPUT chain: Includes the rules applied to packets before they're sent out (either to another network or to an application).

When an incoming packet arrives, the kernel uses iptables to make a routing decision based on the destination IP address of the packet. If the packet is for this server, the kernel passes the packet to the INPUT chain. If the packet

satisfies all the rules in the INPUT chain, the packet is processed by local processes such as an Internet server that's listening for packets of this type.

If the kernel has IP forwarding enabled, and the packet has a destination IP address of a different network, the kernel passes the packet to the FORWARD chain. If the packet satisfies the rules in the FORWARD chain, it's sent out to the other network. If the kernel doesn't have IP forwarding enabled, and the packet's destination address isn't for this server, the packet is dropped.

If the local processing programs that receive the input packets want to send network packets out, those packets pass through the OUTPUT chain. If the OUTPUT chain accepts those packets, they're sent out to the specified destination network.

You can view the current chains, add rules to the existing chains, or create new chains of rules by using the iptables command, which normally requires you to be root to interact with. When you view the current chains, you can save them to a file. If you've configured nothing else, and your system has no firewall configured, typing iptables -L should show the following:

```
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
```

In this case, all three chains — INPUT, FORWARD, and OUTPUT — show the same ACCEPT policy, which means that everything is wide open.

If you're setting up a packet filter, the first thing you do is specify the packets that you want to accept. To accept packets from the 192.168.0.0 network address, add the following rule to the INPUT chain:

iptables -A INPUT -s 192.168.0.0/24 -j ACCEPT

Now add a rule to drop everything except local loopback (the lo network interface) traffic and stop all forwarding with the following commands:

iptables -A INPUT -i ! lo -j REJECT

iptables -A FORWARD -j REJECT

The first iptables command, for example, appends to the INPUT chain (-A INPUT) the rule that if the packet doesn't come from the lo interface (-i ! lo), iptables rejects the packet (-j REJECT).

Before rejecting all other packets, you may add more rules to each INPUT chain to allow specific packets in. You can select packets to accept or reject based on many parameters, such as IP addresses, protocol types (TCP, UDP), network interface, and port numbers.

You can do all sorts of specialized packet filtering with iptables. Suppose that you set up a web server and want to accept packets meant for only HTTP (port 80) and SSH services. The SSH service (port 22) is for you to securely log in and administer the server. Also suppose that the server's IP address is 192.168.0.10. Here's how you might set up the rules for this server:

```
iptables -P INPUT DROP

iptables -A INPUT -s 0/0 -d 192.168.0.10 -p tcp --dport 80 -j ACCEPT

iptables -A INPUT -s 0/0 -d 192.168.0.10 -p tcp --dport 22 -j ACCEPT
```

In this case, the first rulesets up the default policy of the INPUT chain to DROP, which means that if none of the specific rules matches, the packet is dropped. The next two rules say that packets addressed to 192.168.0.10 and meant for ports 80 and 22 are accepted.

3.5 SYSTEM AND NETWORK SECURITY

Because the focus here is on network security, this section discusses only those parts of TCP/IP affecting your system's security.

3.5.1 Monitoring your system

There are some commercial-grade monitoring applications available for free and open source that are highly worth looking into. We'll look at a number of useful tools for system monitoring in this article. Some of these utilities are already included in your Linux distribution, while others are not. All of them are free and simple to obtain.

- Use of syslog:The service, rsyslogd, the system logger as well as the systemd-journald service (journald), both of which help manage and collect log messages from various programs.
- Log parsing: Doing periodic checks on the system's log files is an important part of maintaining security.
- Storing Log Entries: Dedicate a single host on your network to storing log entries. Configure your local logging daemon to send all of its messages to a separate/central loghost and configure the central host appropriately to accept logs from trusted or known hosts.

- Monitoring Bandwidth with MRTG: Monitoring the amount of bandwidth being used on your servers produces some useful information.

3.5.2 Handling attacks

An IT security officer must plan for an attack in the same way that a facilities director plans for fires and a backup administrator plans for backing up and restoring data as needed. This section discusses the most important aspects of Linux.

Trust Nothing (and No One)

If an attacker has successfully gained access to your systems, your servers will be unable to provide you with accurate information about the situation. Root kits might make it harder to detect them. After you've replaced the binaries, you could find that there's nothing else you can do to aid the server. Every server that has been successfully hacked could need to be totally rebuilt from the ground up. Before performing the reinstall, you should try to identify how far the attacker got into the backup cycle so that you can determine when the data is certain to be trustworthy.

Change Your Passwords

It is critical to update all of your passwords if the attacker has obtained your root password or a copy of the password file (or equivalent).

Pull the Plug

You'll need to turn off any remote access to the machine once you're ready to start cleaning up. Before rejoining the server to the network, you may need to disable all network traffic to it until it has been entirely rebuilt with the newest updates.

Simply unplug whatever is connecting the box to the network to accomplish this. Putting a server back on the network while it is still receiving patches nearly guarantees that you will be hit by another assault.

3.5.3 Network security tools

Let’s review few tools that we can use for testing our system.

3.5.3.1 Nmap

The nmap application can be used to search for open TCP and UDP ports on a single host or a group of hosts. nmap can go beyond scanning by attempting to connect to remote listening apps or ports in order to better identify the remote application. This is a powerful and simple tool for an administrator to see what the system exposes to the network, and it's routinely used by both attackers and administrators to get a sense of what they can do to a host.

3.5.3.2 Snort

An intrusion detection system (IDS) allows you to covertly monitor a network point and report on suspicious activities based on packet traces. Snort (www.snort.org) is an open source intrusion detection system (IDS) and intrusion prevention system (IPS) with comprehensive rule sets that are regularly updated with new attack vectors. Any suspicious activity can be reported to a logging host, and numerous open source log-processing tools (such as the Basic Analysis and Security Engine, or BASE) are available to assist make sense of the data gathered.

3.5.3.3 Nessus

The Nessus vulnerability scanner (www.tenable.com/products/nessus-vulnerability-scanner) builds on the nmap concept by adding deep application-level probes and a robust reporting system. Running Nessus against a server is a rapid technique to assess the server's vulnerability. y Examine the server's security. Understanding Nessus' output is the key to understanding it. The report will include a variety of remarks, ranging from informational to high-level. Nessus may log false positives or frightening informational comments depending on how your application is written and what other services you provide on your Linux system.

3.5.3.4 Wireshark

Wireshark can be used for performing network security functions. All of the tools discussed in the preceding sections eat raw network traces to obtain insight into what your server is doing. These tools, on the other hand, don't have the same level of understanding of what your server is meant to perform as you do. As a result, being able to take network traces and scan through them to search for any suspicious activity will be beneficial.

