System Administration Using Linux

Copy Notes

by

Mukesh Singh
(ICT 5th batch, 2073)

Sukuna Multiple Campus
Sundarharaincha-12, Morang.

Unit-1

Familiarizing with Linux and CLI Commands

1.1 Introduction to free and open source software, Unix System Architecture, Unix Philosophy, Introduction to Linux, How is Linux different? The X windows system, Run level or system initialization levels, Virtual Consoles(terminals), Command prompt, using a Linux System, Linux Command line, Logging out, Command Syntax, Files, Creating Files with cat, Displaying files contents with cat, Deleting files with rm, Copying and renaming files with cp and mv, Command History

Introduction to free and Open Source Software

Free and Open Source Software (Foss), is software for which the source code can be freely shared, with anyone, for any purpose.

Foss is software that is both free software and open source software where anyone is freely licenced to use, copy, study and change the software in any way, and the source code is openly shared so that people are encouraged to voluntarily improve the design of the software. This is in contrast to proprietary software, where the software is under restrictive copyright licensing and the source code is usually hidden from the users.

Few examples of Foss:

- Linux
- Android
- MySQL
- Pearl
- PHP
- Python
- Open office

Free Software

"Free Software" means software that respects users' freedom and community. Roughly, it means that the users have the freedom to run, copy, distribute, study, change and improve the software. The term "free software" is sometimes misunderstood - it has nothing to do with price. It is about freedom.

Open Source Software

Open source software is something you can modify as per your needs, share with others without any licensing violation burden. When we say open source, source code of software is available publicly with Open Source licenses like GNU (GPL) which allow you to edit source code and distribute it.

	Free Software (FS)	Open Source Software (OSS)
1.	Software is an important part of people's lives.	1. Software is just software. There are no ethics associated directly to it.
2.	Software freedom translates to social freedom	2. Ethics are to be associated to the people not to the software.
3.	Freedom is value that is more important than any economical advantage.	3. Freedom is not an absolute concept. Freedom should be allowed not imposed.
4.		
4.	Examples: Linux kernel, BSD and Linux OS, GNU Compiler Collection and C library, Apache Web Server etc.	4. Examples: Apache HTTP server, Mozilla firefox, Chromium, the e-commerce platform OS Commerce.

Unix System Architecture

Unix

Unix is a family of multitasking, multiuser computer operating system that drive from the original AT & T Unix, whose development started in the 1970s at the Bell Labs research center by Ken Thompson, Dennis Ritchie, and others.

There are various Unix variants available in the market. Solaris Unix, AIX, HP-Unix and BSD are a few examples. Linux is also a flavor of Unix which is freely available.

Unix Architecture

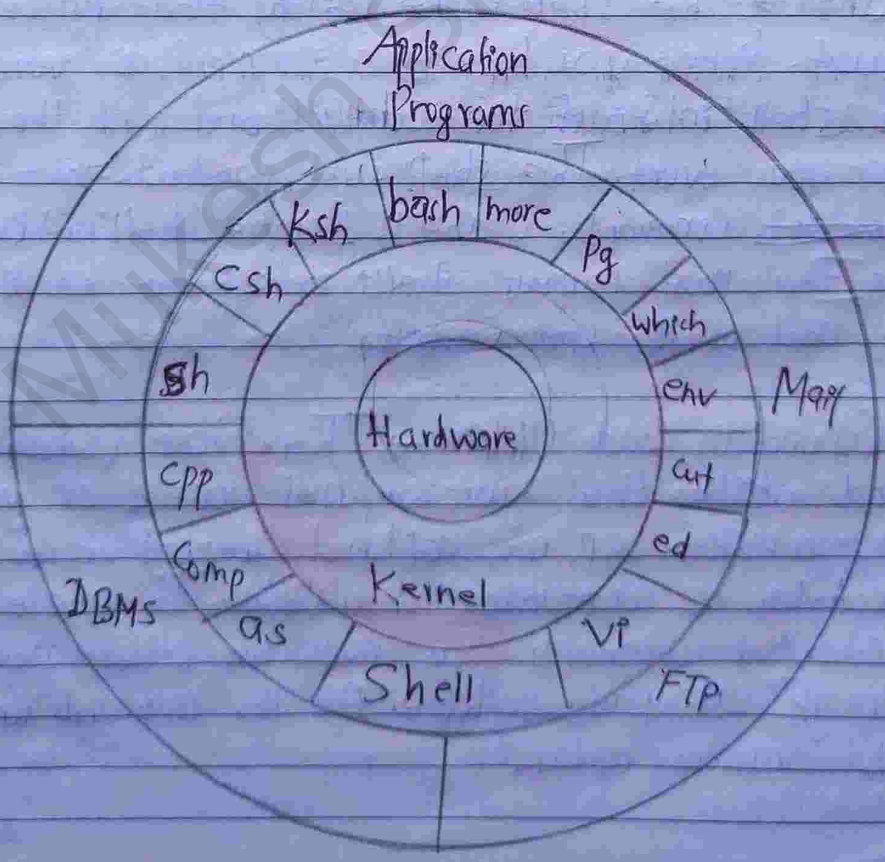


Fig: Basic Block Diagram of Unix System

1. Make each program do one thing well. To do a new job, build a fresh rather than complicated old programs by adding new features.

2. Expect the output of every program to become the input to another, as yet unknown program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.

3. Design and build software, even operating systems, to be tried early, ideally within ~~weeks~~ weeks. Don't hesitate to throw away the clumsy parts and rebuild them.

4. Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.

It was later summarized by Peter H. Salus in A Quarter-Century of Unix (1994).

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs to handle text streams, because that is a universal interface.

~~In their award~~

The main tenets of Unix Philosophy are as follows:

1. Small is beautiful.
2. Make each program do one thing well.
3. Build a prototype as soon as possible.
4. Choose portability over efficiency.
5. Store data in flat text files.
6. Use software leverage to your advantage.
7. Use shell scripts to increase leverage and portability.
8. Avoid Captive user interfaces.
9. Make every program a filter.

Introduction to Linux

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

Initially, Linux was created for personal computers and gradually it was used in other machines like servers, mainframe computers, supercomputers etc. Nowadays, Linux is also used in embedded systems like routers, automation controls, televisions, digital video recorders, video game consoles, smartwatches etc. The biggest success of Linux is Android (Operating System). It is based on the Linux kernel that is running on smartphones and tablets. Due to android, Linux has the largest installed base of all general-purpose operating systems. Linux is generally packaged in a Linux distribution.

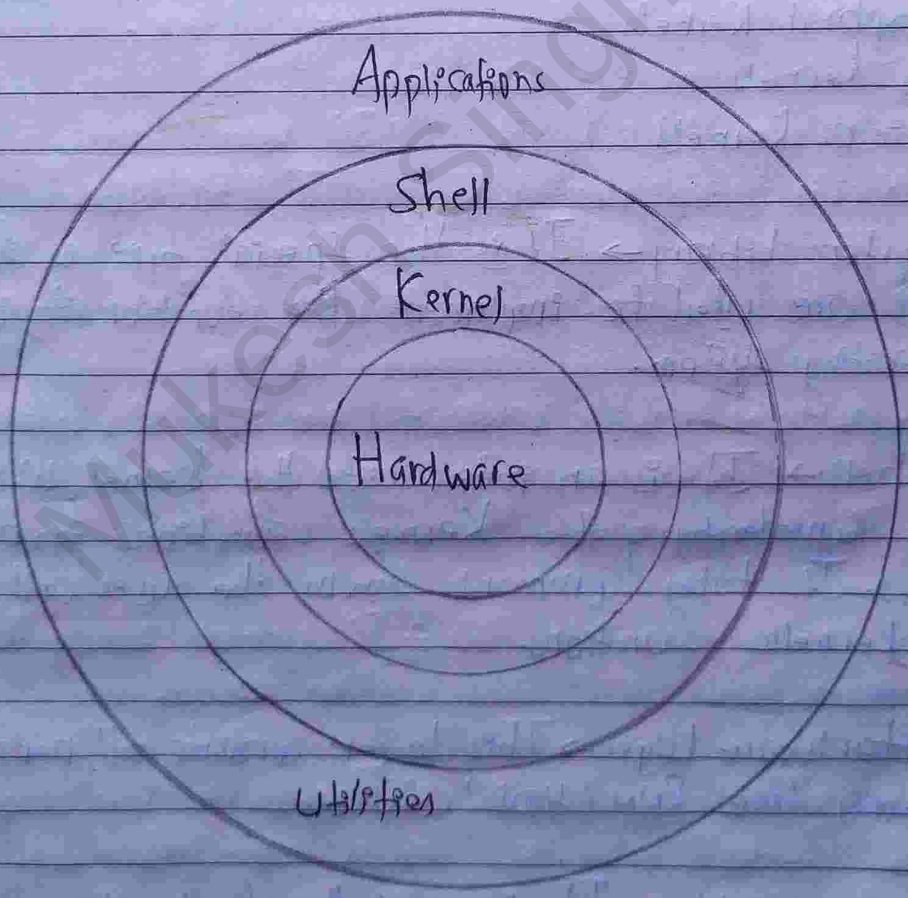
Linux Distribution

Linux distribution is an Operating System that is made up of a collection of software based on Linux kernel or you can say distribution contains the Linux kernel and supporting libraries and software. Around 600+ Linux Distributions are available and some of the popular linux distributions are:

- MX Linux
- Manjaro

- Linux Mint
- elementary
- Ubuntu
- Debian
- Solus
- Fedora
- OpenSUSE
- Deepin

Architecture of Linux



Linux Architecture has following components:

1. Kernel → Kernel is the core of the linux based operating system. It virtualizes the common hardware resources of the computer to provide each process with its virtual resources. They makes the process seem as if it is the sole process running on the machine. The Kernel is also responsible for preventing and mitigating conflicts between different process. Different types of Kernel are:
   - Monolithic Kernel
   - Hybrid Kernels
   - Exo Kernels
   - Micro Kernels

2. System Library → It is the special type of function that are used to implement the functionality of the operating system.

3. Shell → It is an interface to the kernel which hides the complexity of the Kernel's functions from the users. It takes commands from the user and executes the kernel's functions.

4. Hardware Layer → This layer consists all peripheral devices like RAM / HDD / CPU etc.

5. System Utility → It provides the functionalities of an operating system to the user.

Page No.
Date

How is Linux Different?

Linux is different from other operating systems in many ways. First, and perhaps most importantly, Linux is open source software. The code used to create Linux is free and available to the public to view, edit, and - for ~~others~~ users with appropriate skills - to contribute to.

Linux is also different in that, although the core pieces of the Linux operating systems are generally common, there are many distributions of Linux, which include different software options. This means that Linux is incredibly customizable, because not just applications, such as word processors and web browsers, can be swapped out. Linux users also can choose core components, such as which system displays graphics, and other user-interface components.

	Linux	Windows
1.	Linux uses a tree like a hierarchical file system.	1. Windows uses different data drives like C:D:E to stored files and folders.
2.	There are no drives in Linux.	2. Windows has different drives like C:D:E.
3.	Peripherals like hard drives, CD-ROM, printers are also considered files in Linux.	3. Hard drives, CD-ROMs, printers are considered as devices.
4.	There are 3 types of user accounts 1) Regular, 2) Root and 3) Service account.	4. There are 4 types of user account types 1) Administrator, 2) Standard, 3) Child, 4) Guest.
5.		

5. Root user is the super user and has all administrative privileges.	5. Administrative user has all administrative privileges of Computer.
6. Linux file naming is case sensitive. Sample and SAMPLE are 2 different file in Linux OS.	6. In windows, you can't have 2 files with the same name in the same folder.

Linux	Unix
1. It is an open-source OS which ~~can be~~ is freely available to everyone.	1. It is an OS which can be only used by its copyrighters.
2. It is developed by Linux Community of developers.	2. It was developed by AT & T Bell labs.
3. Linux is free to use.	3. Unix is licensed OS.
4. Linux is used in wide variety from desktop, servers, smartphones to mainframes.	4. Unix is mostly used on servers, workstations or PCs.
5. Linux supports more file system than unix.	5. It supports file system lesser than Linux.
6. The source code is available to the general public.	6. The source code is not available to anyone.
7. Some Linux versions are: Ubuntu, Debian, GNU, Arch Linux, etc.	7. Some unix versions are SunOS Solaris, SCO UNIX, AIX, HP/UX, ULTRIX etc.

The X Windows System

The X Windows System (Commonly X11 or X) is an Open source, cross platform, client server computer software system that provides a GUI in a distributed network environment. The X Windows system is also known as X, X11 or X Windows.

It is a windows system for graphics workstations developed at MIT in 1984 with support from DEC. It is based on client/server model: a network computer or workstation run as X server, and client programs running on connected workstations request services from the server. The server handles input and output devices and generates the graphical displays used by the clients.

The X Windows System is a system process that handles the displaying (drawing) of windows, frames, task bar, title bars, resizing windows, and moving windows.

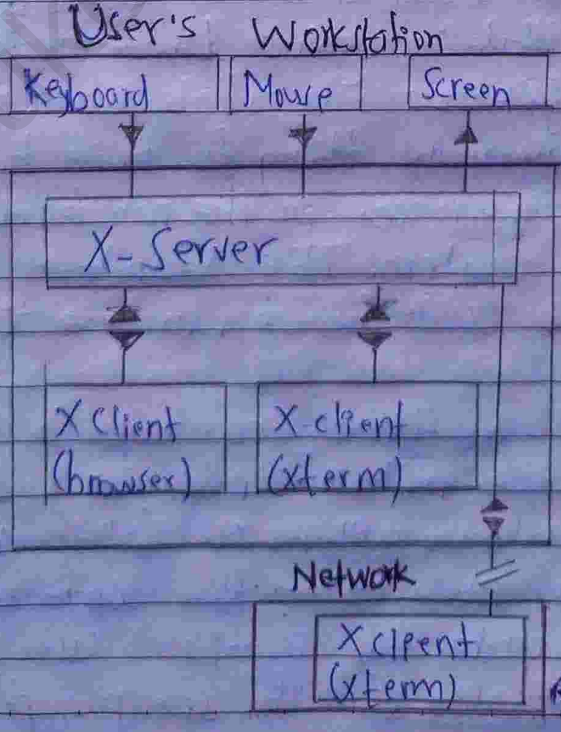


Fig: Architecture of the X Windows System

As in the figure, the X server receives input from a local keyboard and mouse and displays to a screen. A web browser and a terminal emulator run on the user's workstation and a terminal emulator runs on a remote computer but is controlled and monitored from the user's machine.

Run level or System initialization levels

A runlevel is a state of init and the whole system that defines what system services are operating. Runlevels are identified by numbers.

Some system administrators use run levels to define which subsystems are working, e.g. whether X is running, whether the networking is operational, and so on.

A run level in other words can be defined as a preset single digit integer for defining the operating state of your Linux or Unix based operating system.

The important thing to note here is that there are differences in the run levels according to the operating system. The standard Linux kernel supports these 7 different run levels:

- 0 → Turns off the device
- 1 → Single user mode (i.e. mode for administrative tasks)
- 2 → Multi user mode with no NFS (Network File System)
- 3 → Multi user mode under the command line interface (CLI) and not under the graphical user interface (GUI).
- 4 → User definable (i.e. for specific purposes).

.5 → Full mode (Multiple user mode under GUI and this is the standard runlevel for most of the Linux based systems).
.6 → Reboot which is used to restart the system/device.

By default, most of the Linux based system boots to runlevel 3 or runlevel 5. In addition to the standard runlevels, user can modify the present runlevels or even create new ones according to the requirement. Runlevels 2 and 4 are used for user defined runlevels and runlevel 0 and 6 are used for halting and rebooting the system.

Virtual Consoles (terminals)

Linux Console is the place where we can enter commands for system.

A virtual Console (VC) is also known as a Virtual Terminal (VT) is a conceptual combination of the keyboard and display for a computer user interface. It is a feature of some OS such as Linux, BSD, macOS etc. in which the system console of the computer can be used to switch between virtual consoles to access unrelated user interfaces.

In Linux, the user switches between them by pressing the Alt key combined with a function key - for example Alt + F1 to access the virtual console number 1. Alt + ← changes to the previous virtual console and Alt + → to the next virtual console.

.5 → Full mode (Multiple user mode under GUI and this is the standard runlevel for most of the Linux based systems).
.6 → Reboot which is used to restart the system/device.

By default, most of the Linux based system boots to runlevel 3 or runlevel 5. In addition to the standard runlevels, users can modify the present runlevels or even create new ones according to the requirement. Runlevels 2 and 4 are used for user defined runlevels and runlevel 0 and 6 are used for halting and rebooting the system.

Virtual Consoles (terminals)

Linux Console is the place where we can enter commands for system.

A virtual Console (VC) is also known as a Virtual Terminal (VT) is a conceptual combination of the keyboard and display for a computer user interface. It is a feature of some OS such as Linux, BSD, macOS etc. in which the system console of the computer can be used to switch between virtual consoles to access unrelated user interfaces.

In Linux, the user switches between them by pressing the Alt key combined with a function key - for example Alt + F1 to access the virtual console number 1. Alt + ← changes to the previous virtual console and Alt + → to the next virtual console.

Command Prompt

A command prompt is the input field in a text-based user interface screen for an operating system or program.

Command prompt is a command line interpreter application available in most Windows operating system. It is used to execute entered commands. Most of these commands automate tasks via scripts and batch files, perform advanced administrative functions and troubleshoots or solve certain kinds of Windows issues.

Linux Command Line

Linux Command Line is a text interface to your computer. Also known as shell, terminal, console, command prompts and many more, is a computer program intended to interpret commands.

It allows users to execute commands by manually typing at the terminal or has the ability to automatically execute commands which were programmed in "shell scripts".

Logging in and Logging out in Linux

Login tells the system who you are and what you have permission to do. Likewise, when you finish, you will log out so that no one else can access your files without permission.

Logging in the system directly from a (text) terminal:

If you do it in a text terminal, just type your

user name at the prompt; you will be asked for your password; type it and you are in the system.

Logging in the system directly from an X-window session:
To login in an X-window terminal you will find a small window with a "Logging" Type your user name, and then your password; you will then go to an X-window session.

Logging Out of a system from a (text) terminal
To log out (exit) of a text terminal, type logout.

Logging out from an X-Windows session
To exit an X-windows session, use the mouse; click on the right button of it and choose the "Logout" or "exit" option.

Linux Command Syntax

The standard Linux Command syntax is "Command [Option]" and then "<arguments>". The "Command [Option]" and "<arguments>" are separated by blank spaces.

A Linux command is usually an executable program residing on the Linux disc.

A typical syntax can look similar to this:

Command [-argument] [--long-argument] file

Files

In Linux system, everything is a file and if it is not a file, it is a process. A file doesn't include only text files, images and complicated programs but also include partitions, hardware device drivers and directories.

Linux consider everything as a file.

Types of Linux files:

1. Regular files → (-) : → It contains programs, executable files and text files.
2. Directory file (d) → It is shown in blue color - It contains list of files.
3. Special files
   - Block file (b)
   - Character device file (c)
   - Named pipe file (p)
   - Symbolic link file (l)
   - Socket file (s)

Creating files with Cat

The Cat is command is mainly used to read and Concatenate files but it can also be used for creating new files.

To create a new file, run the cat command followed by the redirection operator '>' and the name of the file you want to create. Press 'Enter', type the text and once you are done, press the 'CTRL+D' to save the files.

Syntax

```
$cat > file.txt
```

Displaying files' Contents with Cat

The most basic and common usage of the 'cat' command is to display/read the contents of files.

For example, the following command will display the contents of the '/etc/issue' file on the terminal:

```
$ cat /etc/issue
```

Deleting files with rm

To remove or delete a file in Linux from the command line, we either rm (remove) or unlink command.

To delete a single file:

Syntax

```
$ rm filename
```

To delete a multiple files:

Syntax:

```
$ rm filename1 filename2 filename3
```

To remove all files in current directory

Syntax

```
$ rm *.pdf  or  $ rm *.txt
```

Copying and Renaming files with cp and mv

Copying files with cp Command:

The cp command stands for copy and is used to copy files and directories in linux system.

i) To make a copy of file in current directory:
Syntax: cp <source> <destination>
Example: cp file1.txt file2.txt
Where
cp is represents copy command
file1.txt represents source file
file2.txt represents destination file

ii) To copy a file to another directory
```
cp file.txt /backup
```
We have copied the file 'file.txt' to the '/backup' directory.

iii) To copy multiple files to another directory
```
cp file1.txt file2.txt dir2
```
cp is copy command
file1.txt is source file and file2.txt is another source file.
dir2 is destination directory.

iv) To copy contents of one directory to another
```
cp dir1/* dir2
```

Where,
cp is Copy Command
dir1 represents source directory
dir1/* represents all contents of directory dir1
dir2 represents the destination directory

Renaming files with 'mv' command
The 'mv' command is used for moving as well as renaming a file/directory.
Syntax: mv <current_name> <new_name>
Example: mv original new
Where,
'mv' is move Command
'original' is current name.
'new' is new name.

