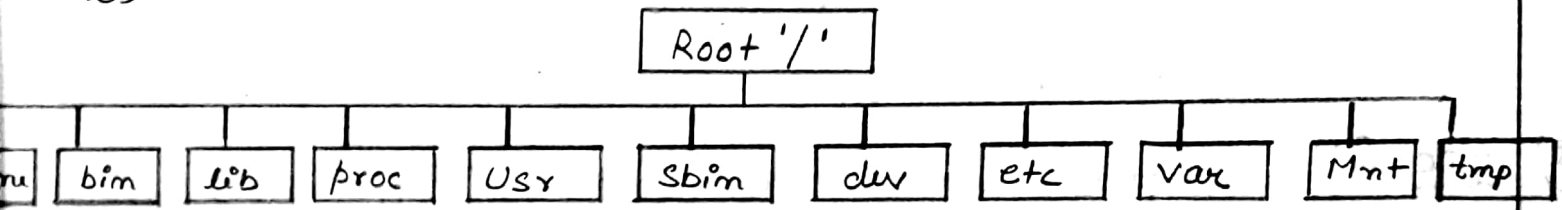
File System
&
Concepts of Blocks
In Linux

File System in Linux :-

Linux uses a hierarchical file structure, i.e. directories are linked in the form of a family tree.

All the H/W devices (I/O devices, storage devices) are also treated as files.

In Linux, the root directory ('/') is the & all other directories are it sub-directories.



(Standard file system in Linux).

Various Directories are :-

home :- It is the user-area for maintaining all user related files.

bin :- Contains all the executable file. Contains commands needed during booting. Other commands required by the regular user are also maintained in this directory.

Lib :- Contain shared Libraries & routines needed by different users.

proc :- contains specials files that either extract information from or send information to the kernel.

usr :- Contain Linux utilities & libraries. In earlier version of Linux it is used to store user's home directory.

sbin :- contains system admin commands intend for the root users of a system.

Dev :- Contains special device files pertaining to the peripheral devices attached to the system.

Etc :- Contains all the configuration files that are local to the Machine.

var :- contains variable data files.

mnt :- Reserved for temporarily mounted file system such as:- CD-ROM, Floppy etc.

Tmp :- Contains ~~temporarily~~ temporary files that are not required after a program has been completed.

Different types of files in Linux :-

Normal files :-

Used for storing data, such as text information & programs.

Directory file :-

Stores information about directory, it's relationship with other directories, file & sub-directories under it & related access permission.

Special files :-

Represent various routines in the kernel from Block & character special files represent device drivers on which terminals, printers & disk drivers are controlled.

Symbolic Link files :-

Contains the location / & full path name of actual files to which they point.

Pipefiles :-

Used as pipeline channel of information to passed from one process to another. They hold their

contents only till the receiving file has not read that information.

Relative path name :-

If the file is in a directory near the working directory a relative path may be used. It gives the location of a directory / file, relative to the current working directory to do this following convention are used.

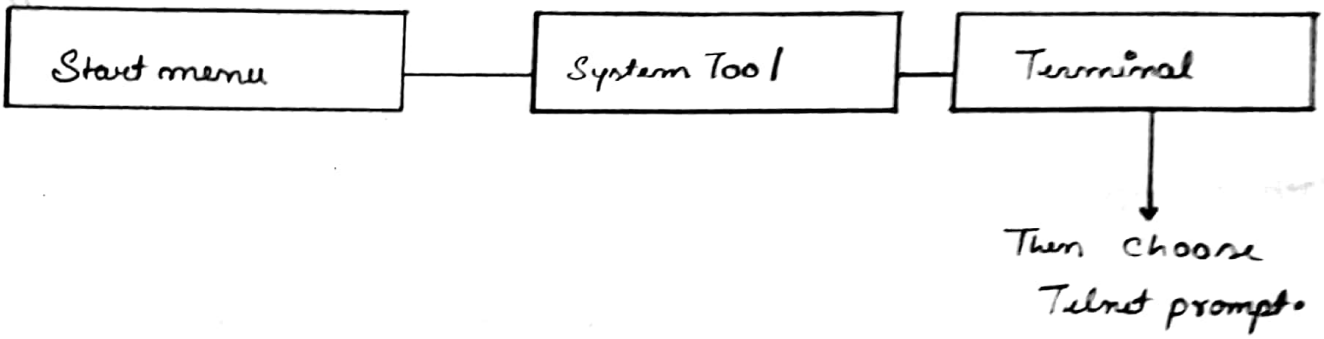
1. Single dot → Current working directory.
2. Double dot → Parent working directory.