3.6 SUMMARY

This chapter covered the fundamentals of TCP/IP and other protocols. We also had an overview of network interface drivers, the tools necessary for performing command-line administration of your network interface(s). We also covered how the ifconfig, ip, nmcli, and route commands can

be used to configure the IP addresses (IPv4 and IPv6) and route entries (IPv4 and IPv6) on Linux-based systems. We discussed the ins and outs of the Linux firewall, Netfilter. In particular, we discussed the usage of the iptables, ip6tables, and nft (nftables) commands. This chapter also covered the basics of network security as it pertains to Linux.

3.7 REFERENCES

1. Linux Administration: A Beginner's Guide, Wale Soyinka, Seventh Edition, McGraw-Hill Education, 2016.
2. Ubuntu Server Guide, Ubuntu Documentation Team, 2016.
3. Mastering Ubuntu Server, Jay LaCroix, PACKT Publisher, 2016.
4. https://access.redhat.com/documentation
5. https://www.linux.com/
6. https://www.netfilter.org/

❄❄❄❄❄❄❄

4

INTERNET SERVICES

Unit Structure :

4.0 OBJECTIVES

After Completion of this chapter, you will be able to understand:

- Domain name services in Linux server.
- Different protocols used in Linux server.
- Network authentication system and security.
- Secure Shell, LDAP and Samba server.

4.1 DOMAIN NAME SYSTEM

The naming system for computers is called *Domain Name System* (DNS). The DNS server is an internet service that maps the domain name into IP address so that human and computer can understand at their ends. This is very important service in the server and backbone of the internet. In Linux, BIND (Berkley Internet Naming Daemon) is used as DNS server on the internet.

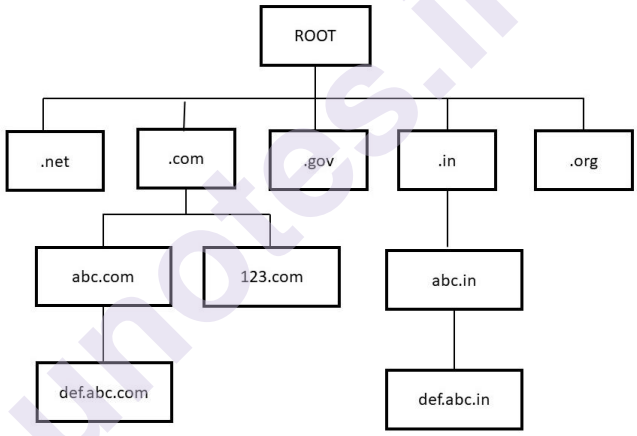


FIGURE 4.1

The naming system of DNS is hierarchical and logical tree structure, which is called *DNS name space.* It has a unique root and can have many numbers of subdomains as shown in figure 4.1. internet namespace root has many top-level domain names. For example .com, .org, .net etc. subdomain is like idol.com and further subdomain is like linux.idol.com.

Naming Standards of DNS

The following characters can be used for DNS

- a-z
- A-Z
- 0-9
- -(hyphen)

4.1.1 Features of domain name System

- Names resolution services provided to the client computers by DNS.
- It allows IP address to hostname mapping.
- There are three modes of DNS servers in Linux: Primary DNS Server, Secondary (Slave) DNS server and caching only DNS server.
- Duplication of DNS database between servers.

4.1.2 Types of DNS Servers

There are three types of DNS Servers as following:

- **Primary DNS Server: -** This server holds information like administrator's details and IP addresses. This contains the master copy of domain's configuration files.
- **Secondary DNS Server: -** Secondary DNS Server is also called Slave DNS Server. These servers take the domain information from primary DNS server, which is read-only. Secondary server is used as a backup when primary server is crash or down.
- **Caching DNS Server: -** This server is used to reduce the workload of primary and secondary DNS servers. This server contains the data that is recently requested by the user.

4.2 FILE TRANSFER PROTOCOL

File Transfer Protocol (FTP) is the commonly used protocol on the internet. The purpose of file transfer protocol is to transfer files between computer hosts over the network. Users can access files on remote system without need the user to log in directly to remote host. Users can access files using some simple commands. It is an application layer protocol. In FTP, the computer of end user is called *local host* and FTP computer (Server) is called *remote host*. Local host and remote host needs to be connected through network. The remote host which is a server must be configured to run FTP services and the host computer must have FTP software to access these services.

4.2.1 Working of FTP

File Transfer Protocol is a client-server protocol. It depends on two channels between server and client: a command channel and a data channel. The command channel controls the conversation and the data channel transfers the contents of the files.

Working

- To access the contents user needs to log on to FTP server but some servers provide the content without login, which is known as anonymous FTP.

- When user demands for a file, the client started the conversation with the server.
- Client can move, copy, upload, delete and download files on a server by using FTP.

FTP has following two modes of working:

- **Active Mode: -** The server initiates a data connection back to client and started transferring files or data, after a client started a session through a command channel request.
- **Passive Mode: -** In passive mode, all the connections are started by client to remote server. Passive mode is useful for firewall and gateways.

4.2.2 Types of FTP

The following are some of the FTP server types:

- **Anonymous FTP: -** Anonymous FTP works on port. It allows the data transfer without encryption. It is mostly used to download the data which is allowed for unrestricted distribution. This is the basic type of FTP.
- **Password-Protected FTP: -** In this type of FTP, password and username is required, as the server may not be secure. This FTP works on port 21.
- **FTP Secure (FTPS): -** When FTP connection is initiated, the FTP enables the transport layer security (TLS). It is used to enable a more secure environment for data transfer. FTPS is also referred as FTP Secure Sockets Layers (FTP-SSL). It works on port 990.
- **FTP over explicit SSL/TLS (FTPES)**: **-** This type of FTP upgrades a connection over port 21 to secured connection. This is used by file sharing services and web to transfer secure files.

4.3 APACHE WEB SERVER

It is a web-server application commonly used in Linux type operating systems. It can also be used in all other operating systems. The name Apache came from the native American tribe 'Apache', which are famous for their skills and strategy making. The web-server provides the content to the use over the web. It can be web pages or any other documents. Apache web server is process-based that initiate a new thread with every connection. It supports many features. Virtual hosting is one of them which allows Apache wed server to provide number of different websites.

4.3.1 Apache Installation

There are many ways of installing this application as following:

1. In this open-source application, anyone can create installer as per their environment. It allows various vendors like Red Hat, Free BSD, Debian etc to configuration of Apache and customize the file location.

2. Another option is to build and install it from the source code, which is platform independent and it works for all types of operating systems.

4.3.2 Apache Virtual Host

The VHost or the Virtual Host is the feature of Apache web server. Through this feature the Apache web server can host multiple websites on the single server. There is no need for separate server machine and software for every website. In Apache configuration file all the domain to be hosted on web server have separate entry.

Types of Apache Virtual Host

- **Name Based Virtual Host**: - this hosting is used to host many different virtual sites on a single IP address. In order to receive Apache request for all websites, we need to set IP address. This can be done by NameVirtualHost directive i.e., httpd.conf/apache2.conf-file.