Command History
In Linux, there is a very useful command to show you all of the last commands that have been recently used. The command is simply called history, but can also be accessed by looking at your bash_history in your home folder. By default, the history command will show you the last five hundred commands you have entered.
Syntax:
```
$ history
```

To see the last 10 commands:
```
history 10
```
or
```
history | tail
```
To remove whole history:
```
history -c
```

1.2 Files and Directories, Examples of absolute Paths, Current Directory, Making and Deleting Directories, Relative paths, Special Dot Directories, Using Dot Directories in Paths, Hidden Files, Paths to Home Directories, Looking for files in the System, Running Programs, Specifying Multiple files (with rm, mkdir, cat), Finding Documentation for Programs, Specifying files with Wildcards, Chaining Programs together, Graphical and Text Interfaces, Text Editor

Files and Directories

Linux store the data and programs in files. These are organized in a directory. In a simple way, a directory is just file that contain other files or directories.

The part of hard disk where we are authorized to save the data is called home directory. To find the home directory we need to type:

```
echo $home
```

Inside the directory, we can create and save files.

To create a file, 'touch' command is used:

Eg: `touch school.txt`

Examples of absolute paths

An absolute path is defined as specifying the location of a file or directory from the root directory (/). In other words, we can say absolute path is a complete path from the start of actual file system from root (/) directory.

Example of absolute path:

① /home/mukesh/manjay.txt
② /var/ftp/pub

Relative path:

Relative path is defined as the path related to the present working directory (pwd). It specifies where a document or directory is located concerning the current working directory.

Suppose I am located in /var/log and I want to change directory to /var/log/kernel. I can use relative path concept to change directory to kernel.

Changing directory to /var/log/kernel by using relative path concept:

pwd /var/logcd kernel

where:

Input: pwd/var/log cd kernel

Output: /var/log/kernel

Changing directory to /var/log/kernel using absolute path concept:

cd/var/log/kernel

Current directory

To get the current working directory, we the pwd command.

For example if we change the directory to "/home/user", pwd will print "/home/user" as the current working directory.

$ cd /home/user
$ pwd
/home/user

Making and deleting directories:

Making / creating directories:
The mkdir command creates new directories.
Syntax:
$ mkdir directory name

To create multiple directories:
$ mkdir dir1 dir2 dir3

Deleting / removing directories:

You can use rm command with the -r option to remove directories that contain files and sub directories.

$ rm -r dir1

The 'rmdir' command removes empty directories:
$ rmdir directories

Special dot directories, using dot directories in paths
Simply dot (.) is used to specify the directory location.
There are two types of dot directories:
① Single dot (.) → It specifies our current directory or can specify hidden file or folder.

② Double dot (..) → It specifies parent directory from our current location.

Example
$ pwd
path → /home/ved/ documents
$ cd. → single dot
$ pwd

path → /home/ved/ documents
$ cd.. → double dot
/home/pwd
/home/ved

Hidden files
On linux, hidden files are files that are not directly displayed when performing a standard ls directory listing.
Hidden files are also called dot (.) files as they begin with ".".
In most case, we need to see hidden files.

To display the hidden file in linux, we can use following commands:
"$ ls -a" → this will display all files implied folder
We can also use another command.

$ dir -a → this will display all the files, hidden files and implied folder.

Paths to Home Directories

This is the first place that occurs after logging into a Linux system. It is automatically created by "/home" for each user in the directory.

Generally, our terminal opens with the user's particular directory. To change directory to home directory, execute the cd command as follows:

```
cd /home
```

The above command will change the directory to home. To execute the home directory, execute the command as follows:

```
ls
```

Looking for files in the System:

Type 'find' ~~command~~ into the command line to track down a particular file by its name or extension.

① To find a file called 'thisfile.txt' in Linux, it will look for it in current and sub-directories:

```
find -name thisfile.txt
```

② Look for all .jpg files in the /home and directories below it:

```
find /home -name *.jpg
```

③ Look for an empty file in the current directory:

```
find -type f -empty
```

Running Programs

An instance of a running program is called process. The procedure to monitor the running process/program in Linux using the command line is as follows:

1) Open the terminal window on Linux.
2) For remote linux server use the ssh command for log in purpose.
3) Type the ps aux command to see all the running process in Linux.
4) Alternatively, you can use the top command or htop command to view running process in Linux.

Type the following ps command to display all the running process:

```
# ps -aux | less
```

Finding documentation for Programs

Use finddoc, it requires only the script name and the name of the program for which you need documentation.

For example:

```
$ ./finddoc grep
```

Specifying files with wildcards

A wildcard in linux, is a symbol or a set of symbols that stands in for other characters. It can be used to substitute for any other character or characters in a string. For example you can use a wildcard to get a list

of all files in a directory that begins with the letter O.

Three types of wildcards a common in Linux:

i) ? → It matches a single character.
For example, O??d matches anything that begins with O and ends with d, and has two characters between like Ond, Oerd etc.

ii) * → It matches any character or set of characters, including no character.
For example, O*d matches anything that begins with O and ends with d like, Oind, Oad.

iii) Bracketed values → It match character enclosed in square brackets.
For example, O[ac]d matches only Oad and Ocd.

Chaining Programs Together

Linux Command Chaining is the method of combining several different commands such that each of them can execute in succession based on the operator that separates them.

Some commonly used chaining operators are as follows:

(i) Logical AND (&&) → The command that follows this operator will execute only if the preceding command executes succesfully.

(ii) Logical OR (||) → The command succeeding this

operator execute only if the preceding command fails.

(iii) Semi-Colon (;) → The command following this operator will execute even if the command preceding this operator is not succesfully executed.

(iv) Pipe (|) → The output of the first command acts as the input to the second command.

(v) Ampersand (&) → This sends the current command to the background.

(vi) Redirection (>, <, >>) → Redirects the output of a command or group of commands to a file or stream.

(vii) NOT (!) → Negates the boolean expression with a command. It is much like an "except" statement.

Graphical and Text Interfaces

An interface is a shared boundary or connection between two dissimilar objects, devices or systems through which information is passed. The connection can be either physical or logical.

1) Graphical User Interface (GUI)

An interface that allows a user to interact with the system visually using icons, widgets, windows and more, is the Graphical User Interface (GUI).

The X windows System (or X) provides a graphical

user interface environment in Linux. It is a client-server windowing system.

2) Text-based User Interface (TUI)
Or, Terminal User Interface.
TUI is a retronym describing a type of user interface common as an early form of human-computer interaction, before the advent of GUIs.
Like GUIs, they may use the entire screen area and accept mouse and other inputs.
TUIs only use text and symbols available on a typical text terminal.

Text Editors
A text editor is a program used for editing text files. Most configuration of Linux system is done by editing text files.
Linux text editor can be used for editing text files, writing codes, updating user instruction files and more. A linux system supports multiple text editor.
There are two types of text editors in Linux:
a) Command-line text editors such as Vi, nano, pico etc.
b) GUI text editors such as gedit (for Gnome), KWrite etc.

Most common text editors are:
1) Vi/VIM editor
2) Nano editor
3) Gedit editor
4) Jed editor
5) Eclipse
6) Kate/KWrite
7) Sublime text editor
8) GNU emacs
9) Notepad ++
10) Pico

1.3 Shells, The Bash Shell, Shell Commands, Command-Line Arguments, Syntax of Command-Line Options, Examples of Command-Line Options, Setting Shell Variables, Environment Variables, Where Programs are found, Bash Configuration Variables, Using History, Reusing History Items, Retrieving arguments from the History, Bash Editing Keys, Combining Commands on One Line, Repeating Commands with for, Command Substitution, Finding Files with locate, Finding Files with find, find Criteria, find actions: Executing Programs

Shells

Shell is a program which provides the interface between the user and an operating system.

Shell is an environment in which we can run our commands, programs and shell scripts.

There are two major types of shells:

1. Bourne Shell → If you are using a Bourne-type shell, the $ character is the default prompt. It is denoted as bsh.
   It has following sub-categories:
   i) Bourne Shell (bsh)
   ii) Korn shell (ksh)
   iii) Bourne Again Shell (bash)
   iv) Posix shell (sh)

2. C Shell → If you are using a C-type shell, the % character is the default prompt.
   It has following sub-categories:
   i) C-shell (csh)
   ii) TENEX/TOPS C shell (tcsh)

The bash shell

In the bash shell, bash means Bourne Again Shell. It is a default shell over several distributions of Linux today. It is compatible to the Bourne shell (sh). It could be installed over Windows OS. It facilitates practical improvements on sh for interactive and programming use which contains:

- Job Control
- Command-line editing
- Shell Aliases and functions
- Unlimited size command history
- Integer arithmetic in a base from 2-64.

Shell Commands

In Linux, Commands are ways or instructions through which you can instruct your system to do some ~~more~~ action. Commands are executed in the Command line.

Syntax: Command [option] [argument]

1) Shell Command for displaying file contents on the terminal

a) cat → used to concatenate the files.

b) more → It is a filter for paging through text one screenful at a time.

c) less → It is used for viewing the files instead of

opening the file.

d) head → Used to print the first N lines of a file.

e) tail → Used to print the last N-1 lines of a file.

2) File and directory manipulating Commands

a) mkdir → Used to create a directory if not already existed.

b) cp → This command will copy the files and directories from the source path to the destination path.

c) mv → Used to move the files or directories.

d) rm → Used to remove files or directories.

e) touch → Used to create or update a file.

3) Extract, sort and filter data Commands

a) grep → Used to search for the specified text in a file.

b) sort → Used to sort the contents of files.

c) wc → Used to count the number of characters, words in a file.

d) cut → Used to cut a specified part of a file.

4) Basical Terminal Navigation Commands

a) Ls → To get the list of all the files or folders.
b) cd → used to change the directory.
c) du → show disk usage.
d) pwd → Show the present working directory.
e) man → Show the manual of any command present in Linux.
f) rmdir → Used to delete a directory if it is empty.
g) locate → Used to locate a file in Linux.
h) echo → Help us to move some data, usually text into a file.
i) df → used to see the available disk space in each of the partitions in your system.
j) tar → Used to work in the tarballs.

5) File Permission Commands

a) chown → Used to change the owner of the file.
b) chgrp → Used to change the group owner of the file.
c) chmod → Used to modify the access / permission of a user.

= Command line Arguments

Command line argument, can be defined as the input given to a command line to process that input with the help of given command.

Command line arguments are nothing but simple arguments that are specified after the name of the program in the system's command line, and these argument values are passed on to your program during program execution.

Arguments can be in the form of file or directory.

Arguments are entered in the terminal or console after entering command. They can be set as a path.
We can also write more than one argument together, they will be processed in the order they are written.

Syntax:
<Command> <argument>
<Command> <argument> <argument>

Example:
cd Downloads
ls Sample
cd/ home/sssit/Desktop
file javatpoint jtp.txt (With two arguments)

Syntax) Examples of Command line options

Command line options are commands used to pass parameters to a program.
Options are generally preceded by a hyphen (-) and for most commands, more than one option can be strung together, in the form:
Command - [option] [option] [option]
Eg.
ls -alR, will perform a long list of all files in the current directory.

Setting Shell Variables

A shell variable is a special type of variable that is set by the shell and is required by the shell in order to function correctly. Some of these variables are environment variables whereas other are local variables.

To define a variable in a shell, we use the variable name. A variable name may be any set of alphabetic character, including the underscore (_).

The name may also include a number, but number cannot be first character in the name. A name may not have any other types of character such as an exclamation point, an ampersand, etc.

Example:

```
$ TEST_VAR='Hello World!'
```

Environment Variables

Environment variables are variables that contain values necessary to set up a shell environment. Environment variables are defined for the current shell and are inherited by any child shells or processes.

Environment variables are dynamic values which affects the processes or programs on a computer. They exist in every OS but types may vary. Environment variables can be created, edited, saved, and deleted and give information about the system behaviour.

To set an environment variable, the 'export' command is used.

Syntax:
export NAME=VALUE

Example
export NEW_VAR="Testing export"

Where Programs are found

The best method for finding linux programs is the 'whereis' command.

The 'whereis' command can find the source code, manuals and the location of a program.

Syntax: Example

```
whereis telnet
```

The output will be:

```
telnet: /usr/bin/telnet /usr/bin/telnet.netkit ...
```

If you just want find the location of the program, you can use switch -b as follows:

```
whereis -b telnet
```

This returns the following result:

```
telnet: /usr/bin/telnet /usr/bin/telnet.netkit...
```

Bash Configuration Variables

The easiest way to set environment variables in bash is to use the "export" keyword followed by the variable name, an equal sign and the value to be assigned to the environment variable.

For example, to assign the value "abc" to the variable "VAR", you would write the following command.

```
$export VAR=abc
```

If you want to have spaces in your value, such as "my value" (you will have to enclose your value in double quotes.

```
$ export VAR="my value"
```

Using History

In its simplest form you can use history command by just typing its name:

Syntax

```
history
```

To see a certain number of commands, you can pass a number to history on the command line.

Syntax:

```
history 10
```

To remove the entire bash history run the following command in terminal:

Syntax

```
history -cw
```

~~# Reusing history items~~

Bash editing keys

'ctrl + L' → Clear the screen, similar to the clear command.

'ctrl + _' → Undo

'ctrl + x, ctrl + e' → Edit the current command in

your Selector.
Alt + Del → Delete the word before the cursor.
Alt + d → Delete the word after the cursor.
ctrl + d → Delete the character under the cursor.
ctrl + h → Delete the character before the cursor (like backspace)
ctrl + k → Cut the line after the cursor to the clipboard.
ctrl + u → Cut the line before the cursor to the clipboard.
ctrl + d → Cut the word after the cursor to the clipboard.
ctrl + w → Cut the word before the cursor to the clipboard.
ctrl + y → Past the last item to be cut.
Alt + t → Swap the current word with previous.
Ctrl + t → Swap the last two characters before the cursor.
Esc + t → Swap the last two words before the cursor.

Combining Commands on one Line

The semicolon (;) operator allow you to execute multiple commands in one line. For this, type the following the following three commands in one line, separated by semicolons and press Enter.

```
ls ; pwd ; whoami
```

This will give you a listing of the current directory (ls), find out which directory you are in currently in (pwd), and display your login name (whoami) all at once.

Repeating Commands with for

1) Wrap your statement (for i in (1..n) ; do someCommand; done), where n is a positive number and someCommand is any command.

2. To access the variable, you need to wrap like this: $(i) e.
3. Execute the statement by pressing the Enter key.

Syntax/ Example
Use the following syntax to run some command multiple times:

```
for i in {1..5}; do echo "I am student $i" ; done
```

OR

```
for ((i=0;i<5;i++)); do echo "I am student $i" ; done
```

Command Substitution

Command substitution is the mechanism by which the shell performs a given set of commands and then substitutes their output in the place of the commands.

Syntax

```
$(Command)  or  `command`
```

Example

```
$ echo "Current path is [`pwd`]"
Current path is [/home/baeldung]
```

In this example, the output of the pwd command replaces the actual command.

Finding files with locate

To use locate, open a terminal and type the locate followed by the file name you are looking for.

Example

```
locate sunny
```

In this example, I'm searching for files that contain the word 'sunny' in their name.

Finding files with find, find criteria

The 'find' command in linux can be used to find files and directories and perform subsequent operations on them.

Syntax

```
find command -options starting/path expression
```

Examples

```
find. -name testfile.txt
```
→ finds a file called testfile.txt in current and sub-directories.

```
find /home -name*.jpg
```
→ find all .jpg files in the /home and sub-directories.

find actions: Execution Programs

1.4 Process Text Streams Using Text Processing with less, wc, sort, uniq, cut, expand, fmt, head, tail, nl, cat, od, pr, split, tac, tr command

A text stream is any information printed to standard output (usually the screen). For example, the ls command sends the list of files to the screen. The following commands intercept and process the text stream in various manners:

i) less Command

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

Syntax:

```
less file_name
```

ii) wc Command

wc stands for Word Count.

It is used to find out number of lines, word count, byte and character count in the file specified in the file arguments. By default, it displays four column output.

Syntax:

wc [option] ... [File] ...

wc -l → print number of lines

wc -w → print number of word

wc -m → number of character

wc -c → display count of bytes present in line

Example: wc -m mukesh.txt

iii) Sort Command →

Sort command is used to sort a file, arranging the records in a particular order.

By default, sort command sorts the file assuming the contents are ASCII.

Sort Command sorts the content of text file line by line.

Example: Suppose we create a data file with name file.txt command:

```
$ cat > file.txt
Ajay
Tausif
Mukesh
Daya
```

Sorting a file: Now use the sort command:

Syntax:

Command

```
$ sort file.txt
```

Output

```
Ajay
Daya
Mukesh
Tausif
```

iv) uniq Command

The 'uniq' command in linux reports or filters out the repeated lines in a file.

Syntax:

$ uniq [option] [input] [output]

Example:
$ cat Sukuna.txt
book
book
Copy

Output
$uniq Sukuna.txt
book
Copy

v) Cut Command
It is a command for cutting out the sections from each line of files and writing the result to standard output.
It can be used to cut parts of a line by byte position, character and field.
Syntax:
[Cut [Option]... [File]]

Example:
$ Cat school.txt
Andra Pradesh
Assam
Bihar
Kerala
Punjab

Output (To extract specific bytes)
$cut -b1,2,3 school.txt
And

Ajanta
Page No.
Date

Ass
Bih
Ker
Pun

vi) ~~Expand Command~~ expand Command

'expand' command is used to convert tabs into spaces in a file and when no file is specified it reads from standard input.

Example:

@ expand Myfile.txt

Expands the file 'myfile.txt', changing tabs to space, and display on standard one.

ⓑ expand -tabs=10 myfile.txt > myfile2.txt

Convert the tabs in the file myfile.txt to 10 spaces each, and write the output to myfile.txt.

vii) fmt Command

fmt command in linux actually works as a formatter for simplifying and optimizing text files.

Syntax:

[$fmt [-width] [option] .. [File]]

fmt by default, with no option formats all the words present in the given file in a single line.

Example

$cat it.txt
hello
everyone
Have

a
nice
day

/* fmt by default puts all words in a single line and prints on stdout. */

$fmt ft.txt
hello everyone. Have a nice day.

Option for fmt command:
-w .. width = width option
Ex:
$ cat ft.txt
hello everyone. Have a nice day

$fmt -w 10 ft.txt
hello ever
yone. Have
a nice day.

viii) head Command
'head' is used to display the first part of a file, it outputs the first 10 lines by default. You can use the -n num flag to specify the number of lines to be displayed.

Syntax Example
$ head /var/log/auth.log

ix) tail Command

tail output the last part (10 lines by default) of a file. Use the -n num switch to specify the number of lines to be displayed.

Syntax

tail [option] .. [File]

Example

$ cat Sukuna.txt

Apple
Ball
Cat
Dog
Egg
fish
Girl
Hen
Iron
Jug
kite
Lion
Monkey
Nest
Orange

Without any option, it display only the last 10 lines of the file specified.

$tail Sutuna.txt
Fish
Girl
Hen
Iron
Jug
Kite
Monkey
Nest
Orange

With option:
$tail -n3 Sutuna.txt
Monkey
Nest
Orange

x) nl Command
It is also a text formatting command. It also number the line in a file.

Syntax:
nl [option] files

Example →
~$n file n3

xi) Cat Command
Cat command in Linux is used to print the content of a file onto the standard output stream. It also allows to write some texts into a file.

Syntax:

Cat [option]... [file]

x:j) od Command

Od Command in Linux is used to convert the contents of input in different formats with octal format as the default format.

Syntax

od [option]... [file]

Example

input: $cat input.txt

100
101
102
103
104
105

output: od -b input.txt

```
0000000 061 060 060 012 061 060 061 012 061
060 062 012 061 060 063 012
0000020 061 060 064 012 061 060 065 012 012
```

The first column in the output of od represents the byte offset in file.

(xiii) pr Command

pr command is format of a file to make it look better when printed. It converts text file for printing.

Syntax: [pr.. [option] ... [file]]

Example: pr -v mukesh.txt

(xiv) Split Command

split command is used to split or break a file into the pieces in linux system. Whenever we split a large file with command then split output file default size is 1000 lines and its default prefix would be 'x'.

Syntax:

split [option] [filename] [prefix]

Example

① split Mukesh.txt

② split -b =5 makesh.txt

xv) tac Command

tac command is used to concatenate and print files in reverse.

This command will write each file to standard output the last line first.

Syntax:

[tac [option] ... [file] ..]

Example
It will print files in reverse
$ tac Sukuna.txt
five
four
Three
Two
One

xvi) tr Command
It is for translating or deleting characters.

Syntax
$ tr [option] -- [redacted] : Set1 [Set 2]

Example
① Convert lower case to Upper case
$ cat greekfile
Output:
WELCOME TO
GeeksforGeeks

② Translate whole space to tabs
$ echo "Welcome To Sukuna" | tr [space] '\t'
output
Welcome To Sukuna

1.5 Perform Basic File Management : File System objects, Directory and file names, file Extensions, wildcard patterns, cp, mv, rm, mkdir, rmdir, touch command, Searching files with grep

Basic file management refers to a way to name, save, backup, organize files/folders and keep track of files on a computer.

File management objects

A filesystem is an organization of data and metadata on a storage device.
Linux views all the file system from the perspective of common set of objects. These objects are the superblock, inode, dentry and file. At the root of each file system is the superblock, which describes and maintains state for the file system. Every object that is managed within a file system is represented in Linux as an inode.

Directory and filenames

A linux system makes no difference between a file and a directory, since a directory is just a file containing names of other files.
Rules for naming files and directory names in Linux:

1) All file names are case sensitive. So filename vivek.txt and VIVEK.txt all are different files.

