LINUX

**Linux File System Structure :-**

**1. /bin**:-bin -stands for binary.s.

Binary is a file which contains the compiled source code.

We can also call it as executable, because it can be executed on the computer.

/bin is a sub-directory of the root directory in Unix/Linux OS.

this directory contains basic commands which is enough for the minimal system function

ex :-ls, cat, cp

2.**/sbin:** system binaries or super user binaries.

 This folder contains commands which are required for changing system properties.

Ex:-adduser,reboot,shutdown0

3**. /boot:**

The contents are mostly Linux kernel files and bootloaderfiles(files needed to start up the operating system)

4.**/dev :**

This contains device files

This file represents your speaker device,keyboard

5. **/etc:**-

it contains all system related configuration files in here or in its sub-directories

A "configuration file" is defined as a local file used to control the operation of a program;

it cannot be an executable binary.

Ex:-adduser.conf, theme config

**6. /cdrom:**-

directory is a standard practice to mount cd, but not necessary. We use media and mnt to mount anything these days

In Linux and other Unix-like operating systems, **/cdrom** is commonly used as the default directory where the contents of a CD or DVD inserted into the computer's optical drive are mounted.

When you insert a CD or DVD, the system may automatically mount it to the /cdrom directory (or it could be mounted manually).

You can view the contents of the disc by navigating to this directory in the terminal.

**7 ./home :**-

The home directory can be said as a personal working space for all the users except root.

There is a separate directory for every user.

 For example, two users ‘satyam' and ‘jack' will have directories like "/home/satyam“ and "/home/jack“

**8 ./lib :-**

The /lib directory contains **shared libraries** and **kernel modules** that are necessary to boot and run the system.

These libraries are used by essential system binaries located in directories like /bin and /sbin.

**9 /media :**-

When you connect a removable media such as USB disk, SD card or DVD, a directory is automatically created under the /media directory for them. You can access the content of the removable media from this directory.

**10 /mnt –**

Mount directoryThis is similar to the /media directory but instead of automatically mounting the removable media, mnt is used by system administrators to manually mount a filesystem

**11 /opt** –

Optional softwareTraditionally, the /opt directory is used for installing/storing the files of third-party applications that are not available from the distribution’s repository.

**12 /proc** :-

It contains useful information about the processes that are currently running.It could be used for obtaining information about a system, we can also edit the config files related to kernel here.

**13 /root :-**

it works as the home directory of the root user. So instead of /home/root, the home of root is located at /root. root directory (/) is different from root user directory.

**14 /tmp :-**

this directory holds temporary files. Many applications use this directory to store temporary files. Even you can use directory to store temporary files.

the contains of the /tmp directories are deleted when your system restarts. Some Linux system also delete files old files automatically so don’ store anything important here.

**15 /var :**-

stores system-generated variable data files. This includes spool directories and files, administrative and logging data, cache, transient and temporary files.

Ex :-/var/spool contains data which is awaiting some kind of later processing/run :-runtime variable data.

 The purposes of this directory were once served by /var/run, system may use both

**16 /srv :**-

This directory gives users the location of data files for a particular service.

For example, if you run a HTTP or FTP server, it’s a good practice to store the website data in the /srv directory.

**17 /usr :-**

User System Resources. in ‘/usr’ go all the executable files, libraries, source of most of the system programs.

For this reason, most of the files contained therein is readonly (for the normal user).

**18 /sys :-**

allows you to get information about the system and its components (mostly attached and installed hardware) in a structured way.

Ex:-device, kernel, firmware

**19 /snap :-**

The /snap directory is, by default, where the files and folders from installed snap packages appear on your system.

**Types of Files in Linux:-**

In Linux everything is treated as File.

All files are divided into 3 types

**1) Normal or Ordinary files:**

These files contain data. It can be either text files (like abc.txt) OR binary files (like

images, videos etc).

**2) Directory Files:**

·These files represent directories.

·In windows, we can use folder terminology where as in linux we can use directory

terminology.

·Directory can contains files and sub directories.

**3) Device Files:**

In Linux, every device is represented as a file. By using this file we can communicate

with that device.

**Note:** short-cut commands to open and close terminal

**ctrl+alt+t** ➔To open terminal

**ctrl+d** ➔To close terminal

**How to check File Type:**

* 1. **s**

We can use ls command to listout all files and directories present in the given directory.

We can get manual documentation for any command by using man.

man ls

It provides complete information about ls command.

Various options of ls Command:

**1) ls**

It will display all files and directories according to alphabetical order of names.