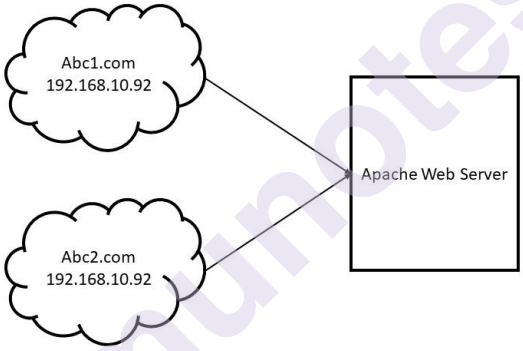


FIGURE 4.2

- **IP-Based Virtual Host**: - In this hosting, we need more than one IP address on the server. The VHost depend on number of IP addresses for example, if server has 5 IP addresses, then it creates 5 IP addresses based virtual hosts.

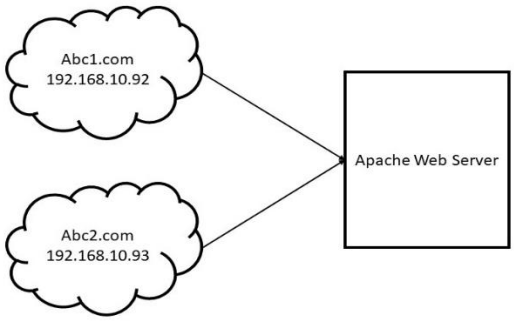


FIGURE 4.3

4.4 SIMPLE MAIL TRANSFER PROTOCOL

SMTP (Simple Mail Transfer Protocol) is used to transfer messages from one server to another through mails. This is used by e-mail systems.

Mail User Agent: - Client or agent application is used to read, write and send email messages. If anyone wants to send a message by a computer, has to use any mail user agents. For example, outlook, Gmail etc.

Mail Transfer Agent: - This agent application works to transferring email messages from one network to another. For example, for Linux: Postfix, Qmail and for windows: exchange server.

Mail Delivery Agent: - It is a software which is responsible for the delivery of messages via email to local recipient. It is also called local delivery agent (LDA).

4.5 POST OFFICE PROTOCOL

Post Office Protocol (POP) is the protocol used by email client applications to fetch email from servers. In POP, server, email messages are downloaded and it deleted the messages automatically on email server after its successful transfer, however this setting can be changed.

This protocol is compatible with MIME (Multipurpose Internet Mail Extension), which authorize for email attachments. This protocol is best for users which do not have a determined connection to the internet or have one system to read email.

The current version of POP is POP3.

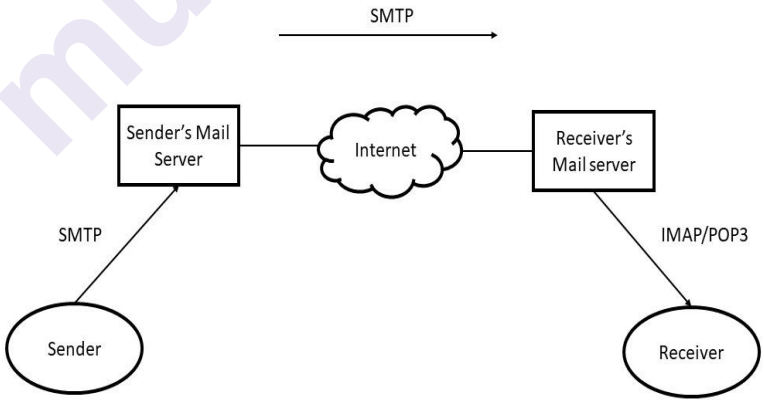


FIGURE 4.4

4.6 INTERNET MAIL ACCESS PROTOCOL

IMAP (Internet Mail Access Protocol) is the protocol which is also used by email client application to fetch email from servers. In IMAP server, users can read or delete messages on the server. It also allows the client to create, delete or rename directories on the server. This is best for users who

connects to mail server via a slow connection and using multiple machines. The user can delete messages locally, so that user can read messages without connecting to IMAP server.

4.7 SECURE SHELL

SSH (Secure Shell) is also known as Secure Socket Shell. It is a network protocol. It gives a secure way to users to access a computer over a network which is unsecured. It provides strong public key and password authentication. It also allows encrypted data communication over a network between two systems. It is also used by network administrators to manage systems remotely like move files, execute commands or enable log in. It uses client-server model which refers to cryptographic network protocol and the utilities that implement that protocol. It ships by default with Linux or Unix server. To secure different types of connections between a local machine and a remote host, SSH connections are used. It is also used to manage hardware, routers, operating systems and file transfer applications.

4.7.1 Capabilities of SSH

- Secure file transfer sessions.
- Secure remote access.
- Automated process and file transfer.
- Secure issuance of commands on network or systems.
- Security of network infrastructure components.

4.8 NETWORK AUTHENTICATION

It is an important task to provide secure environment to the user over the network. Only authenticate users can access the system, so that the data can be secured. There are different services in Linux which connects to the domain.

Authentication means to identify user and verify to a system. It needs user name and password. Then the name and password are compared to information stored in the database. In Linux, a configuration tool LDAP helps to configure the stored data for user credentials.

4.8.1 Confirming User Identities

Authentication means to confirm the identity of the user. The following are some ways to use authentication over networks:

Password-Based Authentication: - Password-based authentication is also called simple authentication. All the software applications permit the user to authenticate by its credentials.

Certificate-Based Authentication: - It is a part of SSL protocol. The user digitally signed a randomly generated of data and sends data and certificate over the network. Then, the server identifies the signature and checks the validity of the certificate.

Smart-Card Based Authentication: - The smart card is also called token. It is the variation of certificate-based authentication. The token stores user certificate. A user inserts the smart card into the system, read the certificate and access the grant.

Kerberos Authentication: - It allows user to undergo a single authentication process. It has short-lived credentials, called ticket-granting tickets (TGTs). The user presents username and password, identify the user and issued a ticket. It can be used to services like email and websites.

4.9 OPEN LDAP SERVER

LDAP (Lightweight Directory Access Protocol) provides a central directory which is accessible from the network, using a client-server architecture. The server checks the user's permission, when a user wants to modify any information in the directory, then it updates the information as requested. Transport layer security (TLS) cryptographic protocol is used for secure communication.

The OpenLDAP is a server package, which is used with Red Hat Enterprise Linux 7.4

4.9.1 Features

The following are the features:

LDAPv3 Support: - This includes secure socket protocol (SSL), transport layer protocol (TLS) and simple authentication and security layer (SASL) for more secure environment.

LDAP over IPC: - Inter-process communication increases the security by eliminating the requirement to communication on a network.

IPv6 Support: - OpenLDAP is not much compatible with IPV6, which is the next generation internet protocol.

LDIFv1 Support: - OpenLDAP is not compatible with this support.

Enhanced Standalone LDAP server: - It has updated thread policy, access control system and better tools.

4.9.2 LDAP Terminology

The following is the terminology used in LDAP:

Entry: - A single unique entry with different or distinguished name in the LDAP directory.