2) You can use upper and lower case letters, numbers, dot (.) and underscore (_) symbols. filenames

3) beginning with dot(.) are hidden files.

3) You can use other special characters such as blank space but they are hard to use and it is better to avoid them.

4) Filenames may contain any character except "/(root directory)", which is reserved as the separator between files and directories in a pathname.

5) The filenames must be unique inside their directory. You can have files or folders with the same names, but in different locations.

6) The new Linux system limit the filenames to 255 characters (255 bytes).

File extensions

Linux doesn't use file extensions; rather, the file's type is part of the file name. To find out the true file type, use the 'file' command.

Wildcard Patterns

A wildcard pattern is a series of characters that are matched against incoming character strings. You can use these patterns when you define pattern matching criteria. Matching is done strictly from left to right, one character or basic wildcard pattern at a time.

"Cp, mv, rm, rmdir, mkdir : touch command, searching files with grep" have are already discussed in the previous topics.

1.6 Job Control and Processes : Job Control, jobs, fg, bg, Process, Process Properties, Parent and Child Processes, ps, ps Options, process monitoring with pstree, top, signaling processes, Common Signals for interactive use, Sending Signals : kill, Sending Signals to Daemons : pidof, Modifying process execution priorities with nice and renice, Job Scheduling with Crond, crontab, anacron and System log

#Job Cntrl ⇒ Job Control refers to the ability to selectively stop (suspend) the execution of processes and continue (resume) their execution at a later point.

#Jobs:

Job is a process that the shell manages. Job means an application program and it is not a system program.

Since job is a process, each job has an associated PID.

There are three types of job statuses:

1) Foreground (fg) ⇒ When you enter a command in a terminal window, the command occupies that terminal window until it completes. This is a foreground job.

2) Background (bg) → When you enter an ampersand (&) symbol at the end of a command line, the command runs without occupying the terminal window. The shell prompt is displayed immediately after you press Return. This is an example of background job.

3) Stopped → If you press 'Ctrl+Z' for a foreground job, or enter the stop command for a background job, the job stops. This job is called a Stopped job.

Note: Except the Bourne shell, the other shells support job Control.

#Job Control Commands

Job Control Commands enable you to place jobs in the foreground or background, and to start/stop jobs. The table describes the job control Commands:

Option	Description
jobs	List all jobs
bg %n	Places the current or specified job in the background, where n is the job ID.
fg %n	Brings the current or specified job into the foreground, where n is the job ID.
Control-Z	Stops the foreground job and places it in the background as a stopped job.

Process

Any running program or command in linux is called process.

A process can be run in 2 ways:

1) foreground Process → Every process when started runs in foreground by default, receives input from the keyboard, and sends output to the screen.

Syntax:
```
$ ls pwd
```
output:
```
$ /home/process/root
```

2) Background Process → It runs in the background without keyboard input and waits till keyboard input is required. Thus, other processes can be done in parallel with the process running in the background since they do not have to wait for the previous process to be completed.

Adding & along with the command starts it as a background process.

Syntax
```
$ pwd &
```

Output
```
[1] + Done pwd
$
```

Process Properties

Process properties refer to process characteristics such as data set-size, kernel scheduling priority, the number of pages of memory, and the number of

page fault:

A process has a series of characteristics, which can be viewed with the 'ps' command:

1) The Process ID or PID → a unique identification number used to refer to the process.

2) The Parent Process ID or PPID → the number of process (PID) that started this process.

3) Nice number → the degree of friendliness of this process towards other processes.

4) Terminal or TTY → terminal to which the process is connected.

5) User name of the real and effective user (RUID and EUID) → the owner of the process.

6) Real and effective group owner (RGID and EGID)
→ The real group owner of a process is the primary group of the user who started the process. The effective group owner is usually the same, except with SGID access mode has been applied to a file.

Parent and Child Processes

Parent Process

A parent process is one that creates a child process using a fork() system call. A parent process may have multiple child processes, but a child

process has only one parent process.
On the success of a fork() system call:
- 0 is returned to the child process.

On the failure of a fork() system call:
- -1 is returned to the parent process, where child process is not created.

Child Process:-
A child process is a process created by a parent process in OS using a fork() system call. A child process may also be called a subprocess or a subtask.
A child process is created as its parent process' copy and inherits most of its attributes. If a child process has no parent process, it was created directly by the kernel.

ps, ps options
ps stands for process status. It is used to list the currently running processes and their PIDs along with some other information that depends on different options.
It reads the process information from the virtual files in /proc file-system. /proc contains virtual files, this is the reason its referred to as a virtual file system.

ps options:

ps provides numerous options for manipulating the output according to our need.

Syntax:

ps [options]

options for ps Command:

1) To view your process:

```
ps -ux
ps -e
```

2) Simple process selection : ps

3) View process not associated with terminal : ps -a

4) View all process except session leaders : ps -d

5) View all process except those that fulfill the specified conditions :

```
ps -a -N
ps -a --deselect
```

6) View all process associated with the terminal : ps -T

7) View all the running process : ps -r

8) View all Processes owned by you : ps -x

Process monitoring with pstree

pstree is a convenient linux command used to show running processes in a tree (data structure). If a username is specified, all process trees rooted at processes owned by that user are shown. pstree is used as an alternative to ps command.

pstree command example:

The basic syntax for pstree is:

```
pstree [options] [pid or username]
```

To display all process trees rooted at processes owned by a specific user with their PIDs :-

```
pstree -p username
```

To display a tree of Processes:

```
pstree
```

To display a tree of processes with PIDs:

```
pstree -p
```

To sort processes by PID instead of by name:

```
pstree -np
```

To wrap long lines:

```
pstree -l
```

top Command

The 'top' command is used to show the active Linux processes. It provides a dynamic real-time view of the running system. Usually, this command shows the summary information of the system and the list of processes or threads which are currently managed by the Linux Kernel.

Examples

1) Exit top command after specific repetition:

```
top -n 10
```

2) Display specific user process:

```
top -u paras
```

3) Highlight running process in top:

```
top -z
```

4) Shows absolute path of process:

```
top -c
```

5) Secure mode:

```
top -s
```

Signaling Process

A signal is an event generated by the Linux systems in response to some condition. Upon receipt of a signal, a process may take action.

A signal is just like an interrupt; when it is generated at the user level, a call is made to the Kernel of the OS, which then acts accordingly.

There are two types of signals:
1) Maskable → signals which can be changed or ignored by the user (e.g. ctrl + C).

2) Non-maskable → signals which cannot be changed or ignored by the user.

Common Signals for Interactive use

The following are the common signals you might encounter and want to use in your programs:

Signal Name	Signal Number	Description
SIGHUP	1	Hang up or shut down and retart process.
SIGINT	2	Interrupts a process (signal (ctrl + C)
SIGQUIT	3	Sends a quit signal (Ctrl + D)
SIGFPE	8	An illegal mathematical operation is attempted.
SIGKILL	9	Quits process and not perform any clean-up operations.
SIGALRM	14	Alarm clock signal (used for timers)
SIGTERM	15	Software termination signal.

Sending signals: kill

The other common method for delivering signals is to use the 'kill' command.

Syntax

~~$ signal~~ $ kill -signal pid

Here, signal is either the number or name of the signal to deliver and 'pid' is the process ID

that the signal should sent to.
for example:

```
$kill -1 1001
```

The above command sends the Hup or Hang-Up signal to the program that is running with process ID 1001.
To send a kill signal to the same process use:

```
$kill -9 1001
```

This kills the process running with process ID 1001.

Sending Signals to Daemons : pidof

The pidof command is used to find out the PIDs of a specific running program. pidof is a simple command that doesn't have a lot of options.

Syntax

```
pidof [options] Program-Name
```

Examples

1) To find pid of any process:

```
pidof bash
```

2) To get only one pid of a program:

```
pidof -s bash
```

3) To find pid of a program and kill it:

```
p=$(pidof chrome)
kill $p
```

Modifying process execution priorities with nice and renice

'nice' command in Linux helps in execution of a program/process with modified scheduling priority. It launches a process with a user-defined scheduling priority.

Whereas 'renice' command allows you to change and modify the scheduling priority of an already running process.

Examples of nice and renice command:

1) To set the priority of a process:

`nice -10 gnome-terminal` This will set the nice value of mentioned process.

2) Changing the priority of the running process:

`sudo renice -n 15 -p 77982` This will change the priority of the process with pid 77982.

Job Scheduling with cron, crontab, anacron ~~and systemtg~~

Job scheduling is a feature that allows a user to submit a command or a program for execution at specified time in the future.

* Cron Command

The cron is a software utility, offered by

Linux that automates the scheduled task at a predetermined time.

It is a daemon process, which runs as a background process and performs the specified operations at the predefined time when a certain event or condition is triggered without the intervention of a user.

Syntax:

```
crond [-f] [-l] [-l loglevel]
```

* Crontab Command

The crontab (abbreviated for "cron table") is a list of commands to execute the scheduled tasks at a specified time.

It allow the user to add, remove or modify the scheduled tasks.

Linux Crontab format:

```
MIN HOUR DOM MON DOW CMD
```

* Anacron Command

'anacron' command is used to execute commands periodically with a frequency specified in days. Its main advantage over cron is that it can be used on a machine which is not running continuously.

Syntax

```
anacron [-s] [-f] [-n] [-d] [-q] [-t anacrontab] [-S spooldir]
        [job]
```

System log

System logs record events that occur within the operating system itself, such as driver errors during start-up, sign-in and sign-out events and other activity.

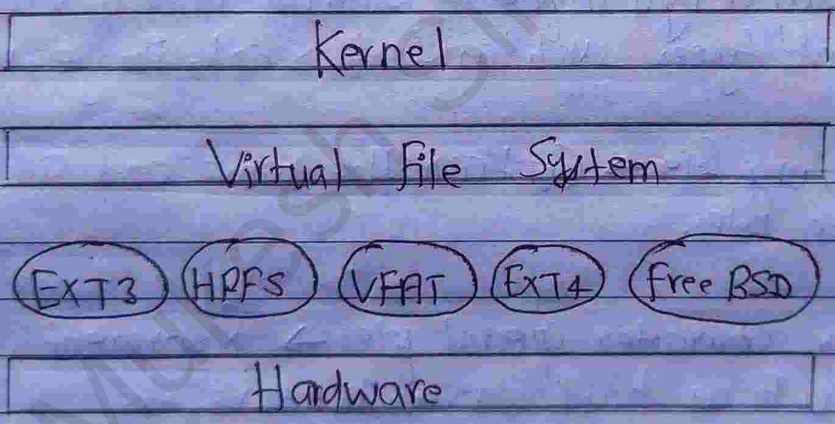
1.7 Filesystem Concepts: Filesystems, The Unified Filesystem, Filetypes, Inodes and Directories

Filesystems

A Linux filesystem is a structured collection of files on a disk drive or a partition. A partition is a segment of memory and contains some specific data. The Linux filesystem contains the following sections:

- The root directory (/)
- A specific data storage format (EXT3, EXT4, BTRFS, XFS and so on).
- A partition or logical volume having a particular filesystem.

Linux filesystem structure:



The Unified Filesystem

The Linux filesystem unifies all physical hard drives and partitions into a single directory structure. It all starts at the top - the root (/) directory. All other directories and their sub-directories are located under the single Linux root directory. This means that there is only one single directory tree in which to search

for files and programs.
This can work only because a filesystem such as /home, /tmp, /var, /opt, or /usr can be created on separate physical hard drives, a different partition, or a different logical volume from the (/) root filesystem and then be mounted on a mountpoint (directory) as part of the root filesystem tree.

File Types

A filetype help us in identifying the type of content saved in the file. Linux supports seven different types of files. They are:

1) Ordinary or regular files → Contain data of various content types such as text, script, image, video etc.

2) Directory files → Contain the name and address of other files.

3) Block or character special files → Represent device files such as hard drives, monitors, etc.

4) Link files → Point or mirror other files.

5) Socket files → Provide inter-process communication.

6) Named pipe files → Allow process to send data to other process or receive data from other process.

How to identify the type of a file?

There are many ways to identify the type of a file in Linux. The easiest way is to use the 'file' command. To find the type of file, specify the name of that file as an argument. For example, to know the file type of the file abc, we the following command:

```
$ file abc
```

Inodes and directories

The inode (index node) is a data structure in a Linux file system that describes a file system object such as a file or a directory. Each inode stores the attributes and disk block locations of the objects data. A directory is a list of inodes with their assigned names.

In Linux, just like a file, a directory has an inode. Rather than pointing to disk blocks that contain file data, a directory inode points to disk blocks that contain directory structures.

Compared to an inode, a directory structure contains a limited amount of information about a file. It only holds the file's inode number, name and the length of the name.

The inode and directory structure contain everything you need to know about a file or directory. The directory structure gives us the filename and inode number. The inode tells us everything else about the file, including timestamps, permissions, and where to find the file data in the file system.

You can see the inode number of a directory just as easily as you can see them for files:

Example

```
ls -lid work/
```

Here we use ls with the -l (long format), -(inode) and -d (directory) options, and look at the 'work' directory.

1.8 Create and Change Hard and Symbolic Links, Working with Compression Utilities, Archiving file

Create and Change Hard and Symbolic Links

A link is simply an additional directory entry for a file or directory, allowing two or more names for the same thing.

1) Hard links

A hard link is a directory entry that points to an inode. You can create hard links only for files and not for directories. The exception is the special directory entries on a directory for the directory itself and ~~parent~~ for its parent (. and ..), which are hard links and maintain the count of the number of sub-directories.

To create a hard link in Linux, we ~~shall~~ will use an utility. For example, the following command creates a hard link named 'tp' to the following 'topprocs.sh'.

```
$ ls -l
$ ln topprocs.sh tp
$ ls -l
```

2) Soft links:

Soft link or symbolic link is a directory entry that points to an inode that provides the name of another directory entry. Symbolic/Soft links are also called Symlinks.

Soft links merely point to ~~the~~ another file or

directory by name rather than by inode. Soft links can cross filesystem boundaries. Deleting the soft link does not delete the target file or directory, and deleting the target file or directory does not automatically remove any soft links.

To create a soft link in Linux, we will use same ln utility with '-s' switch. For example, the following command creates a symbolic link named 'topps.sh' to the file 'toppstas.sh'.

```
$ ln -s ~/bin/toppstas.sh topps.sh
$ ls -l topps.sh
```

Hard link	Soft (Symbolic) link
1. A hard link acts as a copy of the selected file.	1. A soft link acts as a pointer or a reference to the file name.
2. It acces the data available in the original file.	2. It does not acces the data available in the orginal file.
3. Files with hard link take the same inode number.	3. Files with soft link take a different inode number.
4. Hard links are not allowed for directories.	4. Soft links can be used for linking directories.
5. It cannot be used across file systems.	5. It can be used across file systems.
6. Hard links are comparatively faster.	6. Soft links are comparatively slower.

Working with Compression Utilities

A Compression utility or Compression program is a software program that compresses and decompresses various file types.

Transferring files between one computer to another or storing them securely is a major task to both normal and professional users. Sometimes it is not possible to send files above a certain size over the Internet. So, you need utilities that will help decrease your file size without compromising data or its quality. It also helps merge multiple files and reduce the overall file size to help you send it securely over the Internet.

Linux users are blessed with many effective and reliable file compression utilities at their disposal. The major compression utilities are discussed below:

1) tar

The 'tar' file compression is one of the most widely used file compression utilities on Linux. File compressed with this utility have suffix .tar.gz and .tgz, and they are also called tarballs.

Example command:

```
$ tar -czvf file.tar.gz swap1
```

2) gzip

The gzip stands for GNU zip, and it is an open source file compression format used to compress single file. It produces zipped files with the suffix '.gz' extension.

Syntax

```
$ gzip [options] [filenames]
```

3) 7zip

The 7zip is an open source file compression utility that was initially developed for Windows users and was later ported to other systems like linux. It supports multiple file compression formats and is popular for high compression ratio with LZMA and LZMA2 compression techniques.

Syntax

```
$7z a filename.7z filename
```

To install 7zip:

```
$sudo apt-get install p7zip full p7zip-rar
```

4) lzma

The lzma is another file compression utility like zip or tar, and it ships-in pre-installed with linux and its distors. It is a quite fast file compression utility as compared to others.

To create archive:

```
$lzma -c --student filename> filename.lzma
```

5) bzip2

The bzip2 is a free and open source file compression utility. It is a faster file utility as compared to gzip but can only compress a single file at a time.

Syntax

```
$ bzip2 filename
```

6) xz file compression

The xz file compression utility is an upgrade

to the lzma file utility but can only compress a single file at a time. It integrates well with all the Linux distros, even the older releases.

Syntax

```
$ rz filename
```

7) Shar

The shar, short for "shell archive", is a simple and reliable file compression utility for personal and power users.

Syntax

```
$ shar filename > filename.shar
```

Archiving files

An archive is a single file that contains a collection of other files and directories.

Archive files are typically used for a transfer or make a backup copy of a collection of files and directories which allow you to work with only one file instead of many. Likewise, archives are used for software application packaging. The single file can be easily compressed for ease of transfer while the files in the archive retain the structure and permissions of the original files.

We can use 'tar' tool to create, list and extract files from archives. Archives made with tar are normally called "tar files", "tar archives" or since all the archived files are rolled into one - "tarballs". (Tar stands for Tape archive)

Creating Archive files:

1) To create an archive called 'project.tar' from the contents of the 'project' directory, type:

```
$tar -cvf project.tar project
```

ii) To create a compressed archive called 'project.tar.gz' from the contents of the 'project' directory, type:

```
$tar -zcvf project.tar.gz project
```

1.9 Manage file Ownership Permissions: Users and groups, The Superuser: Root, Changing file ownership with chown, Changing file group ownership with chgrp, Changing the ownership of a directory and its contents, Changing Ownership and Group ownership simultaneously, Basic Concepts: Permission on files, permissions on directories, Permissions for different groups of people, Examining Permissions: ls -l, Preserving Permissions when Copying Files, Changing file and directory permissions: chmod, Specifying permissions for chmod, Special Directory Permissions: 'sticky' and setgid, Special File Permissions: Setgid and Setuid.

Manage file ownership and Permissions: User & groups

The Concept of linux file Ownership and permission is crucial in Linux. Here, we will explain linux ownership and permissions and will discuss both of them.

* Ownership of Linux files:

Every file and directory on your linux system is assigned 3 types of owner, given below:

1) User →

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

2) Group

A user-group can contain multiple ~~groups~~ users. All users belonging to ~~the~~ a group will have the same Linux group permissions acces to the file: ~~Suppose you have a~~

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

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

1) Read.
This permission give you the authority to open and read a file. Read permission on a directory gives you the ability to list its content.

2) Write
The write permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory.

3) Execute
In Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code but not run it.

The Super user : root

The Superuser also known as the root user or administrator, is a special user account in linux used for system administration. It is the most privileged user on the linux system and it has access to all commands and files. The root user can do many things an ordinary user cannot, such as installing new software, changing the ownership of files, and managing other user accounts. It is not recommended to use 'root' for ordinary tasks, such as browsing the web, writing texts etc.

Changing file ownership with chown

The chown (stands for change owner) command is used to change the ownership of a file in Linux. In its most basic form, you just provide the name of the new owner and the filename.

Syntax :

```
chown new_owner filename
```

Example:

```
chown mukesh linux
```

Changing file group ownership with chgrp

'chgrp' command in linux is used to change the group ownership of a file or a directory. All files in linux belong to an owner and a group.

Syntax

```
$ chgrp group filename
```

Use the following procedure to change the group ownership of a file:

→ Check the current file ownership using ls -dl:
$ ls -dl test

→ Change the file owner to root - You will need sudo.
$ sudo chgrp root test

→ Group ownership changed to root:
$ ls -dl test

To change the group ownership of a folder:
Sudo chgrp geeksforgeeks GFG

Changing the group ownership of a directory and its contents

→ Use 'chown' to change ownership and 'chmod' to change rights.
→ Use the -R option to apply the rights for all file inside of a directory too.
→ Note that both these commands just work for directories too. The -R option makes them also change the permission for all files and directories inside of the directory.

Example: Syntax
Sudo chown -R username : group directory

Example

```
sudo chown -R mukesh Sukuna
```

It will change ownership of all files and directories inside of directory and directory itself.

Changing Ownership and Groupo ownership Simultaneously

To simultaneously change both the owner and group of files or directories in the linux use the following command structure:

Syntax

```
chown somerusername : somegroupname filename
```

Example

```
chown mukesh : sukuna linux
```

Basic Concepts : Permission of files, permission on directories, Permission for different groups of people

We have already discussed basic concepts in the first page of 1.9.

Changing file/directory Permission with 'chmod':

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

Syntax

```
Chmod permission filename
```

Examining Permissions: ls -l

We can view the permissions by checking the file or directory permission in our favourite GUI file manager or by reviewing the output of the "ls -l" command while in the terminal and while working in the directory which contains the file or folder.

The -l option signifies the long list format. The shows a lot more information presented to the user than the standard command. You will see the file permissions, the number of links, owner name, owner group, file size, time of last modification, and the file or directory name.