**2) ls -r**

It will display all files and directories in reverse of aplhabetical order.

**3) ls | more**

To display content line by line

(To come out we have to use q)

**4)ls -l**

To display long listing of files

**5) ls -t**

To display all files based on last modified date and time. Most recent is at top and old

are at bottom.

**6) ls -rt**

To display all files based on reverse of last modified date and time. Old files are at top

and recent files are at bottom.

**7) ls -a**

a means all

To display all files including hidden files. Here . and .. also will be displayed.

**8) ls -A**

A means almost all

To display all files including hidden files except . and ..

**9) ) ls -h**

display in human readable format

**10) ls -R**

·R means Recursive.

·It will list all files and directories including sub directory contents also. By default ls

will display only direct contents but not sub directory contents.

Eg: All the following commands are equal

$ ls -l -t -r

**6. mkdir:-**

We can create directories by using mkdir command.

1) mkdir dir1

To create a directory

2) mkdir dir1 dir2 dir3

To create multiple directories

3) mkdir dir1/dir2/dir3

To create dir3. But make sure dir1 and in that dir2 should be available already.

4) mkdir -p dir1/dir2/dir3

-p means path of directories.

All directories in the specified path will be created.

First dir1 will be created and in that dir2 will be created and within that dir3 will be created

**7.  rmdir :-**

We can remove directories by using rmdir command.

1) $ rmdir dir1

To remove empty directory dir1

2) 2. $ rmdir dir1 dir2 dir3

To remove multiple empty directories

Note: rmdir command will work only for empty directories. If the directory is not empty then we

will get error. We cannot use rmdir for files. Hence the most useless (waste) command in linux is

rmdir.

8. rm :-

to delete files rm file1 file2

To delete non empty folders

**9.  Copy  (cp)**

1) To Copy from File1 to File2 (File to File)

$ cp source\_file destination\_file

$ cp file1 file2

Total content fo file1 will be copied to file2.

If file2 is not already available, then this command will create that file.

If file2 is already available and contains some data, then this data will be over write with file1

content.

2) To Copy File to Directory:

$ cp file1 file2 output

file1 and file2 will be copied to output directory.

Here we can specify any number of files, but last argument should be directory. output directory

should be available already.

3) To Copy all Files of One Directory to another Directory:

$ cp dir1/\* dir2

All files of dir1 will be copied to dir2 But dir2 should be available already.

4) To Copy Total Directory to another Directory:

$ cp dir1 dir2

cp: -r not specified; omitting directory 'dir1'

Whenever we are copying one directory to another directory,

compulsory we should use -r option. 🕐 $ cp -r dir1 dir2    total dir1 will be copied to

dir2

5) To Copy Multiple Directories into a Directories:

$ cp -r dir1 dir2 dir3 dir4 dir5

dir1,dir2,dir3 and dir4 will be copied to dir5

**10. mv  :-**

Moving and Renaming Directories:

Both moving and renaming activities can be performed by using single command: mv

**1) Renaming of files:**

$ mv oldname newname Eg: $ file1.txt file2.txt

file1.txt will be renamed to file2.txt

2) Renaming of Directories:

$ mv dir1 dir2

dir1 will be renamed to dir2

3) Moving files to directory:

$ mv a.txt b.txt c.txt output

a.txt,b.txt and c.txt will be moved to output directory.

**--------Creation of files -----------**

Creation of Files:

In Linux, we can create files in the following ways:

1. By using touch command (to create empty file)
2. By using cat command
3. By using editors like gedit, vi, nano etc

**9. cat :-**

cat > file1.txt

Eg:

$ cat > file1.txt

Hello Friends

Listen Carefully

Otherwise Linux will give Left and Right ctrl+d To save and exit

If file1.txt is not already available, then file1.txt will be created with our provided data.

If file1.txt is already available with some content, then old data will be over written with

our provided new data.

Instead of overwriting, if we want append operation then we should use >> with cat command.

cat >> file1.txt extra content ctrl+d

**>> for appending**

) If we are using Touch Comamnd, but the File is already available then what will happend?

The content of the file won't be changed. But last modified date and time (i.e., timestamp) will be updated

**10. Touch :-**

touch command is a way to create empty files (there are some other mehtods also).

You can update the modification and access time of each file with the help of touch command.

 creating files using touch

touch file1   ------> change timestamp ( both access and modify time)

touch -a file1    -----> change access file of file1

touch -m file1 ------> change modify time of file1

touch -r file1 file2 -----> use file1's timestamp as reference and change timestamp of file2

                                          ( now file2 timestamp will change and become same as file1 )

try :- touch -r file2 -a file1   and observe using stat command what happens

**11. cat :-**

$ cat < file1.txt

-n  option to give line numbering to file content

-b to give numbering to all lines apart from blank lines

 We can view multiple files content at a time by using cat command.