Attribute: - the information that is related to entry in LDAP directory. It can have single or multiple values.

LDIF: - LDAP data interchange format is a plain text used for LDAP entry.

4.10 SAMBA

SAMBA is Server Message Block (SMB) protocol in Linux. It is used to access resources like shared printers or files on a server. It also executes the Distributed Computing Environment Remote Procedure Call (DCE RPC) protocol which is used by Microsoft windows. It is available for free. The samba can be run as:

- A standalone server.
- As NT4 domain member.
- An active directory.
- Backup domain controller (BDC).
- NT4 primary domain controller (PDC).

4.10.1 Samba Services

The following are the services used in Samba:

- **smbd**: Using SMB protocol, sharing of printers and files can be provided by this service. This service is also responsible for connecting users and resource locking. Install samba package to use this service.
- **nmbd**: This service enables browsing the SMB network to find hosts, domains, work groups and printers. This also provides IP resolution and host name using NetBIOS over IPV4 protocol.
- winbindd: this service allows an interface for the Name Service Switch (NSS) to use NT4 domain users and groups on the local systems. This service must be started before smbd service.

4.10.2 Security Levels

To improve security, SMB refers following ways:

User Level Security: When the user register with the server, each user has their own password. Then the server grant permission to individual users according to their request.

ADS Level Security: To use this level of security the machine running samba needs Kerberos installed. Samba acts as domain member in an active directory environment.

Domain Level Security: This mode correctly works with windows NT domain. Samba pass the username and password to windows NT primary domain controller. This is set to yes of encrypted password parameters.

4.10.3 Samba Terminology

SMB protocol: Samba uses the Server Message Block (SMB) protocol which is based on the NetBIOS services. On all the clients there must be TCP/IP protocol installed because, SMB works on top of TCP/IP protocol with samba. SMB is released by Microsoft, so that the software manufacturers can develop connections to a Microsoft domain network.

CIFS protocol: CIFS is common Internet file system. This protocol is also known as SMB1. It is the version of the SMB protocol. This protocol used to develop a remote file system access protocol on the network. This allows the user to work together and share the files or documents on the network. Due to some security reasons, it is disabled in latest version of samba.

NetBIOS: It is a software interface. It is used for communication among machines and provides a name service. It allows machines, which are connected to the network to reservea name for themselves. Machines can reserve as many names as it wants. It works with different network architecture.

Samba server: This server provides SMB/CIFS and NetBIOS IP naming services to clients. In Linux, there are three daemons for Samba server: smbd for SMB/CIFS services, nmbd for naming services, and winbind for authentication.

Samba client: It is a client system that uses Samba services from a Samba server over the SMB protocol. The operating systems, like Windows and macOS support the SMB protocol. There should be TCP/IP protocol installed on all computers. In Linux, there is a kernel module that allows the integration of SMB resources on the Linux system level. User does not need to run any daemon for the Samba client.

Shares: SMB servers provide resources to the clients which are called shares. Shares are directories and printers, and their subdirectories on the server. It is accessed by its name. The name of the share can be set to any name. the resources can also assign a name. Clients can access the resources by its name.

DC: A domain controller is a server. It handles accounts in a domain. There are additional domain controllers in one domain for data replication.

4.11 NETWORK AUTHENTICATION SYSTEM

Network authentication system provides automatic creation of account and password to synchronize between many computers over the network. Because of large workstations, there are so many client-servers with many systems, for that security can be major issue. So, these systems have an account for every user.

The following are some of the network authentication systems:

1. NetInfo is developed by NEXT computer, but now it is a part of Mac OS X.
2. Network information system (NIS) of Sun Microsystems and NIST.
3. MIT Kerberos, part of Microsoft windows XP.
4. RADIUS (Remote Authentication Dial-In User Service) used by ISPs (Internet Service Provider).
5. Lightweight Directory (LDAP) server that stores account information.

Sometimes administrators do not prefer to use network database management systems. Instead, they use master computer as one of the computer systems. Each computer has its own account. Master computer's /etc/passwd files are distributed to other systems.

4.12 DOMAIN NAME SERVICE

The DNS server also known as name server. It allows to change the IP address for a host without affecting the name-based queries. The information is stored in data elements, called Resource Records (RR) in DNS server. It is organized into tree like structure. The root domain denoted by. (dot), domain name com as top-level domain and next is the second level of directory.

4.12.1 Name Server Types

The following are the name server types:

Authoritative: -These name servers respond to the resource records which are the part of their zones only. It includes both master(primary) and slave (secondary) name servers.

Recursive: -It offers resolution servers, it is not authorized for any particular zone, but for all. It responses for all resolution which are cached in memory for a period of time, which is specified by resource record.

4.13 SECURITY

Security of OS is the basic responsibility of Linux Server administrator. It is a tough task to secure any system properly. But the following are some of the Linux server's security checklist which may not provide the full security but can reduce the risk.

- To update the system is most basic and important thing to do. So, one need to keep the system updated with security patches, which should be latest.
- The system should be updated with latest vulnerabilities via mailing lists or forums.

- Administrators should stop unauthorized or unwanted services on the server.
- Use security shell services.
- Administrators should check the integrity of critical files.
- There should be a good firewall policy.
- Delete unwanted users and create required number of users.
- Administrator should check file permission from filesystems.
- Use BIOS and Boot locker security.
- Administrator should keep a good password policy.

4.14 SUMMARY

Domain Name System: -The naming system for computers is called *Domain Name System* (DNS). The DNS server is an internet service that maps the domain name into IP address so that human and computer can understand at their ends.

Name Space: -The naming system of DNS is hierarchical and logical tree structure, which is called *DNS name space.*

Primary DNS Server: - This server holds information like administrator's details and IP addresses. This contains the master copy of domain's configuration files.

Secondary DNS Server: - Secondary DNS Server is also called Slave DNS Server. These servers take the domain information from primary DNS server, which is read-only. Secondary server is used as a backup when primary server is crash or down.

Caching DNS Server: - This server is used to reduce the workload of primary and secondary DNS servers. This server contains the data that is recently requested by the user.

Local and Remote host: -In FTP, the computer of end user is called *local host* and FTP computer (Server) is called *remote host*. Local host and remote host needs to be connected through network. The remote host which is a server must be configured to run FTP services and the host computer must have FTP software to access these services.

Apache Web Server: - Apache web server is process-based that initiate a new thread with every connection. It supports many features. Virtual hosting is one of them which allows Apache wed server to provide number of different websites.

SMTP: -SMTP (Simple Mail Transfer Protocol) is used to transfer messages from one server to another through mails.

POP: -Post Office Protocol (POP) is the protocol used by email client applications to fetch email from servers.

IMAP: -IMAP (Internet Mail Access Protocol) is the protocol which is also used by email client application to fetch email from servers.