Example :- If your working directory is /home/My Dir, the file second.c under the directory /home/Demo may referred as .../Demo/Second.c

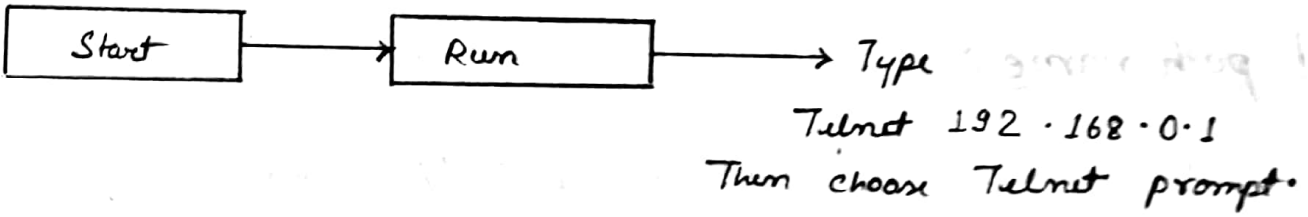
Files & Directory :-

* The commands are types at the shell prompt ($ or #) & the enter key is pressed to view the output of the typed command.
* Prompt is a symbol that appears at the start of a command line.

In a GUI based Linux (Linux 8.0) this prompt can be got by choosing.



To connect server Linux :-



File Handling in Linux

* Naming convention for files.

File naming in Linux follows the convention below:-

* A name can be upto 256 characters.
* A name cannot contain spaces
* Special characters are allowed only the following are use :-
  1. Lower-case letter (a-z)
  2. Upper-case letter (A-Z)
  3. Number (0-9)
  4. Underscore (_), priod (.) & comma (,).

Note:- Linux in highly case sensitive, i.e. lower case & upper-case letters are not treated a like. This holds good for the Linux command as well.

* Path name :-

A file or directory in Linux may be referred to as follows.

* Full path name :-

List each directory, starting from '/' (root) down to the file itself. Each directories & file must be separated by a '/'.

Example:- Suppose my dir is stored the home director then you can refer to a file named (first.c contain in My dir using.

/home/My dir/first.c

* Partial path name :-

If the file is in the working directory or in a sub-directory by below the working directory, the names of higher may i.e. 0

Example:- If your working directory is / home / My dir then file "first . c" is that directory may be reserved simply by first.c .

Commands :-

ls :- function :- list the files in current directory . The names are stored in alphabetical order.

Ex → ls - a

Effect :- Lists all the files including hidden files all the directory.

Mk dir :- function - creati a directory.

Ex → Mk dir Abhay

mkdir

mkdir / Ankit ,

Effect :- Creati a directory Abhay in the current working directory.

Creati a sub-directory 'ankit' under the directory abhay.

mv :- function - moves or renames a given file or directory.

Example:- mv my file our file
(Rename myfile to our file).

mv test. /

i prompt the user before overwriting
u moves only when the source file is ~~newer~~ newer than the destination file or when the destination file is missing.

→ move the test file named test to an existing directory name 'new dir' (existing under the current directory.

rmdir :- Delete / remove the directory provided it is empty.

An error message is displayed, if the specified directory contain files or sub-directories

Options:- p removes the parent along with all it's sub-directory.

V outputs a diagnostic for every directory that is processed.

Telnet 192.108.0.1 prompt
Login - bca password - student $

Command

cat - function - displays the contents of a file.

Example:- cat file.txt

cat > Test

Effect :- Displays the content of file.txt '>' symbol can be used to create a file named test. We would be prompted to enter the contents for the file after you press the enter key. Type the desired data & press. Ctrl + D to terminate the command line.

cd - function → changes the current working directory to another directory.

example:- cd..

Effect:- changes the directory to the parent of the current directory.

pwd - function → Prints full path of the current directory on the command line.

example :- pwd.

Effect:- prints the full path of the current dire-

~~How on the command line:~~ working directories

newdir

rmdir mydir (Remove the empty directory 'mydir')

rmdir

/mydir/sub1/sub2

(Remove sub2, sub1 mydir i.e. all the subdirs are deleted along with the parent directory)

rm -> Delete the specified file.