$ cat file1.txt file2.txt file3.txt

Various utilities of cat Command:

1. To create new file with some content

$ cat > filename data ctrl+d

1. To append some extra data to existing file

$ cat >> filename extra data ctrl+d

1. To view content of file

$ cat < filename or $ cat filename

1. Copy content of one file to another file

$ cat input.txt > output.txt

1. To copy content of multiple files to a single file

$ cat file1.txt file2.txt file3.txt > file4.txt

1. Merging/appending of one file content to another file

$ cat file1.txt >> file2.txt

**12. tac :-**

It is the reverse of cat.

It will display file content in reverse order of lines. i.e first line will become last line and last line will become first line.

This is vertical reversal.

1. **rev :**

rev means reverse. Here each line content will be reversed. It is horizontal reversal.

 cat command will display total file content at a time. It is best suitable for small files. If the file contains huge lines then it is not recommended to use cat command. We should go for head, tail, less and more commands.

1. **head :**

We can use head command to view top few lines of content.

✽ head file1.txt

It will display top 10 lines of file1.txt.

10 is the default value of number of lines.

✽ head -n 30 file1.txt OR head -30 file1.txt To display top 30 lines of the file.

Instead of 30 we can specify any number.

✽ head -n -20 file1.txt

To display all lines of file1.txt except last 20 lines.

✽ head -c 100 file1.txt

To display first 100 bytes of file content

1. **tail :**

We can use tail command to view few lines from bottom of the file. It is opposite to head command.

✽ tail file1.txt

Last 10 lines will be displayed.

✽ tail -n 30 file1.txt OR tail -30 file1.txt OR tail -n -30 file1.txt It will display last 30 lines.

✽ tail -n +4 file1.txt

It will display from 4th line to last line

✽ tail -c 200 file1.txt

It will display 200 bytes of content from bottom of the file.

1. **more :**

We can use more command to view file content page by page.

✽ more file1.txt

It will display first page.

Enter     To view next line

 Space Bar     To view next page

 q     To quit/exit

✽ more -d file1.txt

-d option meant for providing details like --More--(5%)[Press space to continue, 'q' to quit.]

1. **less :**

 By using more command, we can view file content page by page only in forward direction.

If we want to move either in forward direction or in backward direction then we should go for less command.

**d**  To go to next page.(d means down)

**b** To go to previous page. (b means backward)

**Creation of Hidden Files and Directories:-**

If any file starts with '.' , such type of file is called hidden file.

If we don't want to display the files then we have to go for hidden files.

Hidden files meant for hiding data. All system files which are internally required by kernal are hidden files.

We can create hidden files just like normal files, only difference is file name should starts with dot.

touch .securefile1.txt

 cat > .securefile1.txt

Even by using editors also we can create hidden files.

We can create hidden directories also just like normal directories. mkdir .db\_info

Note: By using hidden files and directories we may not get full security. To make more secure we have to use proper permissions. For this we should use 'chmod' command.

**Interconversion of Normal Files and Hidden Files:**

Based on our requirement, we can convert normal file as hidden file and viceversa.

mv a.txt .a.txt

We are converting normal file a.txt as hidden file.

 mv .a.txt a.txt

Similarly directories also

mv dir1 .dir1

mv .dir1 dir1

**---------File Permissions----------**

**File Permissions describe the allowed operations by various users.**

**With respect to file permissions, all users are categorized into the following 4 types.**

**User Categories:**

**user/owner     Represented by 'u'**

**group   Represented by 'g'**

**others     Represented by 'o'**

**all     Represented by 'a'**

**Permission Types:**

**Number    Permission**

**0  No permission**

1. **Execute**
2. **Write**
3. **Execute and Write**
4. **Read**
5. **Read and Execute**
6. **Read and Write**
7. **Read, Write and Execute**

Operations related to permissions:

We can perform the following 3 operations.

+ Add a particular permission to user|group|other|all

-    Remove a particular permission to user|group|other|all

= Assignment a particular permission to user|group|other|all

**. chmod Command:**

chmod means change mode.

We can use chmod command to change file or directory permissions.

Syntax: $ chmod <user\_category><operation><permission> file\_name/directory\_name

Eg: For user add execute permission,for group add write permission,for others remove read permission

$ chmod u+x,g+w,o-r demo.txt

Note: Only owner and super user (root) can change file permissions.