Preserving permissions when copying file

The standard 'cp' command has all you need to retain file permission while copying. You can use the -p option of cp to preserve the mode, ownership, and timestap of the file.

```
cp -p source_file dest_file
```

However you will need to add the '-r' option to the command when dealing with directories. It will copy all sub-directories and individual files, keeping their original permissions intact.

```
cp -rp source_dir/ dest_dir/
```

Changing file and directory permissions: chmod

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

Syntax

```
chmod permission filename
```

Specifying Permissions for chmod

Permissions for owner of the file:
r = read, w = write, x = execute

- chmod +rwx filename to add permissions.
- chmod +-rwx directoryname to remove permissions.
- chmod +x filename to all executable permissions.
- chmod -wx filename to take out write and execute permissions

Permissions for group owners is similar but add "g" for group or "o" for users.

- chmod +g+w filename
- chmod g=wx filename
- chmod o+w filename
- chmod o-rwx foldername

Permission for everyone, use "ugo" or "a" (for all).

- chmod +ugo+rwx foldername to give read, write and execute permission to everyone.

• chmod a=r foldername to give only read permission for everyone.

Special directory permissions: "Sticky" and "Setgid"
Special file permissions: Setgid and Setuid

Three special types of permissions are available for executable files and public directories. When these permissions are set, any user who runs that executable file assumes the user ID of the owner (or group) of the executable file.

1) Sticky Bit

The Sticky Bit is a permission that protects the files within a directory. If the directory has the sticky bit set, a file can be deleted only by the owner of the file, the owner of the directory, or by root. This special permission prevents a user from deleting other users' files from public directories such as /tmp.

```
drwxrwxrwt 7 root sys 400 Sep 27 15:21 tmp
```

2) Setgid

The set group identification (setgid) permission is similar to setuid, except that the process's effective group ID (GID) is changed to the group owner of the file, and a user is granted access based on permissions granted to that group.
The /usr/bin/mail command has setgid permissions.

```
-r-x--s--x 1 root mail 63628 Sep 29 15:25 /usr/bin/mail
```

3) Setuid

When set-user identification (Setuid) permission is set on a executable file, a process that runs this file is granted access based on the owner of the file (usually root) rather than the user who is running the executable file. This special permission allows a user to access files and directories that are normally only available to the owner.

```
-r-sr-sr-x 3 root sys 10458O Sep29 15:29 /usr/bin/passwd
```

1.10 Create and Control Partitions and Filesystems: Concepts: Disks and Partitions, Disk Naming, Using fdisk, Making New Partitions, Changing Partition Types, Making filesystems with mkfs; Mounting filesystems, Mounting a filesystem: mount; Mounting other filesystems, Unmounting a filesystem; Unmount; Configuring mount: /etc/fstab; Filesystem Types, Mount options, Mounting/unmounting a file, a flash drive, cd drive.

Create and Control Partitions and Filesystems Concepts

In Linux, when you create a hard disk partition or a logical volume, the next step is usually to create a filesystem by formatting the partition or logical volume. The how-to assumes you know how to create a partition or a logical volume, and you just want to format it to contain a filesystem and mount it.

Disk Partition

Disk partitioning is the process of dividing a disk into one or more logical areas, often known as partitions, on which the user can work separately.

Partitioning enables you to split your hard drive into multiple parts, where each part acts as its own hard drive and this is useful when you are installing multiple operating system in the same machine.

Partition in Linux are usually created during the installation.

Disk Naming

Linux disks and partition names may be different from other operating systems. You need to know the name that Linux use, when you create and mount partitions. Here's the basic naming scheme:

- The first floppy disk drive is named /dev/fd0.
- The second floppy disk drive is named /dev/fd1.
- The first SCSI disk (SCSI ID address-wise) is named /dev/sda.
- The second SCSI disk (address-wise) is named /dev/sdb / and so on.
- The first SCSI CD-ROM is named /dev/scd0 #, also known as /dev/sr0.
- The master disk on IDE Primary Controller is named /dev/hda.
- The slave disk on IDE Primary Controller is named /dev/hdb.

Using fdisk

fdisk stands for fixed disk or format disk. The fdisk command is a text-based utility for viewing and managing hard disk partitions on Linux. It is one of the most powerful tools you can use to manage partitions, but it's confusing to new users. With the help of fdisk command you can view, create, resize, delete, change, copy and move partitions on a hard drive using the dialog-driven interface.

fdisk allows you to create a maximum of four primary partitions and the number of logical partitions depend on the size of the hard disk you are using.

Syntax

fdisk [options] device

or

fdisk -l [device]

Some basic fdisk commands to manage a partition in Linux based System:

1. View all disk partitions : `sudo fdisk -l`
The partitions are displayed by their device's names, for example: /dev/sda, /dev/sdb, /dev/sdc.

2. View Partition on a Specific disk
`$ sudo fdisk -l /dev/sda`
This command is used to view all disk partitions on device /dev/sda.

3. View all fdisk Commands: To see all the command which are available under disk command you can use /dev/sda partition with fdisk command.

```
$ sudo fdisk /dev/sda
```

4. Create a new hard disk partition: For this go inside the hard drive partition that is the /dev/sda partition, and use the following command:

```
$ sudo fdisk /dev/sda
```

Now you have to type 'n' to create new partition, 'p' for making a primary partition and 'e' for making an extended or logical partition.

5. Delete a hard disk partition: To delete a partition for the hard disk and free up space occupied by that partition for example /dev/sdb, go to the Command menu using the following:

```
$ sudo fdisk /dev/sda
```

and then type d to go to the delete partition menu.

6. How to view the size of your partition:

```
$ sudo fdisk -s /dev/sda
```

Changing Partition Types

You can use the 'fdisk' utility and use the 't' command to set the partition type. The following example shows how to change the partition type of the first partition to 0x83, default on Linux:

```
sudo fdisk /dev/sdc
Command (m for help) : t
Select partition 1
Partition type (type L to list all types) : 83
Change type of partition 'Linux LVM' to 'Linux'.
```

Making filesystems with mkfs

In linux, you can create filesystem using mkfs, mkfs.ext2, mkfs.ext3, mkfs.ext4, mke4fs or mkfs.xfs commands.

The following commands create an ext4 filesystem on the '/dev/sda3' disk partition:

```
$ sudo mkfs.ext4 /dev/sda3
```

Mounting Filesystems

The most commonly used method for mounting the filesystem is either manually using 'mount' command or by adding entries in /etc/fstab file, so that filesystem gets mounted during boot time.

To mount a filesystem in a given location (mount point), use the 'mount' command in the following form:

```
mount [option] Device_name Directory
```

Once, the filesystem is attached, the mount point becomes the root directory of the mounted file system.

For example, to mount the '/dev/sdb1' file system to the /mnt/media directory, you would use:

Eg1
```
$ sudo mount /dev/sdb1 /mnt/media
```

Example 2
```
$ sudo mount /dev/sda3 /data
```

Here, we have mounted '/dev/sda3' partition to '/data' directory.

Mounting a filesystem using /etc/fstab

Use the mount command in one of the following forms to attach a filesystem specified in the '/etc/fstab' file.

mount [option...] Directory
mount [option] device_name

Unmounting a filesystem

Once a filesystem is mounted, you can use umount command to unmount the filesystem. You can unmount the filesystem by using unmount with the device or the mount point.

Syntax:

```
$ unmount Directory
$ unmount device_name
```

Example

To unmount a filesystem (/dev/sdd1) mounted at '/data' you could enter one of the following:

`unmount /data` or `unmount /dev/sdd1`

Filesystem Types

1) ext2, ext3, ext4

These are the progressive versions of Extended Filesystem (ext), which primarily was developed for MINIX. The second extended version (ext2) was improved version. ext3 added performance improvement. Ext4 was a performance improvement besides additional providing additional features.

2) Jfs (Journaled file System)

JFS is an alternative to ext4 currently and is used where stability is required with the use very few resources. When CPU power is limited, JFS comes handy.

3) ~~Xfs~~ ReiserFS
It was introduced as an alternative to ext3 with improved performance and advanced features.

4) XFS
XFS was a high speed JFS which aimed at parallel I/O processing. NASA still usages this filesystem on their 300+ terabyte storage server.

5) Btrfs. (B-Tree File System)
It focuses on fault tolerance, fun administration, repair system, large storage configuration and is still under development.

Mount Options
To specify the mount options, use the -o flag followed by a comma-separated string of options. The following are some of the available options for the mount command:

- auto: Allows the filesystem to be mounted automatically by using the mount ~~command~~ -a command.
- loop: Mounts the image as a loop device.
- noauto: Disallows the automatic mount of the file system by using the mount -a command.
- noexec: Disallows the execution of binary files on the filesystem.

- nouser : Disallows an ordinary user to mount and unmount the filesystem.
- remount : Remounts the filesystem in case its already mounted.
- ro : Mounts the file system for reading only.
- rw : Mounts the file system for both reading and writing.
- user : Allows an ordinary user to mount and unmount the file system.

Mounting / Unmounting a File, flash drive, cd drive

Mounting a flash drive : `mount /dev/sdb1 /mnt/test`

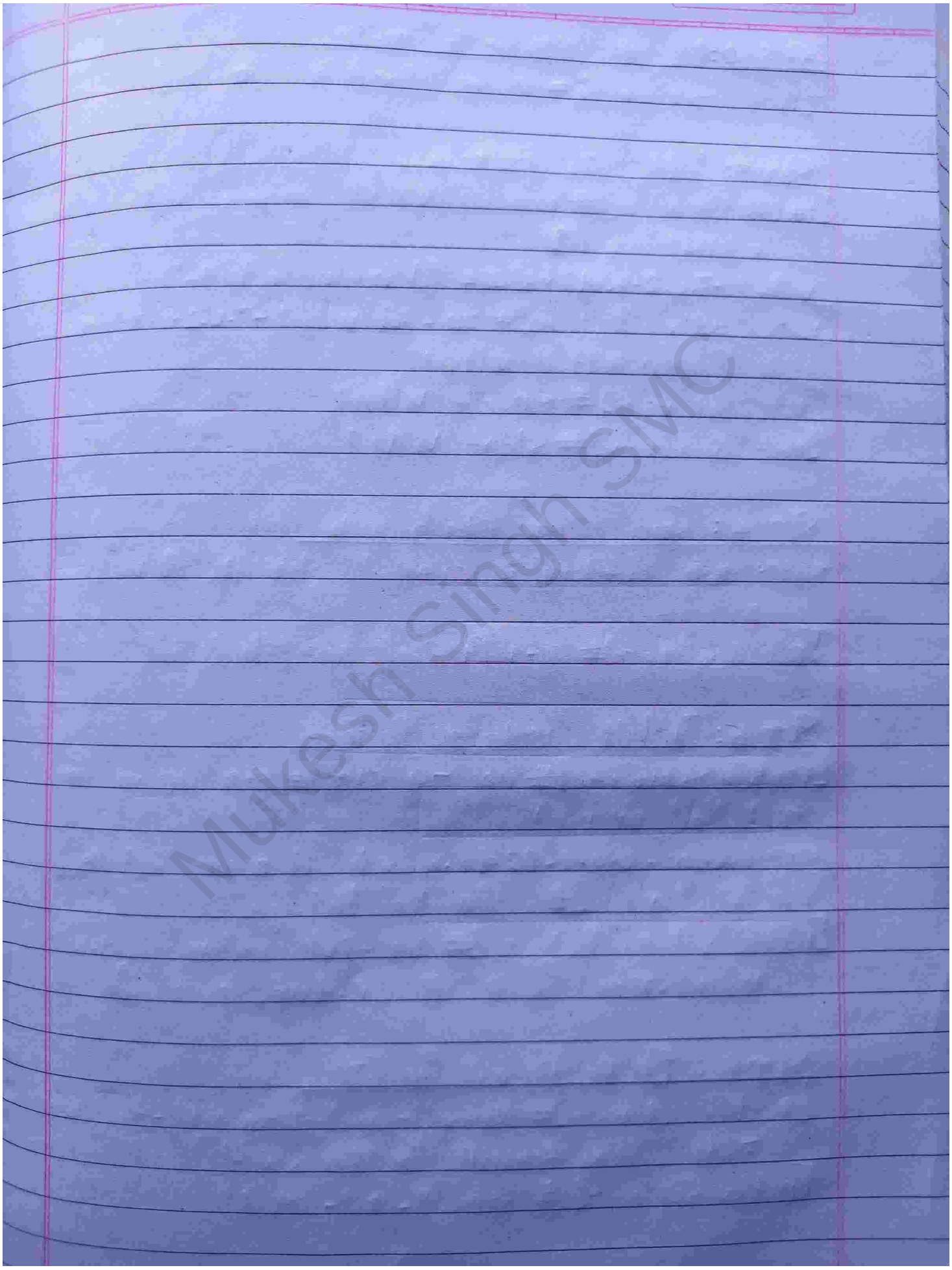
Mounting cd drive : `mount /dev/cdrom /mnt/CD`

Unmounting a flash drive : `umount /dev/sda1`

Unmounting a CD-drive : `umount /dev/cdrom`

1.11 Package management : Install, remove, upgrade package

Package management is a method of installing, updating, removing and keeping track of software updates from specific repositories (repos) in the Linux System.



Unit-2

Installation and Boot Process

2.1 Installation : Client and Server linux installation, Network based Installation

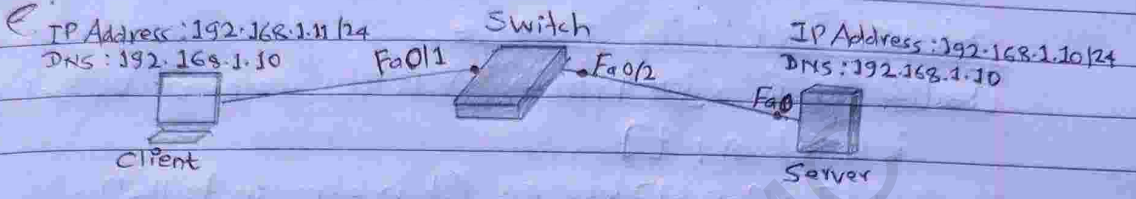


Fig: Client Server Architecture for Network Based OS Installation

Serverside Configuration

Step 1: Configure IP Address on Server

[root@Sukuna ~]# nmcli connection new

NAME	UUID	TYPE	DEVICE
eno16777736	---	802-3-ethernet	---

[root@Sukuna ~]# nmcli connection delete eno16777736 (Remove default connection)

[root@Sukuna ~]# nmcli connection add con-name sukuna type ethernet ifname eno16877736 (Add new Connection Name)

[root@Sukuna ~]# nmcli connection show

NAME	UUID	TYPE	DEVICE
Sukuna	----	802-3-ethernet	eno16777736

[root@Sukuna ~]# nmcli connection modify Sukuna ipv4.addresses '192.168.1.10/24' ipv.dns '192.168.1.10' ipv4.method manual (static IPv4 address)

root@Sukuna ~]# systemctl restart network.service (Restart Network service after assigning an IP address)

[root@Sukuna ~]# ifconfig (for IP address confirmation)

Step 2: Mount cd drive to media
[root@Sukuna ~]# mount /dev/cdrom /media/

Step 3: Install ftp package
[root@Sukuna ~]# rpm -q vsftpd (query for vsftpd package installed or not)
vsftpd-3.0.2-9.el7.x86_64 (it means installed)
[root@Sukuna Package]# rpm -ivh vsftpd-3.0.2-9.el7.x86_64.rpm (for install vsftpd)
[root@Sukuna ~]# enable vsftpd
[root@Sukuna ~]# start vsftpd

Step 4: Copy cdrom files to pub
Note: pub directory is created after install vsftpd package
[root@Sukuna ~]# cp -rf /media/* /var/ftp/pub/

Step 5: Install Createrepo Package:
[root@Sukuna ~]# rpm -q createrepo (query for installed or not)
createrepo-0.9.9-23.el7.noarch (it means installed)
[root@Sukuna ~]# cd/media/Packages
[root@Sukuna Packages]# rpm -ivh createrepo-0.9.9.23.el7.noarch.rpm (for install createrepo package)

Step 6: Enable Firewall for FTP
[root@Sukuna ~]# firewall-cmd --permanent --add service=ftp (allow firewall for ftp service)
[root@Sukuna ~]# firewall-cmd --reload

Step 7: Configure Yum Client

[root@Sukuna ~]# cd /etc/yum.repos.d/
[root@Sukuna yum.repos.d]# vi local.repo
Press i for insert and type following:
[local]
name = local name repository
baseurl = file:///var/pftp/pub/packages
enabled = 1
gpgcheck = 0

Client Side Configuration

Step 1: Boot operating system using bootable ISO files.

Step 2:

2.1 Double click on Network and Hostname
2.2 Turn on Ethernet (by default Ethernet is off)
2.3. Click on Configure
⇒ Choose IPv4 setting
→ Method: Manual
⇒ Click on add
⇒ Address : 192.168.1.11
Netmask: /24
Gateway : 192.168.1.10
DNS: 192.168.1.10
→ Click on Save
→ Click on Done.

Step 4: Configure Installation Source

→ Click on Installation Source
→ On the Network

→ ftp://192.168.1.10/pub
→ Click on Done

Step 5: Software Selection
→ Choose Minimal Installation / Server with GUI.

Step 6: Choose Installation Destination

Step 7: Begin Installation

2.2 Boot Sequence, Kernel Initialization, INIT Process, Boot Loaders, LILO, Selecting What to Boot, Boot loader components, GRUB and grub.conf, Starting the boot process: GRUB, Specifying kernel Parameters, Boot Messages, Kernel initialization; init initialization, Kernel Modules

Boot Sequence

In reality, there are two sequences of events that are required to boot a Linux computer and make it usable: boot and startup. The boot sequence starts when the computer is turned on, and completed when the kernel is initialized and system is launched.

Kernel Initialization

The kernel is the core of any linux system. The first thing that the kernel does is to initialize memory and set up memory allocation table. It initializes all I/O devices including the hard drives and mounts the root file system (/) in the read only mode.

INIT Process

It is always the first program to be loaded executed and is assigned the process ID or PID of 1. Init has specific levels or targets, each of which consists of specific set of services (daemons). These provide various non-operating system services and structures and form the user environment.

Boot Loaders

The boot loader often presents the user with a menu of possible boot options and has a default option, which is selected after some time passes. Once the selection is made, the boot loader loads the kernel into memory, supplies it with some parameter and gives it control.

LILO (Linux Loader)

LILO is a boot loader (a small program that manages a dual boot) for use with the Linux operating system.

It can boot operating systems from floppy disks, hard disks and it does not depend on a specific file system.

Kernel Modules

Kernel modules are pieces of code that can be loaded and unloaded into the kernel upon demand. They extend the functionality of the kernel without the need to reboot the system.

GRUB

GRUB stands for Grand Unified Boot Loader. One of GRUB's most vital capabilities is filesystem navigation that enables straightforward kernel image and configuration choice.

Linux Boot Process:

The following are the 6 high-level stages of a typical Linux boot process:

BIOS	Basic Input/Output System executes MBR
MBR	Master Boot Record executes GRUB
GRUB	Grand Unified Bootloader executes Kernel
Kernel	Kernel executes /sbin/init
Init	Init executes runlevel programs
Runlevel	Runlevel programs are executed from /etc/rc.d/rc*.d/

1. BIOS

- BIOS stands for Basic Input/Output System.
- Performs some system integrity checks.
- Searches, loads and executes the boot loader program.
- It looks for boot loader in floppy, cd-rom or hard drive.
- Once the bootloader program is detected and loaded into the memory, BIOS gives the control to it.
- So, in simple terms BIOS loads and executes the MBR boot loader.

2. MBR

- MBR stands for Master Boot Record.
- It is located in the 1st sector of the bootable disk.

Typically /dev/hda, or /dev/sda.
- MBR is less than 512 bytes in size. This has three components 1) primary boot loader info in 1st 446 bytes 2) partition table info in next 64 bytes 3) mbr validation check in last 2 bytes.
- It contains information about GRUB (or LILO in old systems)
- So, in simple terms, MBR loads and executes the GRUB boot loader.

3. GRUB

- GRUB stands for Grand Unified Boot-Loader.
- If you have multiple kernel images on your system, you can choose which one to be executed.
- GRUB displays a splash screen, waits for few seconds, if you don't enter anything, it loads the default kernel image as specified in the grub configuration file.
- GRUB has the knowledge of the filesystem (the older Linux loader LILO ~~dont~~ didn't understand filesystem).
- GRUB configuration file is /boot/grub/grub.conf (/etc/grub.conf is a link to this).

Here is an example of a simple grub.conf file:

```
#boot=/dev/sda
default=0
timeout=5
splash image=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title CentOS (2.6.18-194.el5PAE)
```

root (hd0,0)
kernel /boot/vmlinuz-2.6.18-194.el5PAE ro root=LABEL=/
initrd /boot/initrd-2.6.18-194.el5PAE.img.

- As you notic from the above info, it contains kernel and initrd image.
- So, in simple terms GRUB just loads and executes kernel and initrd images.

4. Kernel
- Mount the root file system as specified in the "grub.conf" file.
- Then it executes the /sbin/init program, which is always the first program to be executed.
- You can confirm this with its process id (PID), which should always be 1.
- The kernel then establishes a temporary root file system using Initial RAM Disk (initrd) until the real file system is mounted.