Secure Shell: -SSH (Secure Shell) is also known as Secure Socket Shell. It is a network protocol. It gives a secure way to users to access a computer over a network which is unsecured.

Password-Based Authentication: - Password-based authentication is also called simple authentication. All the software applications permit the user to authenticate by its credentials.

Certificate-Based Authentication: - It is a part of SSL protocol. The user digitally signed a randomly generated of data and sends data and certificate over the network. Then, the server identifies the signature and checks the validity of the certificate.

Smart-Card Based Authentication: - The smart card is also called token. It is the variation of certificate-based authentication. The token stores user certificate. A user inserts the smart card into the system, read the certificate and access the grant.

Kerberos Authentication: - It allows user to undergo a single authentication process. It has short-lived credentials, called ticket-granting tickets (TGTs). The user presents username and password, identify the user and issued a ticket. It can be used to services like email and websites.

Authoritative: -These name servers respond to the resource records which are the part of their zones only. It includes both master(primary) and slave (secondary) name servers.

Recursive: -It offers resolution servers, it is not authorized for any particular zone, but for all. It gives responses for all resolution which are cached in memory for a period of time, which is specified by resource record.

Security: -Security of OS is the basic responsibility of Linux Server administrator. It is a tough task to secure any system properly

4.15 EXERCISES

1. What do you mean by Domain Name System?
2. What are the naming Standards for DNS?
3. Explain the features of DNS Server?
4. What are the types of DNS Server?
5. What do you mean by File Transfer Protocol? Explain its working.
6. Explain the types of FTP.
7. What is Apache Server? Explain the ways of installation.

8. Explain Apache virtual host and its types.
9. What is SMTP, POP, IMAP?
10. Explain secure shell and its capabilities.
11. What are the user identities for network authentication?
12. Define the features of LDAP.
13. What are the security levels in samba services?
14. What do you mean by security in Linux server administrator?
15. Define Name Server types.

❄❄❄❄❄❄❄

5

INTRANET SERVICES

Unit Structure :

5.1 OBJECTIVES

- Demonstrate proficiency with the Linux command line interface, directory & file management techniques, file system organization, and tools commonly found on most Linux distributions.
- Effectively operate a Linux system inside of a network environment to integrate with existing service solutions.
- Demonstrate the ability to troubleshoot challenging technical problems typically encountered when operating and administering Linux systems.
- To study how to deploy the NFS

5.2 NETWORK FILE SYSTEM (NFS)

- NFS is a short form of the Network File System (NFS).NFS is a protocol of a distributed file system.
- This protocol emerged in the year of 1984 by Sun Microsystems.
- It is an architecture of the client-server, which contains a client program, server program, and a protocol that forms the communication between the client and server.

- It is that protocol which enables the users to approach the data and files remotely over the network. Every user can easily access & implement the NFS protocol because it is an open standard. Any user can alter the files as if they were on like other protocols.
- The NFS protocol is built on the ONC RPC system.
- This protocol is for the most part implemented in those computing environments where the centralized management of resources and data is critical.
- It uses the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) for accessing / obtaining and delivering the data and files on the network.
- Network File System is a protocol that works on all the networks based on IP address. NFS is implemented in that client-server application in which the server of NFS handle the authorization, authentication, and clients.
- This protocol is used with different kinds of operating systems. Example Apple Mac OS, Unix, and Unix-like operating systems such as Solaris, Linux, FreeBSD, AIX.

Difference Between NFS and CIFS

NFS	**CIFS**
1. NFS is a short form of the Network File System.	1. CIFS is an short form of the Common Internet File system.
2. This protocol is used for transfering the files by Unix and Linux Operating systems.	2. This protocol is used for sharing the files by Windows Operating systems.
3. It is highly scalable.	3. It is low scalable.
4. The speed of communication is fast.	4. The speed of communication is medium.
5. The network File system is not a secure protocol.	5. The Common Internet File System is more secure and authentic than the Network File System.
6. NFS is not a reliable protocol.	6. CIFS is a reliable protocol.
7. This protocol does not provide the session.	7. This protocol provides the sessions.

8. This protocol is simple to implement and set up.	8. Its implementation is complex.
9. This protocol uses 111 port for both TCP and UDP.	9. This protocol operates on 139 and 445 TCP ports and 137 and 138 UDP ports.

5.3 SAMBA

- Samba is a powerful platform of applications that enables UNIX-based systems (such as Linux) to interoperate with Windows-based and other operating systems.
- It is an open source implementation of the Server Message Block (SMB) or Common Internet File System (CIFS) protocol platform.
- Samba transparently provides file and print sharing services to Windows clients as well as other connected networked clients running on the other operating systems.
- This executes through the use of the native Microsoft networking protocols SMB/CIFS. In the system administrator's point of view, this means we can deploy a Linux UNIX-based server and use it to provide file sharing, authentication, printing, and other services to other non-native Linux clients such as Microsoft Windows systems.
- Using Samba means that Windows systems can use their native tongue to talk to the Linux server which means fewer hassles for us and seamless integration for our users.
- Samba provides a well set of configuration options that makes it suitable and flexible for use in various environments..

5.4 DISTRIBUTED FILE SYSTEMS (DFS)

- A distributed file system (DFS) is a file system that is distributed on numerous file servers and locations.
- It allows applications to access and store isolated data in the same method as in the local files.
- It also allows the user to access files from any of the systems.
- It allows network users to pass the information and files in a synchronized and permitted manner.
- In spite of this, the servers have whole control over the data and provide users access control.
- DFS's primary aim is to enable users of physically distributed systems to share resources and share information through the Common File System (CFS).

- It is a file system that runs behalf of the operating systems. Its arrangement is a set of workstations and mainframes that a LAN connects to each other.
- The creation of a namespace process in DFS is transparent to the clients.
- DFS has two components in its services:
    - Local Transparency
    - Redundancy

Local Transparency

It is achieved via the namespace component.

Redundancy

It is achieved via a file replication component.

- In the situation of failure or heavy load, these components work jointly to increase data availability by allowing data from numerous places to be logically merge under a single folder known as the "DFS root".
- It is not required to use both DFS components together, the namespace component can be used without the file replication component, and the file replication component can be used in between servers without the namespace component.
- There are various features of the DFS.
    - Transparency

There are mainly four types of transparency. These are as follows:

1. **Structure Transparency**
    - In structure transparency, multiple file servers must be specified to adaptability, dependability, and performance.
2. **Naming Transparency**
    - There should be no specific details of the file's location in the file's name. When the file is transferred from one node to another, the file name should not be changed.
3. **Access Transparency**
    - Local and remote files must be usable or accessible in the same way. The file system must automatically find the accessed file and deliver it to the appropriate client.
4. **Replication Transparency**
    - When a file is copied across various nodes, the copied files and their locations must be hidden from one node to the next.