Options

f -> ignores non-existant files & does not prompt the user before delete.

i -> prompt the user before actually deleting the file

r -> Removes the contents of directories.

v -> Explains what is being done.

Example:-

rm file.txt

rm file.text
file2.text

(Remove f1.text & file2.txt. You can delete multiple files by separating the file names using the space character.)

File Related Command :-

Cat :-

It is the Multi-purpose command using this we can create a file read the contents of file, append the contents of a file to the end of another file, Overwrite the contents of a file with the contents of another file.

Syntax:- Create a file.

Example:- $ cat > abc ↵

> → Re-direction operator
>
> → Name of your file.

Read the contents of ~~related~~ ~~ff~~ created file.

$ cat abc ↵

Read the content of more than one file at a time.

$ cat abc PQR

Each file name is separated with blank space.

Overwrite the content of a file with the contents of another file.

cat abc > asd ↵

Append the contents of a file to end of the contents of another file.

Syntax:- cat abc >> PQR ↵

rm :-

It is used to remove the existing file, when we execute this command we cannot recovered the deleted file.

$ rm abc ↵ (It removes the file abc ↵

$ rm -i ----> (condition) abc

Are you want to remove ? Y or N.

When we remove the file after the confirmation of user then we use rm command with option i.

We can remove more than one file at a time.

$ rm abc_1, abc_2 abc_3 ↵

cp (copy) :-

This command is used in Linux to copy the contents of file to another file.

Syntax :- $ cp abc pqr ↵ (abc → Source, pqr → Destination)

the contents of file abc is copy into file pqr. In this case, the contents of file abc is overwrite the contents of file pqr.

mv (rename) :-

It is used to rename the existing file & directory. When we execute mv command then the Linux rename the file (existing file) when another name. We can access the file with it's new name not by old name.

Syntax :- mv abc pqr ↵

We re-name the above file abc by pqr, then we can't call it by user.

* cat pqr ↵ (It show the contents of abc).
* cat abc ↵ (file does not exist).

wc (word count) :-

It is a useful command of Linux. It counts the no. of lines, word & character in the specified file.

Syntax:- $ wc ABC ↵

Example 3, 9, 24

ab sb rb

bc bc bc

st uv wx.

$ wc -l abc ↵
= 3 (-l represents line)

$ wc -w abc ↵
= 9 (-w represents word)

$ wc -c abc ↵
= 24 (-c represents character).

grep (search) :-

It is a command which is used to search the specified input globally for a match with the supplied pattern & display it. It is known as "Globally

search Regular Expression & Print it."

Example:- $ grep ab abc ↵ (If word is search).

$ grep "ab is" ↵ (If sentence is search then we use " ").

sort :-

Sort command is used in Linux to arrange the each line of the file in ascending order on the basis of ASCII value of the character of each line. Here, Linux compare the first character of first line to the first character of 2nd line & so on.

Example:- $ sort ABC ↵

ab is that

bc is she.

ab is he.

sort -o :-

Using this option we can store the sorted contents of a file to new file.

sort - o abc1 abc

↓ New file ↓ sorted file.

sort -u :-

Using this option we allow the unique value of the file for sorting.

Example:- sort -u abc ↵

Output:-
ab is
bc is.

ab is
ab is
ab is
bc is

head :-

head command display the contents of file from top to bottom. It accept an option -n (n = 1, 2, 3, ... n) which specify a line count & display the content on screen. When it is to used without an option it display the 1st 10 lines of the file.

Example:- * $ head abc ↵

(It display the 1st 10 line of the file).

* $ head -10 abc ↵

(It display the specific 10 line of the file).

tail :-

It is the counter part of the head command. It display the contents of a file from bottom to top. It accept an option -n where (n = 1, 2, 3, 4, 5 .....) which specify a line count & display the contents on screen.

Example :-

* $ tail abc ↵

(It display the bottom to up to line of the file).

more :-

We can view a file page by page. On executing more command one page of file contents are display on the screen, at the bottom of the screen you will also see the file name & the percentage of the file that has been display. More use the 'space bar' or 'f' to scroll forward page at a time & 'b' used for backward one page at a time.

Syntax :- more abc ↵

ls :-

It display the name of all files present in any specified directory.

Example:- $ ls ↵

lp :- (print)

It is used to send the users print job to the print queue. When several users make a request to print their job, request are put on a queue & linux printed according to their priority. When we submit the jobs for printing using the lp command, it returns a request id that can be used to keep track or to cancel it if required. When we cancel the print request, our jobs gets remove from the print queue.