5. Init
- Looks at the /etc/initab/ file to decide the Linux run level.
- Following are the available run levels.
  - 0 - halt
  - 1 - Single user mode
  - 2 - Multiuser, without NFS
  - 3 - Full multiuser mode
  - 4 - unused

- 5 - X11
- 6 - reboot

- Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.
- Execute 'grep initdefault /etc/inittab' on your system to identify the default run level.
- If you want to get into trouble, you can set the default run level to 0 or 6.
- Typically you would set the default run level to either 3 or 5.

6. Runlevel Programs

- Depending on which Linux distribution you have installed, you may be able to see different service getting started. For example, you might catch "Starting sendmail ... OK."
- Depending on your initlevel setting, the system will execute the programs from one of the following directories.
  - Runlevel 0 - /etc/rc0.d/
  - Run level 1 - /etc/rc1.d/
  - Run level 2 - /etc/rc2.d/
  - Run level 3 - /etc/rc3.d/
  - Run level 4 - /etc/rc4.d/
  - Run level 5 - /etc/rc5.d/
  - Run level 6 - /etc/rc6.d/

Note that the exact locations of these directories varies from distribution to distribution.

Unit-3

User Administration

User management or user administration describes the ability for administrators to manage user access to various IT resources like systems, devices, applications, storage systems, networks, SaaS services and more.

User administration is a core part to any directory service and is a basic security essential for any organization.

User administration enables admins to control user access and on-board and off-board users to and from IT resources.

3.1 Adding new user account, user private group, modifying/deleting user account, group administration, password aging policies, switching account, network users, authentication, configuration, Default file permission, access control lists

Adding new user account

User accounts are the means by which users present themselves to the system, prove that they are who they claim to be and are granted or denied access to the information and resources on a system.

To create/add user account:

```
# useradd <username>
# passwd <password>
```

User private group

- User private groups (UPGs) are a system configuration idiom that allows multiple users of a system to collaborate on files without any permission ~~to delete~~ hassle.
- It makes UNIX groups easier to manage.
- A UPG is created whenever a new user is added to the system. A UPG has the same name as the user for which it was created and that user is the only member of the UPG.

- $ groups (displays all the groups)
- $ sudo addgroup mynewgroup (adding new group)
- $ usermod -a -G teacher Santosh (to add the user santosh to the group teacher)
- $ groups examplusername (displays the name of group of username)
- adduser -G admin jsmith (create a new user account named jsmith and assign that according to the ftp group)
- usermod -a -G teacher, admin, localperson atul (adding user to multiple groups)

modifying /deleting user account

A) Creating a new user

① To add user:

Syntax: useradd [options] username

Example: useradd -m john

② To create a new account for the new user:

$ sudo useradd newuser

③ Creating a "system" user:

$ sudo useradd -r newuser

B. Modifying user accounts:

Syntax: usermod [options] username

① Locking user password:

```
$ sudo usermod -L mukesh
```

② Unlocking user password:

```
$ sudo usermod -U mukesh
```

③ Change user's UID:

```
usermod -u 1999 Webmaster
```

④ Change user's primary group:

```
usermod -g Webusers Webmaster
```

⑤ Change user Account expiry data:

```
usermod -e 2021-11-01 mukesh
```

('mukesh' user is Dec 1 2021, let's change it to Nov 1 2014 using 'usermod -e' option).

⑥ Change a user login name:

```
usermod -l sanjeev mukesh
```

(Changes username from mukesh to sanjeev)

(VI) Change a username home directory and move all the user files:

usermod -d /home/sanjeev -m mukesh

(Changes the directory of mukesh user, which actually is /home/mukesh, to /home/sanjeev and moves all the files with -m option).

C. Removing/Deleting a user account:

Syntax: userdel [options] user_name

Example.

(i) userdel mukesh (deletes mukesh)

(ii) userdel -r mukesh (deletes user mukesh with root directory)

D. To change your own password:

passwd <username>

e.g. passwd mukesh

E. Switching Accounts:

su user2

Group Administration

The group is nothing but a collection of users using which one can reduce the administration work/task in the OS environment.

Types of Group	Group I.D.
Root	0
System User	1-200
Application User	201-999
Normal User	>=1000

cat/etc/group → Contains grouping information.

FORMAT

<Group-name>:x:GID:<Members of the group>

Note

Group of same username is created if newly created user is not assigned to any group.

Adding the group to the system:

Syntax: groupadd group_name

groupadd myproject

Assign user to the group:

Syntax: usermod -g group_name user_name

Password aging policies

Password aging is a mechanism that allows the system to enforce a certain lifetime for passwords. It allows administrator to take various actions such as forcing user to change password, displaying warning messages to change password, disabling user account, etc.

There are seven settings in password aging policy. These are:

1. Last password change date.
2. Password expires date.
3. Password inactive date.
4. Account expires date
5. Minimum number of days between password change.
6. Maximum number of days between password change.
7. No. of days of warning before password expires.

Example

```
[root@localhost~] chage --maxdays 30 --warndays 5 --
                  inactive 3 age
[root@localhost~] chage -l age

Last password change                                : feb 17, 2021
Password expires                                    : Mar 18, 2021
Password inactive                                   : Mar 21, 2021
Account expires                                     : never
Minimum no. of days between password change : 0
Maximum no. of days between password change : 30
No. of days of warning before password expires : 5
```

Switching Accounts

The Su enables to run a command as another user or switch accounts.

To switch account to user1:

Su - user1.

Network users

A network user is a person who utilizes a computer or network service.

Unlike local user accounts, network user accounts are not bound with any particular system. Based on the configuration, a network user can log in to a specific machine or any machine of the network. Local user accounts and network user ~~accounts~~ accounts are used to access a fully featured operating system.

Authentication Configuration

Red Hat Enterprise Linux supports ~~supports~~ several different authentication methods. They can be configured using the authconfig tool or, in some cases, also using Identity Management tools.

1. Configuring local authentication using authconfig.

The local authentication options area defines settings for local system accounts, not the users stored on the back end.

i) Enabling local access control in the UI:

Enable local access control sets the system to check the /etc/security/access.conf file for local user authorization rules.

ii) Configuring local Access Control in the Command line

There are two options for authenticating to enable local Authorization ~~rules~~ controls.

- enablelocalauthorize
- enablepamaccess

Default file permission

By default, when we create a file as a regular user, it is given the permission of rw-rw-r. We can use the Umask (stands for User mask) command to determine the default permission for newly created files.

The Umask is the value that is subtracted from the 666 (rw-rw-rw) permission when creating new files or from 777 (rwx rwx rwx) when creating new directories.

E.g. If the default ~~Umask~~ Umask is 002, new files will be created with the 664 (rw-rw-r) permission and new directories with the 775 (rwx rwx r-x) permission.

Permission Type	Symbolic	Numeric
READ	r	4
WRITE	w	2
EXECUTE	x	1
NO PERMISSION	-	0
FULL PERMISSION	rwx	7

Access Control lists (ACL)

Access Control list provides an additional more flexible permission mechanism for file systems. An ACL specifies which users or system processes are granted access to objects, as well as what operations are allowed on given objects.

Setfacl and getfacl are used for setting up ACL and showing ACL respectively.

E.g. getfacl test/declaration.h

List of Commands for setting up ACL:

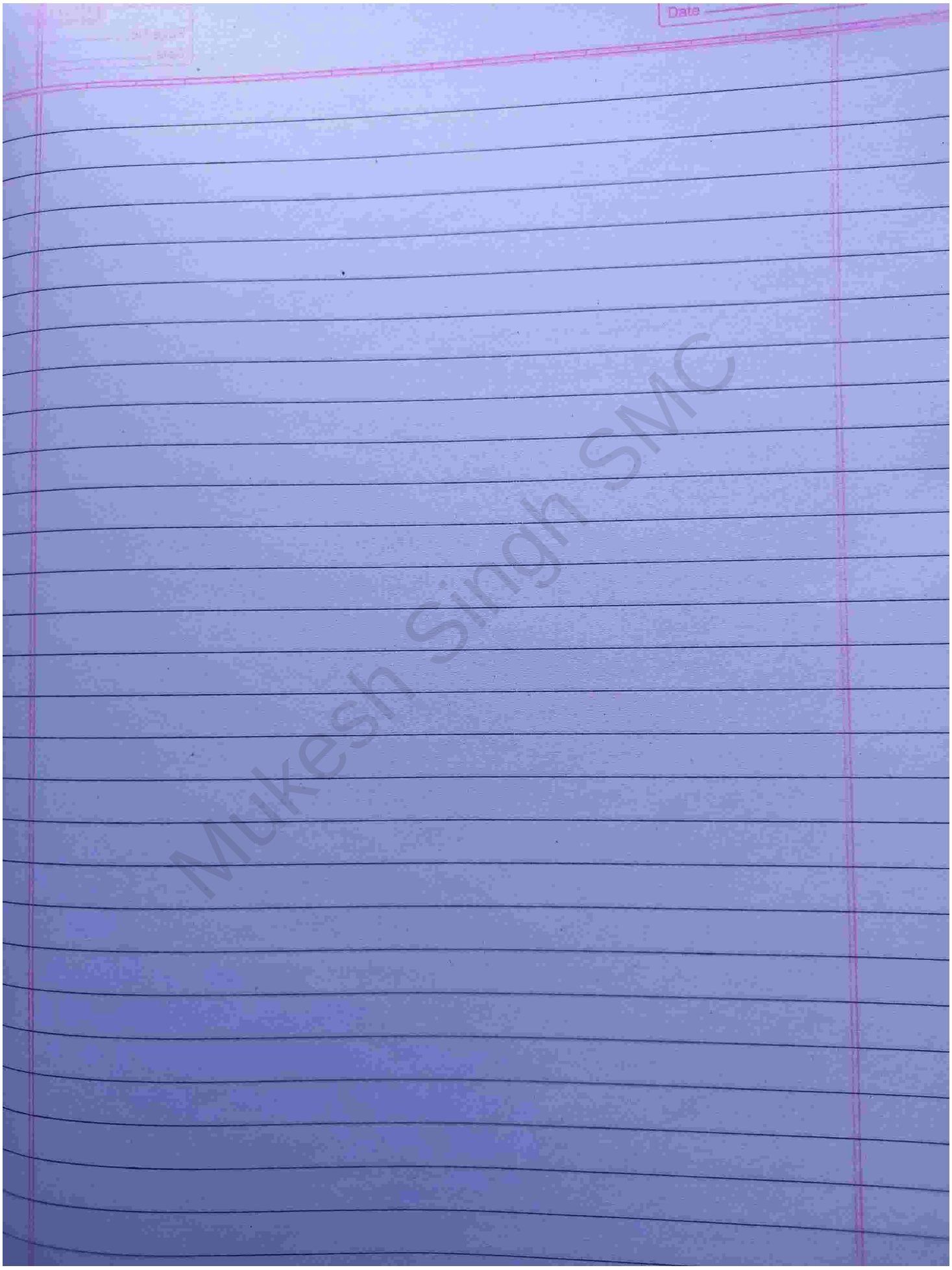
1. To add permission to user:
   setfacl -m "u:user:permissions" /path/to/file

2. To add permission for a group:
   setfacl -m "g:group:permissions" /path/to/file

3. To allow all files or directories to inherit ACL entries from the directory.
   setfacl -dm "entry" /path/to/dir

4. To remove a specific entry
   setfacl -x "entry" /path/to/file

5. To remove all entries
   setfacl -b path/to/file


Date
Mukesh Singh SMC

Unit-4

Set and View Disk Quotas

4.1 What are Quotas, Hard and Soft limits, Per-User and Per-Group Quotas, Block and Inode Limits, Displaying Quota limits; quota, Options in /etc/fstab, Enabling Quota: quotaon, Changing Quota Limits; setquota, edquota and repquota, Software RAID implementation, Creating logical volumes with ~~LVM~~ LVM.

Quotas

Quota is a built in feature of the linux kernel used to set a limit of how much disk space a user or a group use.

Quota is also used to limit the maximum number of files a user or a group can create on linux.

Disk Quotas:

This feature of linux allows the system administration to allocate a maximum amount of disk space a user or group may use.

Hard limit

Hard limit specifies the absolute limit on the disk usages, which a quota user can't go beyond by "hard limit". Hard limit works only when "grace period" is set.

Soft limit

Soft limit indicates the maximum amount of disk usage a quota user has on a partition. When combined with "grace period" it acts as the

border line, which a quota user is issued warnings about his impending quota violation when passed.

Per-User and Per-group Quotas

A Quota is the feature in the Linux kernel that is used to set the limit of how much disk space a user or a group can use. It is also used to limit the maximum number of files a user or a group can create on Linux.

A group quota is the maximum number of disk space that can be used by all files owned by a particular group.

Block and Inode limit

You can define disk quota in two ways:

i) Block (usage) quota/limit → It limits the amount of disk space that can be used.

ii) Inode(file) quota/limits

It limits the number of files that can be created. Each file in Linux consumes a single inode, so the inode limits effectively limits on the number of files a user may own.

Displaying Quota limits

Quota limits and current usage can be displayed for a specific user or a group using the 'gfs-quota' get command. The entire contents of the quota file can also be displayed using the 'gfs-quota list' command.

Displaying Quota Limits for a User:
[gfs-quota get -u User -f Mountpoint]

Displaying Quota Limits for a group:
[gfs-quota get -g Group -f Mountpoint]

Displaying entire Quota file
[gfs-quota list -f Mountpoint]

Configuring Disk Quotas
To implement Disk Quotas, use the following:
1. Enable quotas per file system by modifying the /etc/fstab file.
2. Remount the file system(s).
3. Create the quota database files and generate the disk usage table.
4. Assign quota policies.

Enabling Quota:
* quotaon → The 'quotaon' command enables disk quotas for one or more file systems specified by the filesystem parameter.
The specified filesystem must be defined with quotas in the /etc/filesystem file, and must be mounted.

Syntax
quotaon [-g][-u][-v] {-a|filesystem ...}
By default both user and group quotas are

enabled.
- The -u flag enables only user quotas.
- The -g flag enables only group quotas.
- The -a flag specifies that all filesystems with disk quotas, as indicated by the /etc/filesystem file, are enabled.

E.g. To enable user quotas for the /user filesystem, enter: quotaon -u /user

Changing Quota Limits :

i) setquota → The 'setquota' command sets and changes the quota limits for a user, a group or a file set. A grace period starts when you reach the soft quota limit.

Syntax

setquota filesystem {[-u {userName / userID}] | [-g {groupName / groupID}] | --- default [-i {filesetName / filesetID}] [-h shard] [-s {soft}] [-H {inodeHard}] [-S {inodesoft}] [-c {clusterID / clusterName}]

Example

[setquota gpf50 -u user1 -h 100M -s 70M]

This example specifies a quota for user on file system gpf50 with a hard limit of 100 Megabytes and a soft limit of 70 megabytes.

ii) edquota

edquota command is used to edit the quotas for a user or a group. This is just an editor form where

we can change the quota values.

Eg.1.) Edit user
edquota -u John
(disk quotas for user John (uid: 503):)

2) Edit group
edquota -g Sukuna
(disk quotas for group Sukuna (gid 504)).

iii) repquota
The 'repquota' command prints a summary of quotas and disk usage for a file system specified by the filesystem parameter.
If the -a flag is specified instead of a file system, the repquota command prints the summary for all file systems enabled with quotas in the /etc/filesystem file.
By default, both user and group quotas are printed.

Eg: To print a summary of user quota in the /u file system, enter:
[repquota -u /u]

RAID

RAID stands for Redundant Array of Inexpensive Disks. It is more commonly known as Redundant Array of Independent Disks. RAID is a data storage virtualization technology that combines multiple physical disk drive components into one or more logical units for the purposes of data redundancy, performance improvement, or both.

There are many RAID level such as RAID 0, RAID 1, RAID 5, RAID 10, etc.

Hardware RAID:

Hardware RAID is a physical storage device which is built from multiple hard disks. While connecting with system, all disk appears as a single SCSI disk in system.

Software RAID

Software RAID is a logical storage device which is built from attached disks in system. It uses all resources from system. It provides slow performance but cost nothing.

Software RAID implementation

→ First you need to have a linux distribution installed on your hard drive. We will name it /dev/sda.
→ Then we are going to grab two hard drives which will be named /dev/sdb and /dev/sdc in this post.
→ We will create special file system on /dev/sdb

and /dev/sdc
→ And finally create the RAID 1 array using the mdadm utility.

Steps
1. format hard drive
sudo fdisk -l

2. Install mdadm
mdadm is used for managing MD (multiple devices), also known as linux Software RAID.
Eg.
sudo mdadm --examine /dev/sdb /dev/sdc

3. Create RAID 1 logical drive
sudo mdadm --create /dev/md0 --level=mirror --raid-devices=2 /dev/sdb/ /dev/sdc]
Now we can check it with
cat /proc/mdstat

4. Create File System on the RAID 1 logical drive.
Let's format it to ext4 filesystem
sudo mkfs.ext4 /dev/md0
Then create a mount point /mnt/raid/ and mount the RAID 1 drive.

sudo mkdir /mnt/raid1
sudo mount /dev/md0 /mnt/raid1
We can use the command to check how much disk space we have.

df -h /mnt/raid/

5. Test
Let's go to /mnt/raid/ and create a text file
cd /mnt/raid/
sudo nano raid1.txt

Write something like
This is raid 1 device.

Save and close the file. Next, remove one of your drive out from your computer and check the status RAID1 device again.
sudo mdadm --examine /dev/sdb1 /dev/sdc1.

Creating Logical Volumes with LVM
LVM stands for Logical Volume Management. It is a system of managing logical volumes or file-systems, that is much more advanced and flexible than the traditional method of partitioning a disk into one or more segments and formatting the partition with a filesystem.

- Create the Logical Volume:
First create the logical volume (LV) from existing free space within the volume group. The command below creates a LV with a size of 50GB. The volume group name is MyVG01 and the logical volume Name is stuff.

```
lvcreate -L +50G --name stuff MyVG01
```

Create the filesystem

Creating the logical volume does not create the filesystem. That task must be performed separately. The command below creates an EXT4 filesystem that fits the newly created logical volume.

```
mkfs -t ext4 /dev/MyVG01/stuff
```

Add a filesystem level

Adding a filesystem label makes it easy to identify the filesystem later in case of a crash or other disk related problems.

```
e2label /dev/MyVG01/stuff stuff
```

Mount the filesystem

At this point, you can create a mount point, add an appropriate entry to the /etc/fstab file and mount the system.

You should also check to verify the volume has been created correctly.

You can use the df, fus, and vgs commands to do this.

Unit-5

Network Configuration

5.1. Introduction to IPv4 and IPv6 addres, Dynamic and Static IP address configuration, Network Interface Configuration, Verifying IP Connectivity, Defining a localhost name, Setting Default gateway, router, Diagnosing Network startup issues, Network troubleshooting commands

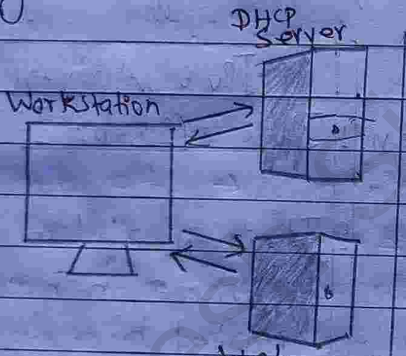
Introduction to IPv4 and IPv6 address:

IPv4 and IPv6 are internet protocols version 4 and internet protocol version 6. IP version 6 is the new version of Internet Protocol, which is way better than IP version 4 in terms of complexity and efficency.

	IPv4 Addres		IPv6 Addres
1.	It is a 32-bit IP addres.	1.	It is a 128-bit IP addres.
2.	It is a numeric addres and its binary bits are separated by a dot (.).	2.	It is an alphanumeric addres ss whose binary bits are separated by a colon (:). It also contains hexadecimal.
3.	Number of header field is 12.	3.	Number of header field is 8.
4.	Length of header field is 20.	4.	Length of header field is 40.
5.	Has checksum fields.	5.	Does not have checksum fields.
6.	It is unicast, broadcast and multicast address.	6.	It is unicast, multicast and anycast type address.
7.	It offers five different classes of IP address, Class A to E.	7.	It ~~offers~~ allows storing unlimited number of IP address.

8. You have to configure a newly installed system before it can communicate with other systems.	8. In IPv6, the configuration is optional, depending upon on functions needed.
9. Example of IPv4 address: 12.244.233.165.	9. Example of IPv6 address: 2001:0db8:0000:0000:0000:ff00:0042:7879

Dynamic and Static IP address Configuration



Dynamic IP Address

Static IP Address

Static IP Address	Dynamic IP Address
1. It is provided by ISP. (Internet Service Provider)	1. It is provided by DHCP (Dynamic Host Configuration Protocol)
2. It does not change any time.	2. It may change any time.
3. It is less secure.	3. It is quite secure than Static IP address.

4.	It is difficult to designate.	4	It is easy to designate.
5.	The device designed by static IP addren can be traced.	5	The device designed by Dynamic IP addren can't be traced.
6.	It is more stable than dynamic IP address.	6	It is less stable than static IP address.
7.	The cost to maintain the static IP address is higher than dynamic IP address.	7	The cost to maintain the Dynamic addren is less than static IP address.

To make network configurations, you have to open (as root or with sudo) the /etc/network/interface file a text editor.

```
$ sudo vim/etc/network/interfaces
```

Set a dynamic IP address:

This is how a network having dynamic IP address is configured:

```
auto etho
iface etho inet dhcp
```

Explanations

- auto etho → enable at startup the etho interfaces.
- iface etho inet dhcp → consider that interface etho comes from interface etho, inet tells you that the next configuration is IPv4 and dhcp that the dynamic IP is assigned by dhcp server.