- **Scalability**
  - The distributed system will automatically increase over time when more machines are added to the network, or two networks are linked to each other.
  - A good DFS must be planned to scale quickly as the system's number of nodes and users increases.
- **Data Integrity**
  - Many users usually share a file system.
  - The file system is required to secure the integrity of data saved in a transferred file.
  - A concurrency control method should correctly synchronize simultaneous access requests from several users who are competing for access to the same file.
  - A file system often provides users with atomic transactions that are high level concurrency management systems for data integrity.
- **High Reliability**
  - The chance of data loss must be finite as much as feasible in an effective DFS.
  - Users must not feel forced to make backups of their files due to the system's unreliability.
  - Alternatively, a file system should back up key files so that they may be restored if the originals are lost.
  - As a high-reliability strategy, many file systems use fixed storage.
- **High Availability**
  - A DFS should be able to role in the case of a partial failure, like a node failure, a storage device crash, and a link failure.
- **Ease of Use**
  - The user interface of a file system in multiprogramming must be easy, and the commands in the file must be minimal.
- **Performance**
  - The average time it takes to convince a client is used to assess performance. It must perform similarly to a centralized file system.

5.5 NETWORK INFORMATION SERVICE (NIS)

- Network Information Service (NIS) is a distributed database that enables the maintenance of consistent configuration files throughout the network.
- NIS is the recent name for the service originally known as *Yellow Pages* (YP). NIS and YP are functionally identical
- NIS is a part of the Network File System (NFS) software package that consist commands and spirit for NFS, NIS, and other services.
- In spite of NFS and NIS being installed together as one package, each is independent and each is configured and managed individually.
- Components of NIS
    - The NIS environment is built of *clients* and *servers* logically grouped together in a *domain*.
    - Each domain has a specific set of characteristics.
    - These characteristics are described in *maps,* or databases, that specify certain system information such as user names, passwords, and host names.

Servers

- An NIS *server* is a host that supply configuration information to other hosts on the network.
- Servers hold a set of maps and run the ypserv daemon, which processes requests from clients for information contained in maps.
- There are two types of servers namely a *master* server and a *slave* server.

Master Servers

- A *master* server is the single host in a distinct domain that maintains the authoritative maps.
- The master server runs ypupdated daemon, which causes slave servers to update their copies of the maps.
- The master server also runs the yppasswdd daemon, which processes requests to change users' passwords.
- Characteristics of the master server include:
    - Accessible by the system administrator. If something goes wrong, or if updates need to be made, it is **simple** to reach the master server.

- The master server generally stays active for long periods of time. It is stable so systems that depend on it can depend on uninterrupted service.
- Accessible from the network. In spite of networks can be complex with the presence of many gateways or bridges, the master server is available from most systems on the network
- For a small number of hosts, each host can use the master server directly. Therefore, for a larger number of hosts in a domain, the master server can become overloaded. To balance the NIS processing load and provide services when the master server is inaccessible, additional hosts can be designated as slave servers.

Slave Servers

- NIS *slave* servers operate intermediaries between clients and the master server by keeping exact replicas of the master server's maps.
- All alterations to the maps are made on the master server.
- The changes are reflected from the master server to the slave servers.
- Once a slave server is attach to the domain, it is able to answer the same queries that the master is able to answer.
- In this way, slave servers can assist with extra load on the master server without violating the authority of the master server.
- Slave servers also act as a backup in the situation where the master server or the network fails. A client requesting information waits upto a server responds.
- This waiting time differs depending on the reason the server is unreachable.
- Adding slave servers grow the availability of information even if the master server is unavailable.
- Normally, there should be at least one slave server for individual domain.
- The number of slave servers in a domain should be steady to achieve the desired level of availability and response time without adding the expense of copying data to too many systems.

Clients

- NIS *clients* make up the most of hosts in a NIS domain.
- Clients run the ypbind daemon, which allows client processes to obtain information from a server.

- Clients do not preserve maps themselves, but rather query servers for system and user account information. Clients do not make a dissimilarity between querying the master server or a slave server.
- To access system information accommodate in a map, a client makes a Remote Procedure Call (RPC) to a server.
- The server looks at its local database and returns the requested information to the client.
- NIS clients locate the server by relay on the networks that are directly connected to the client machine.
- Since these broadcast messages are not forwarded by network gateways, if there is no NIS server that can be reached except using a network gateway, the client must mention a server when starting the ypbind daemon.
- Every request for system information needs a server contact, and the speed of network can affect the response time. Although a local retrieval is normally faster than a network retrieval, the advantages of NIS outweigh the compromise in access time.

NIS Domain

- An NIS domain is a group of systems that are logically grouped together.
- A group of hosts that share the same set of NIS maps belong to the same domain.
- The hosts are normally grouped together in the domain for a common reason; for example, when functioning in the same group at a particular location.
- Each NIS host is assigned to a domain when the system starts. The domain name must be set on all hosts that plan to use NIS.
- There is one master server per NIS domain, and the systems in the domain are commonly on the same network.

NIS Maps

- NIS *maps* are databases that show certain system information such as user names, passwords, and host names, in a database format called Database Management.
- Each map is built from a standard text file by associating an index *key* with a *value*.
- For example, the information in the master server's /etc/hosts file is used to generate a map that uses each host name as a key, and the IP address as the value.
- The key and value pairs (also called as *records*) that are created from the entries in the/etc/hosts file contain the *hosts.byname* map.

5.6 LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL (LDAP)

- Lightweight Directory Access Protocol (LDAP) is an internet protocol that works on TCP/IP model & is used to access information from directories.
- LDAP protocol is generally used to approach an active directory.

Features of LDAP:

1. Functional model of LDAP is simpler due to this it excludes duplicate, rarely used and difficult features.
2. LDAP is simple to understand and implement.
3. LDAP uses strings to represent data

Directories:

- Directories are a set of objects with the same attributes, organized in a logical and hierarchical manner.
- For example, Telephonic Directories. It is a distributed database application used to handle attributes in a directory.

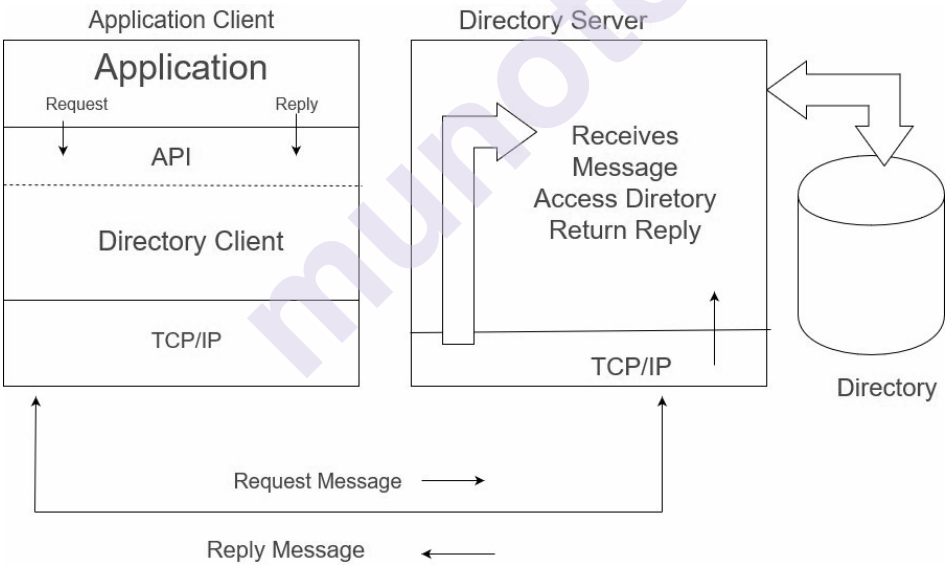


Fig. 1 LDAP

LDAP explained operations for accessing and modifying directory entries like:

- finding for user specified criteria
- Adding an entry
- Deleting an entry
- Modifying an entry

- Changing the distinguished name or relative distinguished name of an entry
- Comparing an entry

LDAP Models:

LDAP can be explained with the help of this four models upon which it depends:

1. **Information Model:**
   - This Information model explains the structure of information stored in an LDAP Directory. Here the basic information stored in a directory is called an entity.
2. **Naming Model:**
   - This model illustrate how information in an LDAP Directory is organized and identified.
3. **Functional Model:**
   - LDAP characterizes operations for accessing and modifying directory entries .
   - Here LDAP operations in a programming language independent manner LDAP operations can be split into following categories:
     - Query
     - Update
     - Authentication
4. **Security Model:**

This model describes how information in the LDAP directory can be protected from unauthorized access. It is based on the BIND operation. There are several bind operations that can be performed.

5.7 DYNAMIC HOST CONFIGURATION PROTOCOL (DHCP)

- Dynamic Host Configuration Protocol (DHCP) is a network management protocol used to dynamically allocate an IP address to any of the device, or node, on a network so they can communicate with each other using IP (Internet Protocol).
- DHCP automates and centrally handles these configurations. There is no demand to manually assign IP addresses to new devices.
- Therefore, no need for any user information or configuration to connect to a DHCP based network.
- DHCP can be executed on local networks as well as on the large enterprise networks.

- DHCP is the default protocol used by respective routers and networking devices.
- DHCP is also called Request for comments 2131.

Function of DHCP:

- DHCP controlled and handle the provision of all the nodes or devices added or dropped from the network.
- DHCP stores the unique IP address of the host using a DHCP server.
- It sends a common request to the DHCP server whenever a client/node/device, which is configured to work with DHCP, connects to a network.
- The server acknowledges by providing an IP address to the client or node or device.
- DHCP is also used to configure the actual subnet mask, default gateway and DNS server information on the node or device.
- At present, there are many versions of DHCP available for use in IPV4 (Internet Protocol Version 4) and IPV6 (Internet Protocol Version 6).

Working of DHCP

- DHCP runs at the application layer of the TCP/IP protocol stack to dynamically assign IP addresses to DHCP clients or nodes and to allocate TCP/IP configuration information/data to the DHCP clients.
- Information like subnet mask information, default gateway, IP addresses and domain name system addresses.
- DHCP is based on client-server protocol in which servers control a pool of unique IP addresses, as well as information about client configuration parameters, and assign addresses out of those address pools.

The DHCP lease process works as follows:

- First of all, a client e.g.network device, must be connected to the internet.
- DHCP clients request an IP address. usually the client broadcasts a query for this information.
- DHCP server acknowledges or responds the client request by providing IP server address and other configuration information. This configuration information also consists of a time period, called a lease, for which the allocation is valid.
- While refreshing, a DHCP client requests the same parameters, but the DHCP server may allocate a new IP address. This is based on the policies set by the administrator.

Components of DHCP

While working with DHCP, it is necessary to understand all of the components. Following are the list of components:

- DHCP Server: DHCP server is a networked device running the DCHP service that takes IP addresses and related configuration information. This is normally a server or a router but can be anything that acts as a host, like an SD-WAN appliance.
- DHCP client: DHCP client is the endpoint that gets configuration information from a DHCP server. This can be device like computer, laptop, IoT endpoint or anything else that needs connectivity to the network. Many devices are configured to receive DHCP information by default.
- IP address pool: IP address pool is the scale of addresses that are available to DHCP clients. IP addresses are typically handed out serially from lowest to the highest.
- Subnet: Subnet is the division segment of the IP networks. Subnet is used to keep networks attainable.
- Lease: Lease defines the length of time for which a DHCP client carries the IP address information. When a lease run out, the client has to renew it.
- DHCP relay: A host or router that reads for client messages being broadcast on that network and then forwards them to a configured server. The server then sends responses back to the relay agent that moves them along to the client. DHCP relay can be used to centralize DHCP servers in spite of having a server on each subnet.

Benefits of DHCP

There are following benefits of DHCP:

- **Centralized administration of IP configuration:** DHCP IP configuration information can be stored in a single location and enables that administrator to centrally manage all IP address configuration information.
- **Dynamic host configuration:** DHCP automates the host configuration process and eliminates the need to manually configure individual hosts. When TCP/IP (Transmission control protocol/Internet protocol) is first deployed or when IP infrastructure changes are required.
- **Seamless IP host configuration:** The use of DHCP ensures that DHCP clients get accurate and timely IP configuration IP configuration parameters such as IP address, subnet mask, default gateway, IP address of DNS server and so on without user intervention.

- **Flexibility and scalability:** Using DHCP gives the administrator increased flexibility, allowing the administrator to easily change IP configuration when the infrastructure changes.

5.8 MYSQL

- MySQL is the most popular Open Source Relational SQL database management system. MySQL is one of the best RDBMS being used for developing web-based software applications.
- A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.
- Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those types of systems.
- Nowadays, we use relational database management systems (RDBMS) to store and manage huge volumes of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as Foreign Keys.

A Relational DataBase Management System (RDBMS) is a software that

- Enables you to implement a database with tables, columns and indexes.
- Guarantees the Referential Integrity between rows of various tables.
- Updates the indexes automatically.
- Interprets an SQL query and combines information from various tables.

RDBMS Terminology

Before we proceed to explain the MySQL database system, let us revise a few definitions related to the database.

- Database − A database is a collection of tables, with related data.
- Table − A table is a matrix with data. A table in a database looks like a simple spreadsheet.
- Column − One column (data element) contains data of one and the same kind, for example the column postcode.
- Row − A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.
- Redundancy − Storing data twice, redundantly to make the system faster.
- Primary Key − A primary key is unique. A key value can not occur twice in one table. With a key, you can only find one row.
- Foreign Key − A foreign key is the linking pin between two tables.

- Compound Key − A compound key (composite key) is a key that consists of multiple columns, because one column is not sufficiently unique.
- Index − An index in a database resembles an index at the back of a book.
- Referential Integrity − Referential Integrity makes sure that a foreign key value always points to an existing row.