How to check Permissions of existing File: By using ls -l command:

Total 9 permissions. First 3 are user permissions, next 3 are group permissions and next 3 are others permissions.

user permissions: rwuser can perform both read and write operations but not execute operation

group permissions: r--

group members can perform only read operation and cannot perform write and

execute operations

others permissions: r-other members can perform only read operation and cannot perform write and execute operations.

User Permissions + Group Permissions + Others Permissions order is important

Eg 1: $ chmod u+x demo.txt adding execute permission to the user

Eg 2: $ chmod u+w,g+rw,o+r demo.txt adding write permission to the user adding read and write permissions to the group adding read permission to the others

Eg 3: $chmod u+x,g-w,o+w demo.txt adding execute permission to the user removing write permission from the group adding write permission to the others

Eg 4: $ chmod u=rw,g=rw,o=r demo.txt

Now user permissions: rwgroup permission: rwothers permission: r--

Eg 5: $ chmod a=- demo.txt

Now user permissions: --group permission: --others permission: ---

Eg 6: $ chmod a=rwx demo.txt

Now user permissions: rwx group permission: rwx others permission: rwx

**Read Permission to the File:-**

If the file not having read permission then we are not allowed to view content of the file. Hence cat, head, tail, more, less commands won't work.

**Write Permission to the File:-**

If the file not having write permission, then we cannot modify the content of the file.

**Execute Permission to the File:-**

If the user not has executed permission on any file, then he cannot execute that file as a program.

**Read Permission to the Directory:-**

If the user has read permission on any directory, then he can list out the contents of that directory. i.e he can use ls command.

**Write Permission on the Directory:-**

If the user has write permission on any directory, then he is allowed to modify the content of that directory. i.e he can add new files and remove existing files.

**Execute Permission to the Directory:-**

If the user not has executed permission on any directory, then he is not allowed to enter into that directory. i.e he cannot use cd command.

**Note:-**  If the user not having read permission on any file, then he cannot execute that file even though he has executed permission.

**Linux vs Security:**

The virus files usually created by others.

others are not having execute permission on our directories. Hence others are not allowed to add virus files to our directories.

Hackers are not having executed permission on our directories. Hence they cannot read our file data.

Because of this, Linux is considered as more secured operating system. Linux follows 2 levels of security.

1st level: login with credentials

2nd level: File and Directory permissions

**Note:**We are using permission types as r,w,x and these are considered as symbolic permissions. But we can also specify permissions by using octal number, such type of permissions are called numeric permissions.

**Numeric Permissions:-**

We can specify permissions by using octal number.

Octal means base-8 and allowed digits are 0 to 7.

          0  000     No Permission

         1    001     Execute Permission

          2    010     Write Permission

         3     011     Write and execute Permissions

         4     100     Read Permission

         5     101     Read and execute Permissions

         6     110     Read and write Permission

         7  111     Read, Write and execute Permissions

**Note:**

4 Read Permission

2 Write Permission

1 Execute Permission

It is more easy to remember

    5     4+1     r-x

    3     2+1     -wx

    6     4+2     rw

**chown :-**

chown means change owner.

Only root user can perform this activity.

chown root demo.txt

Now the owner of demo.txt is root.

**. nano :-**

It is command line editor.It can be used to create new files and edit content of existing files

 ctrl+g (F1) Display this help text

 ctrl+x (F2) Close the current file buffer / Exit from nano

 ctrl+o (F3) Write the current file to disk

 ctrl+r (F5) Insert another file into the current one

 ctrl+w (F6) Search forward for a string or a regular expression

ctrl+\ (M-R) Replace a string or a regular expression

ctrl+k (F9) Cut the current line and store it in the cutbuffer

 ctrl+u (F10) Uncut from the cutbuffer into the current line

But main important options:

ctrl+o To save content

 ctrl+x To quit from the editor

**\*\*Vi:\*\***

   - Vi is a classic text editor that has been around for a long time.

   - It has two modes: command mode (for navigation and manipulation) and insert mode (for actual text entry).

   - Basic Commands:

     - To open a file: `vi filename`

     - Switch to insert mode: Press `i`

     - Save and exit:

       - Press `Esc` to ensure you are in command mode.

       - Type `:wq` and press `Enter` to save and exit.

Mkdir -p f1/f2/f3

Cd f1/f2/f3

Touch t1.txt

Cat > t1.txt

Nano f1.txt

Vim f1.txt

Chmod 766 f1.txt

1)F1/F2/F3

1. F3 - file
2. Cat - overwrite
3. Nano