Page No.
Date

Set a Static IP address

We have to little more work to do for configuring a network interface with static IP address. Edit your /etc/network/interfaces file again, so that it looks like this:

```
auto eth0
iface eth0 inet static
address 192.168.10.5
netmask 955.255.255.0
network 192.168.10.0
broadcast 192.168.10.255
getway 192.168.10.1
```

Explanations

- auto eth0 → enable the startup of eth0 interface.
- iface eth0 inet static → Consider that iface eth0 comes from interface eth0, inet tells you that the network configuration is IPv4 and static that your network interface has static ip addresses.

- address - the network's IP address
- netmask - the network's mask address
- network - the network's address
- broadcast - the broadcast address
- gateway - the gateway address

Network Interface Configuration

Configuring network interfaces involves assigning IP address, setting network parameters and hardware dependent values, specifying network interface and viewing our storage system's network configuration.

- Type sudo vim /etc/network/interface to open the file in the vim editor.
- Editing your configuration file will keep your changes every time the system restarts.

Verifying IP Connectivity by using ping Command

1) Click the start button and click on run.
2) Type command in the text field labeled open and then click the OK button.
   The DOS prompt window appears
3) At the blinking cursor, type 'ipconfig' and then press the 'Enter' key.
4) At the blinking cursor, type ping <ip.address> where <ip address> is the ip address of the computer you are trying to contact on the network i.e. 192.168.2.3
5) Press the 'Enter' key.
   Four replies from that IP address will be returned.

Defining a local host name

In computer network, local host is a ~~host~~ host -name that refers to the current computer used to access it.

- Host information is inside /etc/hosts. Open it with text editor. (here vim text editor).
- $ sudo vim /etc/hosts (for vim editor 'i' to enter in edit mode, :x! for save and exit).
- We can change localhost to any name.

Setting Default gateway

- For example to change the default gateway of the eth0 adapter to 192.168.1.254, you would type:

```
$ sudo route add default gw 192.168.1.254 eth0
```

Setting default routes

A default route in TCP/IP network is a setting that tell the device how to forward the packets when their destination IP is not in the same subnet of the device.

- to list Kernel's Routing Cache information:
  $ route -Cn
- To Reject Routing to a Particular Host or Network
  $ route add -host 192.168.1.51 reject
- If you want to reject an entire network (192.168.1.1 - 192.168.1.255), then add the following entry.

$ route add -net 192.168.1.0 netmask 255.255.255.0 reject

Diagnosing Network Startup issues

Basic Network Problems / issues:

1) Connectivity Problem

The port or interface on which the device is connected or configured can be physically down or faulty due to which the source host will not be able to communicate with the destination host.

2) Cable Problem

The cable which is used to connect two devices can get fault, shortened or can be physically damaged.

3) Configure issue

Due to a wrong configuration, looping the IP, routing problem and other configuration issues, network fault may arise and the service will get affected.

4) Traffic Overload

5) Network ID issues

Due to improper configuration of IP addresses and subnet masks routing IP to the next loop, the source will not be able to reach the destination IP through the network.

Network troubleshooting Commands

i) ifconfig → ifconfig (interface configurator) command is used to initialize an interface, assign IP Address to interface and enable or disable interface on demand.

```
$ ifconfig
```

ii) To enable or disable specific interface:

To Enable etho
```
# ifup etho
```

To disable etho
```
# ifdown etho
```

iii) Ping Command
Ping (Packet Internet Groper) command is the best way to test connectivity between two nodes, whether it is LAN or WAN.

```
$ ping 4.2.2.2
```

iii) Traceroute Command
Traceroute is a network troubleshooting utility which shows number of hops taken to reach destination, also determine packets travelling path.

```
$ traceroute 4.2.2.2
```

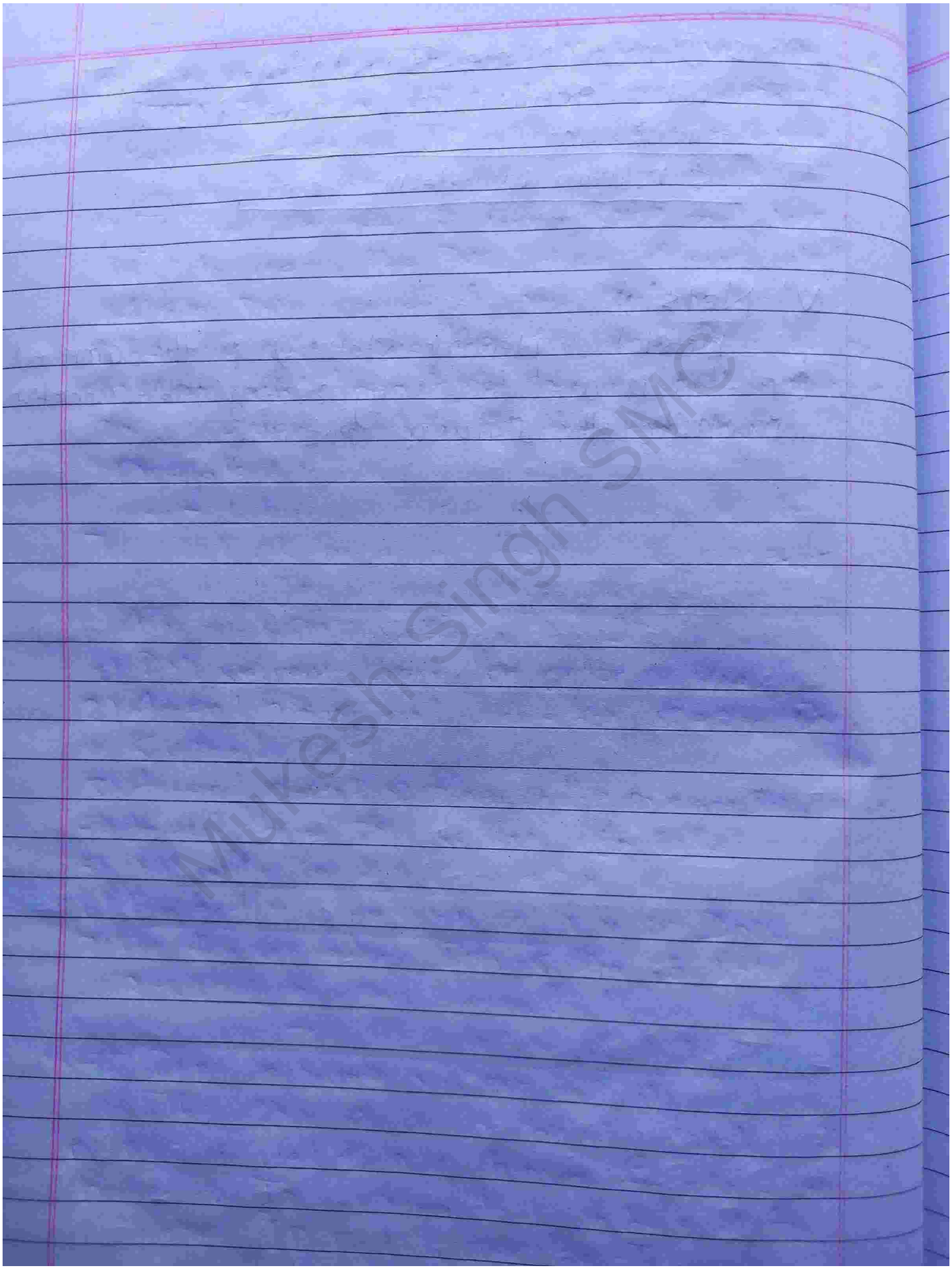
iv) NsLookup Command
nslookup command also used to find out

DNS (Domain Name Server) related query. The following example shows a record (IP Address) of tecmint.com.

```
$ nslookup www.tecmint.com
```

V) Netstat

Netstat (Network Statistics) is the command line that is used to display routing table, connection information, the status of ports, etc.


Mukesh Singh SMC

Unit-6

Organizing Network System

6.1 Configuring DNS: Host name resolution, The Stub resolver, DNS-Specific Resolvers, Trace a DNS query, Forward lookups, Reverse Lookups, Implementing a DNS server, Adding data to the Name server, adding slave DNS.

Configuring DNS: Host name resolution

DNS is a host name resolution service that we can use to determine the IP address of a computer from its host name.

This enables users to work with host names, such as http://www.msn.com rather than an IP address such as 192.168.5.102.

Basic DNS settings:

We can configure basic DNS settings by completing the following steps:

1) Click Start and then click on network. In Network Explorer, click Network and sharing center on the toolbar.

2) In network and sharing center, click 'Manage Network Connections'.

3) In Network Connections, right click the connection we want to work with and then select properties.

4) In the Local Area Connection status dialog box, click properties. This displays the local Area Connection Properties dialog box.

5) Double-click on Internet Protocol version 6/4 (TCP/IPv6) or (TCP/IPv4) as appropriate for the type of IP address we are configuring.

6) If the Computer is using DHCP and we want DHCP to specify the DNS server address, select Obtain DNS server Address automatically. Otherwise, select we the following DNS server address and then type primary and alternate DNS server address in the text boxes provided.

7) Click OK twice and then click close.

Advanced DNS Settings

1) DNS server addresses in order of use.
2) Append primary and Connection Specific DNS Suffixes.
3) Append parent suffixes of the primary DNS suffix.
4) Append these DNS suffixes (In Order).
5) DNS Suffix for this Connection.
6) Register this Connection's address in DNS.
7) Use this Connection's DNS suffix in DNS registration.

Domain Name System (DNS)

The DNS is a distributed database that is mainly used to map human friendly domain names to IP addresses that can be used to deliver IP packets over the Internet.

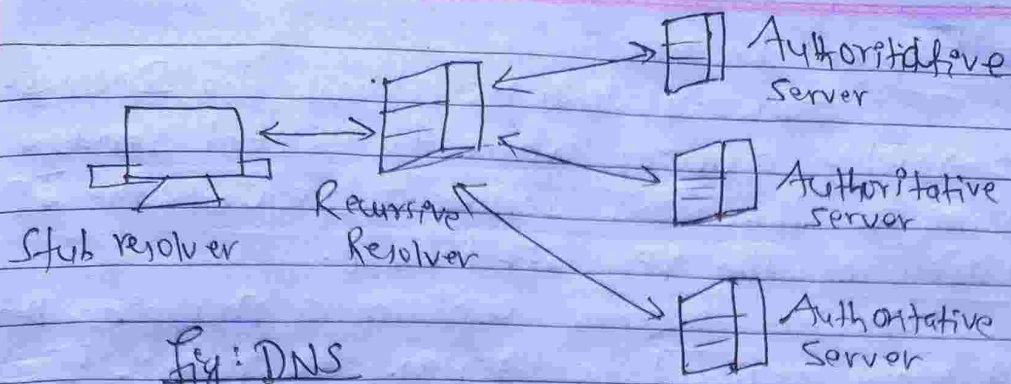


Fig: DNS

The Stub Resolver

The Stub resolver is a component of the DNS that is accessed by application program when using the DNS for eg. resolving domain names to IP addresses.

The stub resolver simply serves as an intermediary between the application requiring DNS resolution and a recursive DNS resolver.

DNS-Specific Resolvers / DNS Resolvers

It is also called resolvers, DNS resolvers are the names given to computers. Commonly located with the Internet Service Provider (ISPs) or institutional networks that are used to respond to a user request to resolve a domain name.

These Computers translate a domain name into an IP address.

Trace a DNS-query

The recursive resolver typically performs a number of successive query to the DNS, to obtain the answer to the query sent by the stub resolver.

Forward Lookups

Forward lookup allow the DNS server to resolve queries where the client sends a name to the DNS server to request the IP address of the requested host.

Forward lookups contains a mapping between host names and IP addresses.

When a Computer requests an IP address by providing a host name, the forward lookup zone is required to find the IP address for the given host name.

Ex- When you type www.google.com in our browser, the forward lookup zone will be required and the IP address 157.166.255.19 will be returned which is actually the IP address of that file.

Reverse Lookup

Reverse DNS Zones perform the opposite task as forward lookup. They return the fully qualified domain name (FQDN) of a given IP address.

For ex: a client cloud send the IP address of 69.163.177.2 to a DNS Server. If the server hosted a reverse zone that included that IP address, it would return the FQDN for that address, such as www.youtube.com.

Implementing a DNS Server

1. Network Information
2. Install bind
3. Configure Cache NameServer
4. Test the Cache NameServer
5. Configure Primary/Master NameServer
6. Build the forward Resolution for Primary/Master Nameserver.
7. Build the reverse Resolution for Primary/Master NameServer.
8. Test the DNS Server.

Adding data to the NameServer

Once you are using the nameserver, follow these steps:

1) Login to your Name.com account.
2) Click on the My Domains button, located on the top right hand corner.
3) Click on the domain name you wish to create an NS records for.
4) Click manage DNS records.
5) Here you will add the desired NS ~~rec~~ records, provided by your host. You would create at least two records since you need a minimum of two nameservers for it to resolve.

Adding Slave DNS

To setup slave DNS server, do the following:

1) Install BIND

```
apt-get install bind
```

2) Allow creating new zones with 'rndc'. In the '/etc/bind/named.conf.options' file, in the 'options {}' directive, type 'allow-new-zone yes;'

3) Specify the IP address from which control instructions should be accepted and set BIND to listen on all accessible network interfaces. In the '/etc/bind/named.conf.local' file, type

```
controls {
  inet * port 953 allow {<plesk-ip>;<another-
plesk-ip> 127.0.0.1;};
  };
```

4. Restart the BIND service by issuing the following command:

```
invoke-rc.d bind9 restart
```

5. Be sure to remember the secret key located in the /etc/bind/rndc.key file.

```
key "rndc-key" {
  algorithm hmac-md5;
  secret "vwoxonI4n4CVRUhKAQAAJA==";
};
```

That is the slave name server is set up.

6.2 Configuring DHCP: Services of DHCP, Configuring an IPv4 DHCP Server

DHCP

Dynamic Host Configuration Protocol (DHCP) is a network management protocol used to automate the process of configuring devices on IP networks, thereby allowing them to use network services such as DNS, NTP and any communication protocol based on UDP or TCP.

DHCP is a networking protocol used to assign IP addresses to network devices.

DHCP Configuration in Linux

A. Server Side Configuration:

Step 1: Install DHCP Package

```
# yum install dhcp
```

Step 2: Update /etc/sysconfig/dhcpd File

```
# vim /etc/sysconfig/dhcpd
```

Step 3: Configure DHCP Server

```
# cp /user/share/doc/dhcp-4.2.5 /dhcpd.conf.example
  /etc/dhcp/dhcpd.conf
```

Step 4: Edit /etc/dhcp/dhcpd.conf file

```
# vim /etc/dhcp/dhcpd.conf
```

Step 5: Start DHCP Service

```
# systemctl start dhcp
```

Client Sede Configuration.

Step 1: Modify IP Address setting to DHCP mode.

```
# nmcli connection modify LAN ipv4.method auto
```

Step 2: Restart Network Senice

```
# systemctl restart network service
```

Step 3: Check IP address and Chack Connection to Server

```
# ifconfig
# ping 192.168.10.1 (Server ip address)
```

Services of DHCP

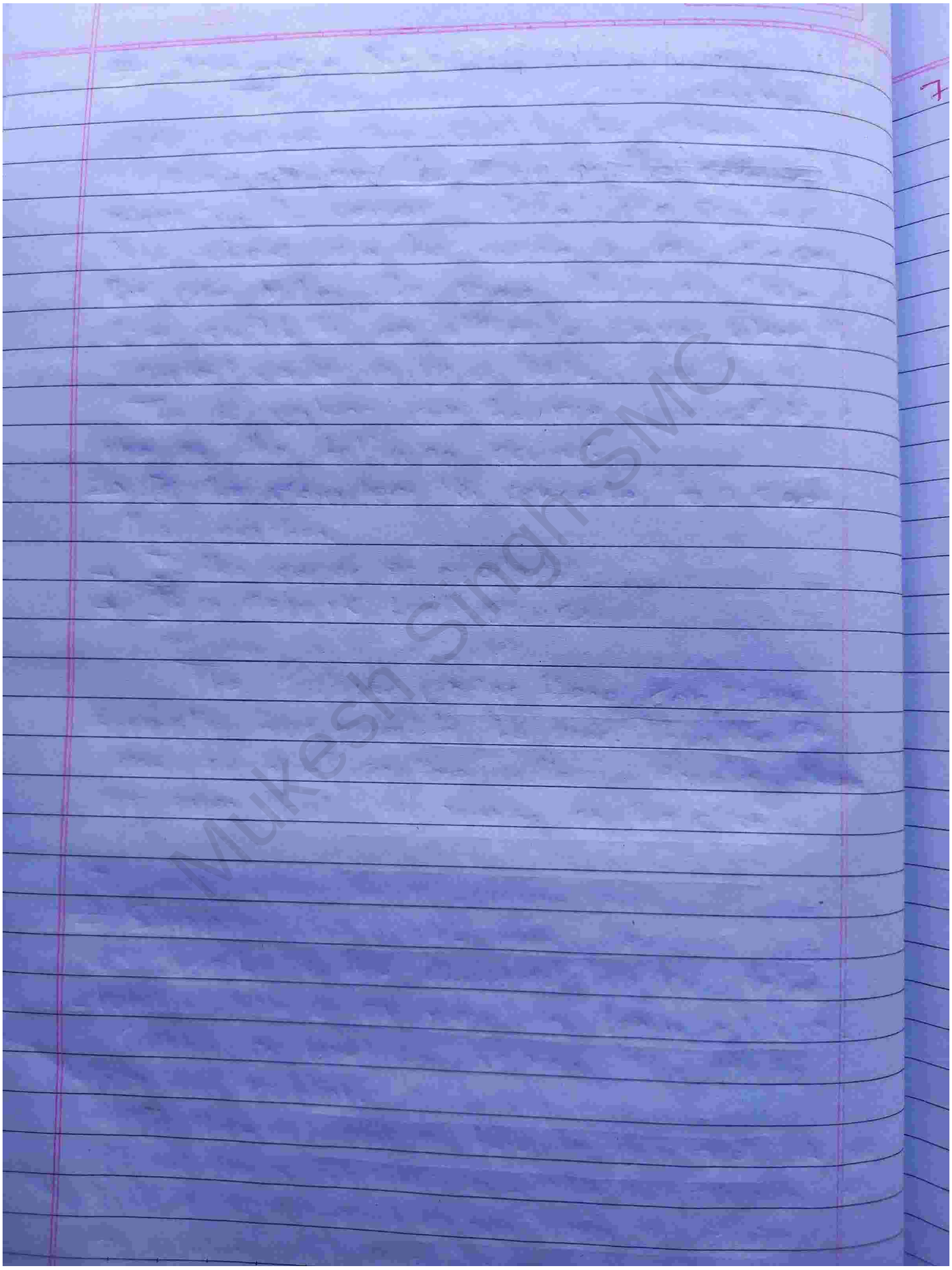
1) Starting, Stopping and restarting the DHCP Service
It affects the running of the daemon only at the current session.

2) Enabling and disabling DHCP Service
It affects running of the daemon for the current and future sessions. If you disable the DHCP server, the currently running daemon terminates and does not start when you reboot the server.

3) Unconfiguring the DHCP Service.
It shuts down the currently running daemon, prevent the daemon from starting on system reboot, and gives you the option

of removing the DHCP data tables.

~~Configuring an IPv4 DHCP~~ Server


Mukesh Singh SMC

Unit-7

Network File Sharing Services

7.1 Introduction to FTP server, NFS Server, Client-Side NFS, Samba

Introduction to FTP Server

- An FTP server is a computer which has a file transfer protocol (FTP) address and is dedicated to receiving an FTP connection.

- An FTP server needs a TCP/IP network for functioning and is dependent on usage of dedicated servers with one or more FTP clients. In order to ensure that connections can be established at all times from the clients, an FTP server is usually switched on.

- An FTP server is important component in FTP architecture and helps in exchanging of files over Internet.
- An FTP server is also known as FTP site.

NFS Server

Network File Sharing (NFS) is a protocol that allows us to share directories and files with other linux clients in a network. The directory to be shared is usually created on the NFS server and files added to it.

If a file system resides on a computer's disk and the computer makes the file system available to other components on the network, that computer acts as a server.

Client-side NFS

The computers that are accessing the file system are said to be clients.
Clients access the files on the server by mounting the server's shared file systems.
Nfs client can be aware of multiple servers that are making the same data available and can switch to an alternate server when the current server is unavailable.
The filesystem can become unavailable if one of the following occurs.

→ If the filesystem is connected to a server that crashes.
→ If the server is overloaded.
→ If a network fault occurs.

The failover under these conditions, is normally transparent to the user. Thus, the failover can occur at anytime without disrupting the processes that are running on the client.

Samba

Samba is the standard Windows interoperability suite of programs for Linux and Unix.
Samba is used for sharing linux file to Windows network.
If there is any machine in our network, we should use samba, Windows does not support NFS. But if all our machines use Linux, it's faster, easier to setup and more secure.

Samba is a piece of software which adds the CIFS (Common Internet File System) filesharing protocol to Linux / Unix.