Example:- $ lp abc ↵

cancel :-

If for some reason we want to cancel the submitted job for printing, there we use cancel

command. We must know the request id for the cancel job.

lpstat :-

It show the status of our print job whether it is successful or not. It show status of all printing job that we assign to system using lp command.

Syntax:- * lpstat ↵

(It show the status of all printing job assign to the system).

* lpstat 2314 ↵

(It display the status of specific job whose id is 2314).

cut :-

It cut or pick-up a given no. of characters or field from the specified file. If we have a database of employee & want to see the name & address of all employee then we use cut command.

Example:- cut -c 10 abc ↵ (-c → character)

cmp :-

It compares two files byte by byte & the location of the 1st mismatch is display to the screen.

Example:- $ cmp abc1 abc2 ↵

comm :-

It is used to display the common field present in both files. It also display their differences. The file must be sorted before using comm command.

Example

cat > abc1	cat > abc
Sonu	Rohit
Sohan	Kavi
Amit	Shahi
Babla	Rohan
Rohan	Aman
Aman	

sort abc1

sort abc

comm abc1 abc ↵

Output:-

Somu	Rohit	Rohan
Sohan	Kavi	Aman
Amit	Shahi	
Bablu		
(Unique value of file, abc,)	(unique value of file abc.)	(common value of both file.

alias :-)

Bash shell support the use of aliases that we assign short hand name (user-defined name) to frequently used command. It is done with alias statment. It is used as follow:-

Syntax:- $ alias common = comm ↵

common ↓ user-defined

comm ↓ System define.

unalias :-)

We can remove the aliases created for any command.

Example:- $ unalias cat ↵

(It removes all aliases creati from command cat)

history :-

The history command display the event no. of all previously command executed by user. The Bash & corn support a versatile history features that treats a previous command as an event & associate with an event no. Using this no. you can recall previous command if required. The exclamation symbol (!) is used to repeat previous command.

Example:- $ history ↵

Event no:

← 112 cat

121 comm

131 cmp

141 history

$! 131 ↵ (for specific).

Directory Related Command :-

pwd :- (Present working directory)

When we execute the pwd command on shell

prompt then the Linux return the name of your working directory.

Syntax:- $ pwd ↵

Example:- home / ABC
↓ root directory. ↳ Sub-directory.

mkdir :-

Using this command we can create any directory as per our requirement.

Syntax:- $ mkdir directory name ↵

Eg:- $ mkdir abc ↵
↳ dir name.

Creating more than one directory at a time.

mkdir abc , def , gh ↵

Create a directory in any directory.

mkdir abc / abc1 ↵

cd (change directory) :-

Using this command we can make a directory as a working directory.

Syntax:- $ cd directory name ↵

Example:- cd abc ↵

cd.. :-

Using this we can make a parent directory as working directory.

Example:- mkdir abc/PQR/RST

cd abc/PQR/RST ↵

abc/PQR/RST $ cd.. ↵

abc/PQR $

cd./ :-

Using this command we return to our home directory.

Example:- cd./ ↵.

rmdir :-

We can remove a directory from rmdir command and the deleted directory must be empty.

Example:- rmdir abc ↵

rmdir -p :-

Using this we can remove a directory as well as

it also removes the parent directory of a directory.

Syntax:- rmdir -p abc/pqr/rst.

File permission :-

ls -L :- This command display the description of file & sub-directory present in a directory.

$ ls -L ↵

-rw-r--r-- 1 Kumar user 1914 May 4 10:34 : 45 abc

- File Type
- 1 No. of Link
- Kumar owner name
- user group-name
- 1914 size in byte.

- - → ordinary file.
- d → directory file.
- r → read
- w → write
- - → no permission
- x → Execute.

User are of three (3) types :-

1) owner → u
2) group → g
3) other → o

The 1st column show the type of file, it may be an ordinary file (-) or directory file. 2 to 4 column it signify the permision owner user of a file. 5 to 7 column. It show the permision for a group user (r--) 8 to 10 column. It show the permission for other user (r--)

11 column: It show the no. of link of a file (1).

12-16 column. It show the name of user who creat the file. (kumar).

17-20 column. It shows the ~~size of a file in byte~~ group name of a file (user).

21-24 column. It shows the size of a file in byte (1914).