MySQL Database

- MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons
- MySQL is released under an open-source license. So you have nothing to pay to use it.
- MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
- MySQL uses a standard form of the well-known SQL data language.
- MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
- MySQL works very quickly and works well even with large data sets.
- MySQL is very friendly to PHP, the most appreciated language for web development.
- MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
- MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

5.9 LAMP APPLICATIONS FILE SERVERS

- LAMP installations (Linux + Apache + MySQL + PHP/Perl/Python) are a popular setup for Ubuntu servers.
- There are a plethora of Open Source applications written using the LAMP application stack. Some popular LAMP applications are Wiki's, Content Management Systems, and Management Software such as phpMyAdmin.
- One advantage of LAMP is the substantial flexibility for different database, web server, and scripting languages. Popular substitutes for MySQL include PostgreSQL and SQLite. Python, Perl, and Ruby are also frequently used instead of PHP. While Nginx, Cherokee and Lighttpd can replace Apache.

- The fastest way to get started is to install LAMP using tasksel. Tasksel is a Debian/Ubuntu tool that installs multiple related packages as a co-ordinated "task" onto your system. To install a LAMP server:

At a terminal prompt enter the following command:

sudo tasksel install lamp-server

After installing it you'll be able to install most *LAMP* applications in this way:

1. Download an archive containing the application source files.
2. Unpack the archive, usually in a directory accessible to a web server.
3. Depending on where the source was extracted, configure a web server to serve the files.
4. Configure the application to connect to the database.
5. Run a script, or browse to a page of the application, to install the database needed by the application.
6. Once the steps above, or similar steps, are completed you are ready to begin using the application.

- A disadvantage of using this approach is that the application files are not placed in the file system in a standard way, which can cause confusion as to where the application is installed. Another larger disadvantage is updating the application. When a new version is released, the same process used to install the application is needed to apply updates.

- Fortunately, a number of *LAMP* applications are already packaged for Ubuntu, and are available for installation in the same way as non-LAMP applications. Depending on the application some extra configuration and setup steps may be needed, however.

5.10 EMAIL SERVICES

There are three components to a mail service on a Linux email server:

- Mail user agent (MUA) is the GUI, the part that lets you write and send emails, like Thunderbird or Outlook.

- Mail transport agent (MTA) is the bit that moves the mail (as the name suggests). MTAs like Sendmail and Postfix are the parts that waft your communications from place to place through the ether.

- Mail delivery agent (MDA) is the component that sends out messages sent to you on your local machine, so they get to the appropriate user mailbox. Postfix-maildrop and Procmail are examples.

- In order to configure a Linux mail server, you'll first need to check if Postfix is already installed. It's the default mail server on the lion's share of Linux distributions these days, which is good because server admins like it a lot.

Here's how to check if it's already on the system:

$ rpm -qa | grep postfix

If not, this is how you install it on Red Hat distributions:

$ dnf -y install postfix

Next, run it and activate it on system start-up:

$ systemctl start postfix

$ systemctl activate postfix

For distributions based on Debian, like Ubuntu, you'd install them like this:

$ apt-get -y install postfix

- As you configure the Linux mail server you will receive a prompt to choose how you want to configure your Postfix mail server.

  You'll be presented with these choices:

- No configuration
- Internet site
- Internet with smarthost
- Satellite system and Local only

5.11 CHAT APPLICATIONS

- The popularity of Linux has been able to replace Windows at many workplaces, and the same scenario was also reported for personal users.
- Hence many popular apps from various platforms such as Android and Windows are being integrated into Linux and its distros.
- Business emails/chats are also being replaced by instant messaging and communication apps as they provide more options to share files, photos, and videos, making the whole process easier.
- It would be great to use messaging apps on the Linux desktop that we use on our mobile devices.
- According to your mood, apps like these give you the flexibility to use the messaging app wherever you want.

1. Telegram

Telegram is one of my favorite instant messaging and communication apps out there. It is an open-source app that can be used on a mobile device as well as desktop. Cloud-based technology makes it easy to access from anywhere and several devices at once.

2. Skype

Skype is one of the oldest messaging and communication apps for desktops out there, but it is still one of the best for video conferencing. Right from college lectures to business meets, everything is being done on Skype for almost a decade, and its worth has soared up even more in a recent situation.

3. Slack

Slack is a powerful messaging app specially developed for business use as it offers a lot of options and configurations. It is one of the best out there but comes with a price tag. It lets you chat and have a conversation with your colleagues in real-time, just what every business needs for smooth functioning. It comes with a modern user interface which makes it very easy to use, even for a newbie.

4. Wire

The wire is another modern desktop messaging platform ideal for business use. It provides quite a competition to Slack in terms of features offered. These messaging apps are voice, video, conference calls, file sharing, and external collaboration. Everything is secured by end-to-end encryption.

5.12 VIRTUAL PRIVATE NETWORKING.

- VPN stands for the virtual private network. A virtual private network (VPN) is a technology that creates a safe and encrypted connection over a less secure network, such as the internet.
- A Virtual Private Network is a way to extend a private network using a public network such as the internet.
- The name only suggests that it is a Virtual “private network” i.e. a user can be part of a local network sitting at a remote location. It makes use of tunneling protocols to establish a secure connection.

VPN example:

- Think of a situation where the corporate office of a bank is situated in Washington, USA. This office has a local network consisting of say 100 computers. Suppose other branches of the bank are in Mumbai, India, and Tokyo, Japan. The traditional method of establishing a secure connection between head office and branch was to have a leased line between the branches and head office which was a very costly as well as troublesome job. VPN lets us overcome this issue in an effective manner.

15.13 SUMMARY

- NFS has been around for a long time now, and as such, it has gone through several revisions of the protocol specifications. The revisions are mostly backward-compatible, and each succeeding revision can support clients using the older versions.
- Samba is a powerful suite of applications that helps UNIX-based systems (such as Linux) interoperate with Windows-based and other operating systems.
- A distributed file system (DFS) is a file system that is distributed on various file servers and locations.
- Network Information Service (NIS) is a distributed database that allows you to maintain consistent configuration files throughout your network.
- Dynamic Host Configuration Protocol (DHCP) is a network management protocol used to dynamically assign an IP address to nay device, or node, on a network so they can communicate using IP (Internet Protocol).
- MySQL is the most popular Open Source Relational SQL database management system. MySQL is one of the best RDBMS being used for developing web-based software applications.
- VPN stands for the virtual private network. A virtual private network (VPN) is a technology that creates a safe and encrypted connection over a less secure network, such as the internet.

5.14 REFERENCE FOR FURTHER READING

- Linux Administration A Beginners Guide, Wale Soyinka, McGraw Hill

5.15 UNIT END EXERCISES

1. Explain the difference between NFS and CIFS?
2. Write a short note on Samba
3. What is DFS ? Explain the types of DFS.
4. What is NIS? Explain the components of NIS.
5. What is LDAP? Explain its features.
6. Explain the working of DHCP.

❄❄❄❄❄❄❄