7.2 Implementation of : File transport Protocol (FTP), service, Network file sharing service, Samba Server

Implementation of File transfer protocol (FTP) Server

Server Side Configuration

1. Install package for FTP Server
   # yum install vsftpd
   # yum install ftp

2. Enable and start FTP service
   # systemctl enable vsftpd
   # systemctl start vsftpd

3. Check status for FTP Service
   # systemctl status vsftpd
   (if service Active : active (exited) it means service is started.

4. Enable firewall for NFS Server
   # firewall-cmd --permanent --add-service=ftp
   # firewall-cmd --reload

5. edit file for ftp to allow network address
   # vi /etc/vsftpd/vsftpd.conf
   listen = YES
   Listen_ipv6 = NO

:wq

6. Check ftp connection
#ftp 192.168.10.1 (FTP Server IP address)

* Client side Configuration

1. Goto Browser
ftp://192.168.10.1 (FTP Server IP address)

Implementation of NFS services

Server Side Configuration:
1. Install package for NFS Server
yum install nfs-utils

2. Enable and start nfs service
systemctl enable nfs-server.service
systemctl start nfs-server.service

3. Check status for nfs-server service
#systemctl status nfs-server.service
(if service Active: active(exited) it means service is started.

4. Enable firewall for NFS service
firewall-cmd --permanent --add-service=nfs
firewall-cmd --permanent --add-service=rpc-bind
firewall-cmd --permanent --add-service=mount

5. Make directory for NFS file path
mkdir /nfsserver

6. Change directory owner (for nfsserver directory)
chown nfsnobody : root /nfsserver/

7. Change permission for /nfsserver directory
chmod 770 /nfsserver/

8. edit file for nfs exports to network address
vi /etc/exports
/nfsserver 192.168.10.0/24 (rw, sync)

9. Check Exported Network and directory path for Confirmation
exports -avr

Client Side Configuration

1. Check mounted location to Client
showmount -e 192.168.10.1 (IP address of NFS Server)

2. Mount Network Location to local Server
mount -t nfs -o sync 192.168.10.1 :/nfsserver /media

Implementation of Samba Server
Server Side Configuration
1. Install Package Configuration
yum yum install samba samba-client samba-common

2. Create directory for samba share
mkdir /samba

3. Create use for samba
useradd smbuser1
passwd smbuser1
Assign password
smbpasswd -a smbuser1

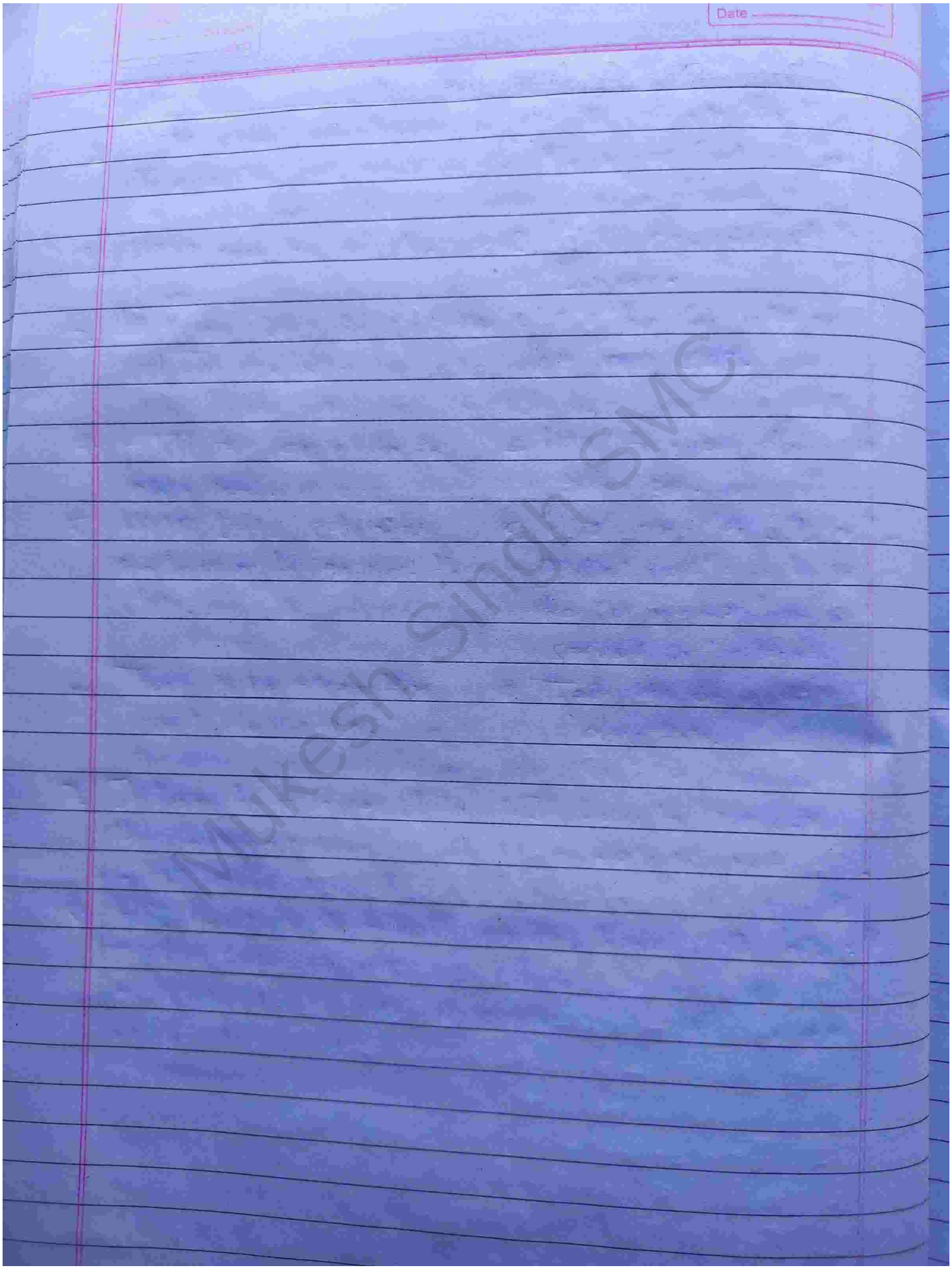
4. Create group for samba user
groupadd smbuser

5. Change owner for share directory
chown :smbuser /samba

6. Change group for smbuser1

7. After installing the samba packages, enable samba services to be allowed through system firewall with these commands.
firewall-cmd --permanent --add-service=samba
firewall-cmd --reload

8. The main samba configuration file is /etc/samba/smb.conf.


Date
Mukesh Singh SMC

Unit-8

Web, Email and database Services

8.1 Implementing web (HTTP and HTTPs) services: Introduction to apache, HTTPD, Server installation and basic configuration, using .htaccess file, Using CGI, Securing Access website

HTTP

HTTP stands for Hypertext Transfer Protocol. HTTP offers set of rules and standards which governs how any information can be transmitted on the World Wide Web. HTTP provides standard rules for web browsers and servers to communicate.

HTTP is an application layer network protocol which is built on top of TCP. HTTP uses hypertext structured text which establishes the logical link between nodes containing text.

It is also known as "stateless protocol" as each command is executed separately, without using reference of previous run command.

HTTPs

HTTPs stands for Hypertext Transfer Protocol Secure. It is highly advanced and secure version of HTTP. It uses port no. 443 for Data Communication. It allows secure transaction by encrypting the entire communication with SSL. It is a combination of SSL/TLS protocol and HTTP.

HTTPs also allows you to create a secure encrypted connection between the server and the browser. It offers bi-directional security of data. This helps you to protect potentially sensitive information from being stolen.

	HTTP		HTTPS
1.	HTTP stands for Hypertext Transfer Protocol.	1.	HTTPS stands for Hypertext Transfer Protocol Secure.
2.	HTTP operates at Application Layer.	2.	HTTPS operates at Transport Layer.
3.	HTTP by default operates on port 80.	3.	HTTPS by default operates on port 443.
4.	HTTP transfers data in plain text.	4.	HTTPS transfers data in ciphertext.
5.	HTTP Lacks security mechanism to encrypt the data.	5.	HTTPS provides SSL or TLS Digital Certificate to secure the communication between server and client.
6.	There is no encryption and decryption.	6.	There is encryption and decryption.
7.	It helps to transfer data through web pages.	7.	It helps to transfer data securely via network.
8.	URL begins with http://	8.	URL begins with https://

Introduction to Apache

Apache http server is free and open source web server that delivers web content through the Internet.

Apache is just one component needed in a web application stack to developer web content.

Apache is considered as open source software, which means the original security code is freely available for viewing and Collaboration.

The main advantage of Apache is ability to handle the large amount of traffic with minimal configuration.

HTTPD

HTTPD is also known as Apache HTTP Server. HTTPD stands for Hypertext Transfer Protocol Daemon.

It is a software program that runs in the background of a webserver and waits for incoming server requests.

The daemon answers the requests automatically and serves the hypertext and multimedia documents over the Internet using HTTP.

Server Installation and Basic Configuration

Before you begin the installation program, you need to make a number of preliminary decisions.

You must perform the following tasks before starting the installation on a Linux server:

i) Creating a user account on a Linux server.

ii) Checking required disk space on a Linux server.

iii) Confirming installation of FTP or SCP.

iv) Configuring the host file.
v) Library requirements
vi) Configuring databases for Linux.
vii) Checking IPv6 configuration on Linux.

Using .htaccess files

The .htaccess file is a powerful website file that contains high-level configuration of our website.

On servers that run Apache, the .htaccess file allows to make changes to our website's configuration without having to edit server configuration files.

Using .htaccess file is a powerful tool for managing our server, but it can be tricky. Make sure we are familiar with making changes to our server before we start editing .htaccess files.

Common uses of .htaccess files

- to ~~make~~ add redirections for certain URLs.
- to load custom error pages, like 404 pages.
- Force our site to use https instead of http.
- to password protect certain directories in our server.
- prevent hot linking.

Using CGI

Common Gateway Interface (CGI) program are external programs that server calls to process data.

We can set up our virtual server to serve dynamic content in two ways:
using Jave based web applications and using non-Java based applications such as CGI, ASP, Server Side HTML (SHTML) tags and NSAPI programs.

Features of CGI

→ It is a very well defined and supported standard.
→ CGI is a technology that interfaces with HTML.
→ CGI is the best method to create a counter because it is currently the quickest.
→ CGI standard is generally the most compatible with today's browsers.

Securing Access Website

Steps:

1) Use secure passwords.
2) Be careful when opening emails.
3) Install Software updates
4) Use a secure website hosting service.
5) An SSL certificate keeps information protected.
6) Secure folder permission.
7) Run regular website security checks.
8) Update website platforms and scripts
9) Install security plugins.
10) Watch out for XSS attacks
11) Beware of SQL injection

8.2 Email: Essential of Email operation, SMTP protocol, MTA, Implementation and Configuration email: Sendmail Configuration (incoming and outgoing), Postfix (incoming and outgoing) Procmail MTA. Configuration, Dovecot setup

Email

Email is a service which allows us to send the message in electronic mode over the Internet. It offers an efficient, inexpensive and real time mean of distributing information among people.

Email is probably the one of the most popular and useful uses of a Linux based server or desktop system. It is surprising to some people to find out how complex the email structure is on a Linux system and this complexity can be a little overwhelming (amazing, vast) to the Linux new-comer.

Essential email operations:

i) Creating email account
ii) Composing and sending email
iii) Reading Email
iv) Replying Email
v) Forwarding Email
vi) Deleting Email

SMTP Protocol

SMTP Stands for Simple Mail Transfer Protocol. This is the protocol used by email systems to transfer mail messages from one server to

another.

The protocol is essentially the communication language that the MTAs use to talk to each other and transfer messages back and forth.

It can send a single message to one or more ~~recei~~ recipients. Sending message can include text, voice, video or graphics.

It can also send the message on network outside the Internet.

The main purpose of SMTP is to set up communication rules between servers.

Working of SMTP

i) Composition of mail
ii) Submission of mail
iii) Delivery of mail
iv) Receipt and processing of mail.
v) Access and Retrieval of mail.

MTA

A Mail Transfer Protocol (MTA) is a software that transfers emails between the computers of sender and a recipient.

MTA is the part of the email system that does much of the work of transferring the email message from one computer to another (either on the same local network or over the Internet to a remote system).

Once configured correctly, most users will not have any direct interaction with their chosen MTA unless they wish to reconfigure it for any reason.

There are many choices of MTA available for Linux including Sendmail, Postfix, Fetchmail, Qmail and Exim.

Implementation and Configuration email:

1. Sendmail Configuration (incoming and outgoing)

Sendmail's core purpose, like other MTAs, is to safely transfer email among hosts, usually using the SMTP protocol.

However, Sendmail is highly configurable, allowing control over almost every aspect of how email is handled, including the protocol used.

In order to use Sendmail, first ensure the Sendmail package is installed on your system by running, as root:

```
~]# yum install sendmail
```

In order to configure Sendmail, ensure that sendmail-cf package is installed on your system by running, as root:

```
~]# yum install sendmail-cf
```

Sendmail configuration file is located at `/etc/mail/sendmail.cf`.

Avoid editing the `sendmail.cf` file directly. To make configuration changes to Sendmail, edit the `/etc/mail/sendmail.mc` file, back up the original `/etc/mail/sendmail.cf` file, and then restart the

sendmail service.

As a part of restart, the sendmail.cf file and all binary representations of the databases are rebuild.

systemctl restart sendmail

2. Postfix (Incoming and Outgoing) Procmail MTA Configuration

- Postfix is a Sendmail-compatible MTA that is designed to be secure, fast, and easy to configure.

To improve the security, Postfix uses a modular design, where small processes with limited privileges are launched by a master daemon.
(Daemon is a long-running background process that answers requests for services)

Basic Postfix Configuration:

By default, Postfix does not accept network connection from any host other than the local host. Perform the following steps as root to enable mail delivery for other hosts on the network:

- Edit the /etc/postfix/main.cf file with a text editor, such as vi.
- Uncomment the mydomain line by removing the '#' hash sign, and replace domain.tld with the domain the mail server is servicing, such as example.com.
- Uncomment the myorigin = $mydomain line.

- Uncomment the myhostname line, and replace host.domain.tld with the hostname for the machine.

- Uncomment the mydestination = $myhostname, localhost.$mydomain line.

- Uncomment the mynetworks line, and replace 168.100.189.0/28 with a valid network setting for hosts that can connect to the server.

- Uncomment the inet_interfaces = all line.

- Comment the inet_interfaces = localhost line.

- Restart the postfix service.

Once these steps are complete, the host accepts outside emails for delivery.

Dovecot Setup

Dovecot is an open source application that allows us to receive emails on linux server in total security both through IMAP and POP3 protocol.

The primary purpose of Dovecot is to act as mail storage server. Mail is delivered to the server using some mail delivery agent (MDA) and stored for later access with an email client (mail user agent, MUA).

Dovecot installation:

To install basic dovecot server with common POP3 and IMAP functions, run the following command:

```
sudo apt install dovecot imapd dovecot-pop3d
```

But it is simple to install dovecot with 'yum'.

```
$ yum install-y dovecot
```

Once the installation is Complete, you can enable the service and start it with systemctl:

```
$ systemctl enable dovecot
$ systemctl start dovecot
```

Configuring Dovecot:

The main configuration file is located at:
/etc/dovecot/dovecot.conf

There are three Dovecot configuration options we will cover: listen, protocols and mail_location.

i) listen

The 'listen' configuration option sets the IP addresses where you want the service to listen. Usually, you use an asterisk (*) here as your value, which is a wild card meaning all IPv4 addresses. For IPv6 addresses, you would use a double colon (::). Here's how to add both:

```
listen = *, ::
```

Page No.
Date

8.3

ii) protocols

The 'protocol' configuration option allows you to specify which protocols you would like to support, such as IMAP and POP3. Usually, LMPT is listed here as well, which stands for the Local Mail Transfer Protocol.

```
protocols = imap, pop3, lmpt
```

iii) mail_location

The mail_location configuration option sets where the mail is picked up from. By default, this setting is empty, which means that Dovecot attempts to locate your mail automatically. The format of the mailbox location specification option is as follows:

```
mailbox-format:<path>[:key=<value>...]
```

To configure dovecot, edit the file: '/etc/dovecot/dovecot.conf' and its included config files in '/etc/dovecot/conf.d/'.

By default, all installed protocols will be enabled via an include directive in '/etc/dovecot/dovecot.conf'

A basic self-signed SSL certificate is automatically set up by package ssl-cert and used by Dovecot in /etc/dovecot/conf.d/10-ssl.conf

Once we have configured Dovecot, restart its daemon in order to test our setup.

```
sudo service dovecot restart
```

8.3 Setting up and Administering Database Server (MySQL)

MySQL

My SQL is the most popular opensource, Relational Database Management System. MySQL is a most popular database server for Linux systems, it also supports a large number of platforms. In MySQL, we can easily created a stored procedure and execute SQL queries. MySQL Community edition is freely downloadable version and use for your applications.

Setting up and Administering Database Server (MySQL):

Installing and Configuring MySQL:

1. Install the MySQL database server package.
   You can use the Yum to install MySQL on Oracle Linux:

```
sudo yum install mysql-community-server
```

2. Start MySQL service -

```
sudo systemctl start mysqld
```

3. Launch ~~mysql~~ the MySQL Command-Line Client

```
mysql -u root -p
```

4. Create a user and a strong password:

```
mysql> create user 'mukesh' identified by '<strong password>';
```

5. Create the database and grant all access to the user, for example, mukesh as follows:

```
mysql > create database mukesh;
mysql > grant all on mukesh.* to 'mukesh';
```

6. Configure your MySQL installation to handle large BLOB entries, such as AMC Agent (installation) bundle and MST binaries. To handle BLOB entries, edit 'my.cnf' file.

To edit 'my.cnf' file :

a. Open the 'my.cnf file in an editor. You can find 'my.cnf' file in one of the following locations:

- /etc/my.cnf
- /etc/mysql/my.cnf
- ~~/etc~~ $MYSQL_HOME/my.cnf
- [datadir]/my.cnf

b. Set the options 'max_allowed_packet' and innodb_log_file_size in the [mysqld] section to the values shown:

```
[mysqld]
max_allowed_packet = 300M
innodb_log_file_size = 768 M.
```

c. Restart the MySQL service to apply changes:

```
sudo systemctl mysqld restart
```

Administering MySQL (MySQL Administration)

Section 1

- Start MySQL Server : `Sudo service mysqld start`
- Stop MySQL Server : `Sudo service mysqld stop`
- Restart MySQL Server : `Sudo service mysqld restart`

Section 2 Users, Roles and Priveleges

i) Create Users

```
CREATE USER [IF NOT EXIST] account_name
IDENTIFIED BY 'password';
```

ii) Grand Privelege : Syntax

```
GRANT privilege [,privilege], ...
ON privelege_level
TO account_name
```

Syntax

E.g

```
GRANT SELECT
ON employees
To bob@localhost
```

Example

iii) Revoke Priveleges

```
REVOKE privelege
    privelege [,privelge]...
ON [object_type] privelge_level
FROM user1 [,user2]...;
```

Syntax

```
REVOKE INSERT, UPDATE
ON classmodels.*
FROM rfc@localhost;
```

Example

iv) Manage Roles - manage roles in the database

```
CREATE ROLE
crm_dev,
crm_read,
crm_write;
```

```
GRANT crm_read
To crm_read1@localhost;
```

Grants role to users:

v) Show Granted Privileges

```
SHOW GRANTS
  FOR musk@localhost;
```

vi) Drop users - show how to delete user account

```
DROP USER account_name
```
-Syntax

```
mysql> drop user dbadmin@localhost
```
-Example

vii) Change Password

```
SET PASSWORD FOR 'dbadmin@localhost' = bigshark;
ALTER USER dbadmin@localhost IDENTIFIED BY 'littlewhale';
```

viii) Show Users

```
SELECT
  User
FROM
  mysql.user;
```

ix) Rename user:

```
RENAME USER old_user1
To new-user;
```

x) Lock user accounts

```
CREATE USER account_name
IDENTIFIED BY 'password'
ACCOUNT LOCK;
```

xi) Unlock user accounts

```
ALTEL USER [IF EXISTS] account_name
ACCOUNT UNLOCK
```

Section 3 Show Commands

i) Show Databases - displays all database in the MySQL Server.

```
SHOW DATABASES;
```

ii) Show Tables - list all tables in a given database

```
SHOW TABLES;
```

iii) Show Columns - lists all Columns in a table

```
SHOW COLUMNS FROM table_name;
```

iv) Show Processlist - show the current process in the MySQL server.

```
SHOW [FULL] PROCESSLIST;
```

Section 4 Backup and Restore

i) Backup - make backup of one or more databases using the mysqldump tool.

```
mysqldumb --user=root --password=sukuna --result file=
c:\backup\classicmodel.sql --.
```

(username: root; password: sukuna; backup path: c:\backup\classicmodel.sql)