May 4 10:34:45

It show the date & time of creat a file.

abc -> abc is the name of file.

chmod :-)

This command is used to set the permission of a file to other (group user, other user). It can be run by the user (owner) & by the super-user. We can use this command in two ways :-)

1) Relative Manner.
2) Absolute Manner.

Relative Manner :-)

chmod only change the permission specified in the command line leave the other permission unchanged.

```
ls -L ↵
ls -L abc ↵
```

(Here we assign write permission to group-user).

```
chmod g+w o-r abc ↵
```

we return back the read permission from other user on file.

+ → assign the permission.
- → Revoke " " .

```
chmod g+rw abc
chmod go+r abc ↵
```

or

chmod g+r o+r abc ↵

Absolute Manner :-

all ← owner → read + write = 6
all ← group → read + execute = 5
other → read = 4

read → 4
write → 2
execute → 1
no permission = 0

read + write = 4+2 = 6

read + execute = 4+1 = 5.

write + " = 2+1 = 3

chmod 654 abc ↵

chmod 770 abc ↵

Piping :-

A single linux command does not able to perform a problem we require more than one command to perform that problem.

The linux provide a piping facility which connects commands. It can be useful to re-direct the output of one (1) command so that it become the input of another command. Here, we use o

pipe character (|) to perform piping for a command.

Filter :-

It is a program (command) which receive a flow of data from standard input process it & sent the result to the standard output.

For Example:-

grep, wc, sort etc.

grep ab abc ↵

grep is a filter which accept ab grep & perform searching operation in file abc if it is found then display the result.

File System :-

A file system is a group of file which is created on our disk. When we install a linux O.S on our system then it assign a fixed portion of space of Hard disk. There must be one file system in a Hard disk. The disk space located to a linux file system is made up blocks. Each blocks

are 512 bytes, 1024 bytes or 2048 bytes. The block size depends upon how the file system is implemented for a particular application. All the block belonging to the file system are logically divided into 4 (four) parts.

Boot Block :-

The 1st block of the file system is known as Boot Block. It contain a small Boot strap program (Master Book Record). The program is loaded into Memory when the system is Booted. It loads the kernel into Memory.

Super Block :-

The Boot Block is followed by Super Block. It is the balance-sheet of every Linux File system. It contain global file information about disk uses & availability of data block & inodes.

It contain the following information.

The size of the file system.

i) The block size used by the file system.
ii) The no. of block free data blocks available.
iii) The no. of free inodes available.
iv) The last time of updating.
v) State of file system it may be clean or dirty.

Inode Table :-

The information related to all created file is stored in an inode table on the disk. For each file there is only one inode entry in the Inode Table. Each entry is 64 bytes in size & contain the details of that file. It contains following details :-

i) Owner of the file.
ii) Types of file.
iii) Group to which the owner belong.
iv) Data & time of last access of the file.
v) No. of links to the file.
vi) Size of the file.
vii) Address of block where the file is physically present atleast.

Data Block :-

All files data are stored in data block. It contain the actual file contents. All allocated block can belong to only one (1) file in the file system. This block cannot be used for storing any other files contents.

Input - Output Re-direction :-

> → Output re-direction operator.

< → Input " " " .

In all O.S there is a standard input & output device the output device is the display screen & standard input device is the keyboard. Linux command get their input from standard input device & send their outputs to the standard output device. Linux allow us to change the standard input & output using re-direction operator.

- There are 3 (three) types of re-direction operator used in Linux:-

Output re-direction operator (>)

It implies re-direction of output using this we can send the output of a command or program file to a file or device such as Printer.

Example:- sort abc ↵

sort abc > pqr ↵

Input re-direction operator (<)

It implies re-direction of input. It allow a user to take the input needed for a command from a file rather than the keyboard.

Example:- sort ↵

sort < abc ↵

sort ↵	Output
Rohit	Kavi
Kavi	Mohan
Sohan	Rohit
Mohan	Sohan

3) **Append re-direction operator :-**

Using this operator we can append the contents of a file to another file.

Example:- cat abc >> abc1

Output :-

abcd
efg.

abc1
abc
abcd
efg.

cat abc pqr rst >> abc1 (for more than 1 file).

* **tee :-**

It is a command which handle a character string by duplicating it's input. It's saves one copy in a file & writes the other to the standard output.

$ ls | tee abc | grep pq ↵

The output of ls command become the standard input of tee command. Now it send one copy of the input to grep command using pipe character & store the other copy in a file abc.