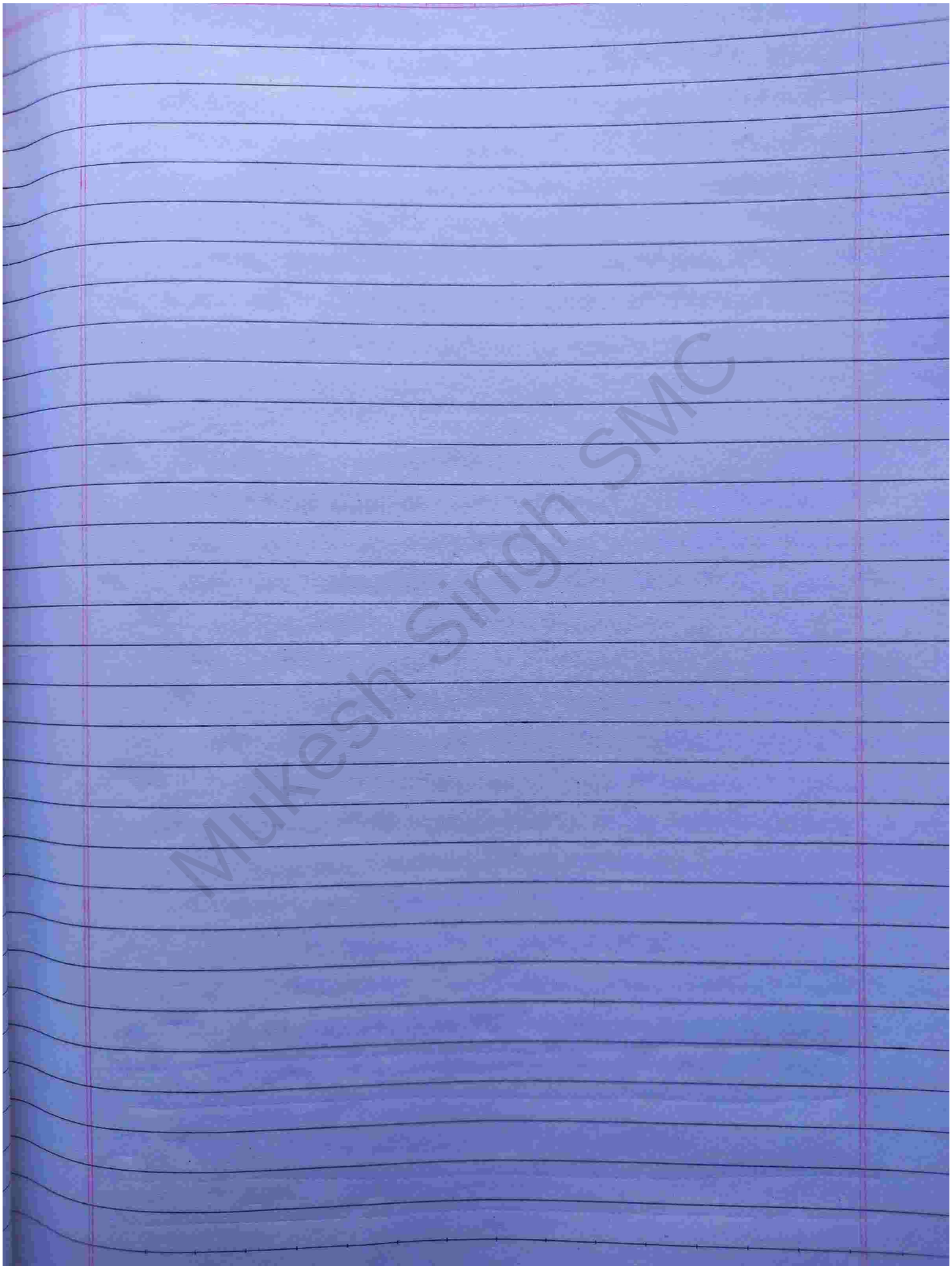
ii) Restore - restore a database from a dumb file.

```
mysql> drop database mydb;
mysql> source c:\backup\mydb.sql
```

Section 5 Database Maintenance

1) Maintain Tables - provide you with commands to maintain tables.

```
ANALYZE TABLE payments;
OPTIMIZE TABLE orders;
CHECK TABLE orders;
REPAIR TABLE employees;
```


Mukesh Singh SMC

Unit - 9

Securing Data and Network

9.1 Introduction to Cryptography, symmetric encryption, asymmetric encryption, PKI, Digital Certificates, generation of Digital Certificates, Open SSH overview, SSH authentication, using SSH keys with/without Passphrase, Using an SSH tunnel

Cryptography

Cryptography is the technique ~~where~~ which is used for doing secure communication between two parties in the public environment where unauthorized users and malicious attackers are present.

It is a method of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it.

Cryptography not only protects data from theft or alteration, but can also be used for user authentication.

In cryptography, there are two process i.e. encryption and decryption performed at sender and receiver end respectively.

Encryption

Encryption is the mechanism to convert the readable plaintext into unreadable (scrambled text (i.e. ciphertext) by using some algorithm and key.

Decryption

Decryption is the reverse process of encryption. It converts ciphertexts back to the plaintext by using some algorithm and key.

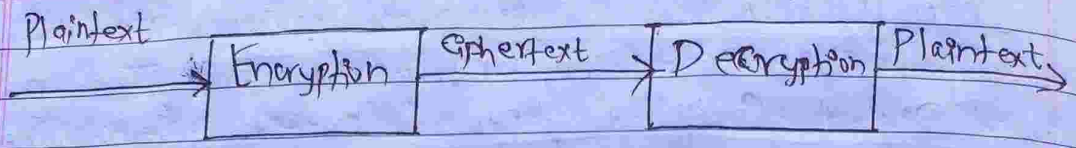


Fig: Encryption-Decryption

Types of Cryptography:

1. Private-key Cryptography (Symmetric encryption)
2. Public-key Cryptography (Asymmetric encryption)

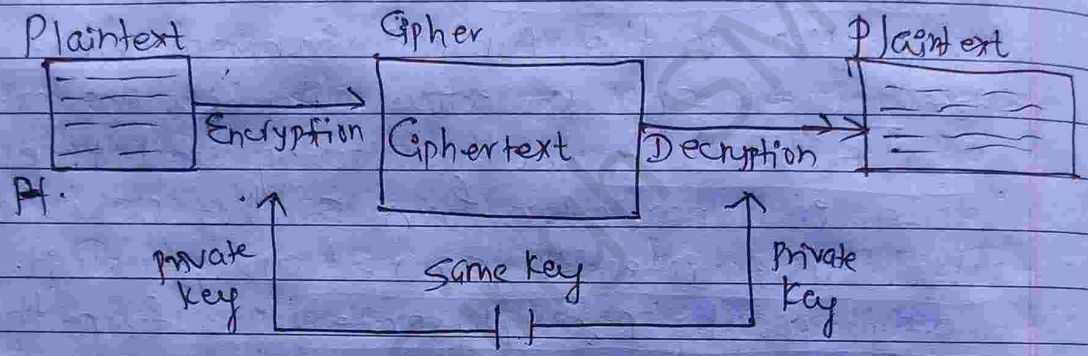


Fig: Symmetric Encryption

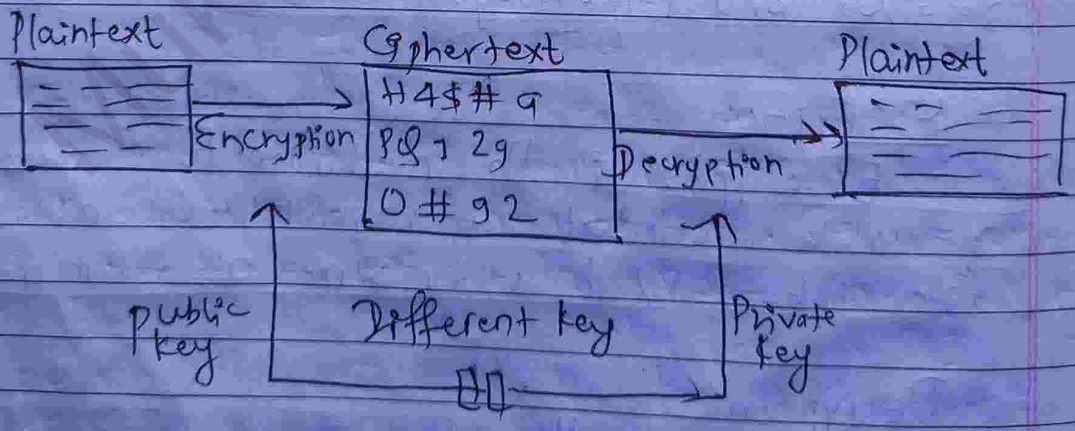


Fig: Asymmetric Encryption

	Symmetric Encryption		Asymmetric Encryption
1.	Same key is used for encryption and decryption. Key must be kept secret.	1	Different keys are used for encryption and decryption. Public key for encryption and private key for decryption.
2.	Encryption & decryption process is more faster than public key cryptography (Asymmetric encryption)	2	Encryption and decryption process is slower than private key cryptography (Symmetric encryption).
3.	It can't be used for other systems than achieving Confidentiality.	3	It can also be used in digital signatures and authentication systems.
4.	The size of Ciphertext is same or smaller than the original plaintext.	4	The size of ciphertext is same or larger than the original plaintext.
5.	It is required when large amount of data is required to transfer.	5.	It is required ~~when~~ to transfer small amount of data.
6.	It only provide Confidentiality.	6	It provides confidentiality, authenticity and non-repudiation.
7	Examples: 3DES, AES, DES, and RSA.	7.	Examples: DSA, RSA, ECC, Diffie-Hellman etc.

Public Key Encryption (PKI)

The comprehensive system required to provide public-key encryption and digital signature services is known as a public-key infrastructure.

The purpose of a public-key infrastructure is to manage key and certificates. By managing keys and certificates through a PKI, an organization establishes and maintains a trustworthy networking environment.

A typical PKI consists of hardware, software, policies, and standards to manage the creation, administration, distribution and revocation of keys and digital certificates.

Digital certificates are the heart of PKI as they affirm the identity of the subject and bind that identity to the public key contained in the certificate.

Components/elements of PKI:

1. Digital Certificates

PKI functions because of digital certificates. A digital certificate is like a driver's license - it is a form of electronic identification for websites and organizations. Secure connection ~~are made~~ between two ~~parties~~ communicating machines are made available through PKI because the identities of the two parties can be verified by way of certificates.

Digital Certificates are based on the ITU standard X.509 which defines a standard certificate format for public key certificates. And ~~certificates~~

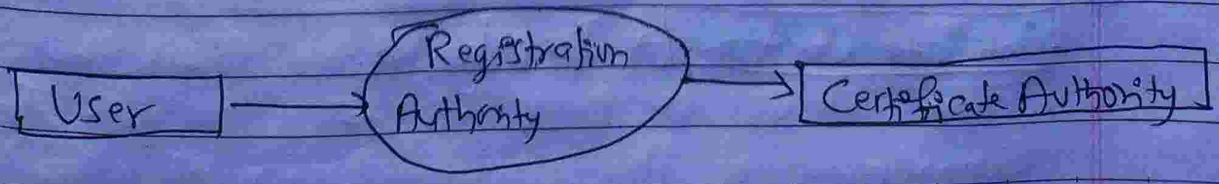
certification validation.
Hence, digital certificates are sometimes also referred to as x.509 certificates.

2. Certificate Authority (CA)
A trusted third party that acts as the root of trust and provides services that authenticate the identity of individuals, computers and other entities. CA prevents falsified entities and manage the life cycle of any given number of digital certificates within the system.

3. Registration Authority (RA)
Registration Authority (RA), which is authorized by the Certificate Authority (CA) to provide digital certificates to users on a case-by-case basis. All of the certificates that are registered, received and revoked by both the CA and RA are stored in an encrypted certificate database.

4. Certificate database.
It stores certificate requests and issues and revokes certificates.

5. Certificate store -
It resides on a local computer and is a place to store certificate and private keys.



Generation of Digital Certificates

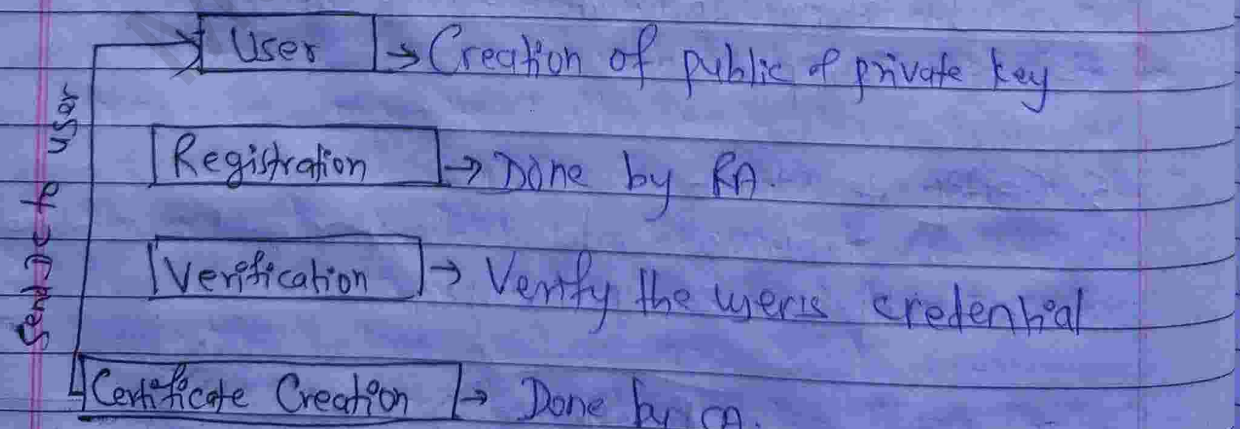
Steps

Step 1: Key generation is done either by user or registration authority. The public key which is generated is sent to the registration authority and private key is kept secret by user.

Step 2: In the next step the registration authority registers the user.

Step 3: Next step is verification which is done by registration authority in which the users credentials are being verified by registration authority. It also checks that the user who send the public key have corresponding private key or not.

Step 4: In this step, the details are sent to the certificate authority by registration authority who creates the digital certificate and give it to users and also keeps a copy to itself.



~~Open~~ SSH ~~Overview~~

SSH, also known as Secure Shell or Secure Socket Shell, is a network protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network.

SSH provides strong password authentication and public key authentication, as well as encrypted data communication between two computers connecting over an open network, such as the Internet.

In addition to providing strong encryption, SSH is ~~used~~ widely used by network administrators for managing systems and applications remotely, enabling them to login to another computer over a network, execute commands and move files from one computer to another.

Open SSH

Open SSH is the premier connectivity tool for remote login with the SSH protocol. It encrypts all traffic to eliminate eavesdropping, connection hijacking, and other attacks. In addition, Open SSH provides a large suite of secure tunneling capabilities, several authentication methods and sophisticated configuration options.

The OpenSSH suite consists of following tools:

- Remote operations are done using ssh, scp and sftp.
- Key management with 'ssh-add', 'ssh-keysign', 'ssh-keyscan' and 'ssh-keygen'.

- The server side consists of sshd, sftp-server and ssh-agent.

Open SSH is developed by a few developers of the OpenBSD project and made available under a BSD-style license.

OpenSSH is incorporated into many commercial products, but very few of those companies assist OpenSSH with funding.

Contributions towards OpenSSH can be sent to the OpenBSD foundation.

SSH Authentication

An SSH server can authenticate clients using a variety of different methods. The most basic of these is password authentication, which is easy to use but not the most secure.

Although passwords are sent to the server in a secure manner, they are generally not complex or long enough to be resistant to be repeated, persistent attackers.

SSH key pairs are two cryptographically secure keys that can be used to authenticate a client to an SSH server. Each key pair consists of a public key and a private key.

The private key is retained by the client and should be kept absolutely secret. Any compromise of the private key will allow the

attacker to log into servers that are configured with the associated public key without additional authentication.

The associated public key can be shared freely without any negative consequences.

When a client attempts to authenticate using SSH keys, the server can test the client on whether they are in possession of the private key. If the client can prove that it owns the private key, a shell session is spawned or the required command is executed.

Using SSH keys with/without passphrase

When creating SSH keys, we can create them with or without a passphrase.

If we do create a key with passphrase, we will be asked for passphrase everytime we try to communicate with our Git repository in Beanstalk. Using a passphrase increases the security when we are using SSH keys.

But using a key without a passphrase can be risky. If someone obtains a key (from a back tape, or a one-time vulnerability) that does not include a passphrase, the remote account can be compromised.

Setup SSH passwordless Login

Step 1: Check for existing key pair

Run the following 'ls' command to see if existing SSH keys are present:

```
$ ls -al ~/.ssh/id_*.pub
```

Step 2: Generate a new SSH key pair:
The following command will generate a new 4096 bits SSH key pair with your email address as a comment:

```
$ ssh-keygen -t rsa -b 4096 -c "your_email@domain.com"
```

Press Enter to accept the default file location and filename:

Output

```
Enter file in which to save the key (/home/your_user_name/.ssh/id_rsa):
```

Next the 'ssh-keygen' ~~tool~~ tool will ask you to type a secure passphrase
If you don't want to use a passphrase just press 'Enter'

Output

```
Enter passphrase (empty for no passphrase):
```

Step 3: Copy the public key
Now that you have generated SSH key pair, in order to be able to login to your server without a password you need to copy the public key to the server you want to manage.
On your local machine terminal type:

```
$ ssh-copy-id remote_username@server_ip_address
```

You will be prompted to enter the remote-username password:

Output

```
remote_username@server_ip_address's password:
```

Step 4: Login to your server using SSH keys

To test it just try to login to your server via SSH:

```
$ssh remote_username@server_ip_address
```

If everything went well, you will be logged in immediately.

Using an SSH tunnel

SSH tunnelling is a method of transporting arbitrary networking data over an encrypted SSH connection.

It can be used to add encryption to legacy applications.

It can also be used to implement VPNs (Virtual Private Networks) and access intranet services across firewalls.

SSH tunneling enables adding network security to legacy application that do not natively support encryption.

For example, entire country wide ATM networks run using tunneling for security.

9.2 Configuring a firewall, Applying ACL

Firewall

A firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules.

A firewall is a network security system designed to prevent unauthorized access to or from a private network.

Firewalls can be implemented on both hardware and software or a combination of both.

Configuring a firewall

Steps to Configure a firewall in Linux manually:

Step 1: Up your Linux Security

Prior to configuring a firewall for your linux system, it is equally important to make sure your Linux system is is upto date with the latest security updates installed, as well as, operating system version is also up to date.

Step 2: Configuring iptables

iptables is a command-line firewall utility program that allows filtering traffic. The iptables tool decides which packages can come in and go out based on the rules it is configured to

follow. It uses policy chains to allow or block the traffic. There are three type of policy chains:

1. Input - used to control the behaviour for incoming connections i.e packages going to local sockets.

2. Forward - used for packets that aren't being delivered locally i.e. packets routed via the server.

3. Output - used for outgoing connections i.e. packets ~~granted~~ generated locally.

Iptables installation commands for:

- enterprise and cent os:

  # sudo yum install iptables-services

- For ubuntu os:

  # sudo apt-get install iptables

Step 3 : Decide what to block

If you want to block/drop connections for a particular IP address, run the following command:

#iptables -A INPUT -s 10.10.10.10 -j DROP

Where 10.10.10.10 is the IP address you want to drop.

Step 4: Deciding which ports to keep on

The decision to leave ports open depends on your server and what you are using your server for. Here are some ports you can leave open.

Incoming Connections:

Port Number / Protocol for reason:

- 993 / tcp & udp for IMAP (to receive emails)
- 143 / tcp & udp for Insecure IMAP

- 110 / tcp for POP3 (another way to receive emails)
- 22 / tcp for SSH (Secure connection from machine to machine)
- 9418 / tcp for GIT (version control system)

Outgoing Connections:
Port Number / Protocol for reason:
- 80 / tcp for HTTP
- 443 / tcp for HTTPS (Secure HTTP)
- 993 / tcp & udp for IMAP (to receive emails)
- 143 / tcp & udp for IMAP
- 53 / udp for DNS
- 21 / tcp for FTP
- 465 / tcp for SMTP
- 25 / tcp for insecure SMTP
- 22 / tcp for SSH
- 9418 / tcp for GIT

Step 5: Saving your firewall configuration
Run the following command to save your configuration settings and restarting your firewall:

```
# iptables -L -n
#iptables -save / sudo tee /etc/sysconfig/iptables
# service iptables restart
```

Applying ACL

An ACL (Access Control List) is a set of rules that is usually used to filter network traffic.

ACLs can be configured on network devices with packet filtering capabilities such as routers and firewalls.

ACLs contain a list of conditions that categorize packets and help you determine whether to allow or deny network traffic.

Types of ACL:

i) Standard access lists:

Allow you to evaluate only the source IP address of a packet. Standard ACLs are not as powerful as extended access list, but they are less CPU intensive for the device.

ii) Extended access lists

Allow you to evaluate the source & destination IP addresses, the type of layer 3 protocol, source and destination port and other parameters.

- setfacl and getfacl are used for setting up ACL and showing ACL respectively.

List of Commands for seting up ACL:

1) To add permision to for user

```
setfacl -m "u:user:permissions" /path/to/file
```

2) To add permissions for a group

```
setfacl -m "g:group:permissions" /path/to/file
```

3) To allow all files or directories to inherit ACL entries from the directory it is within

```
setfacl -dm "entry" /path/to/dir
```

4) To remove a specific entry

```
setfacl -x "entry" /path/to/file
```

5) To remove all entries

```
setfacl -b /path/to/file
```