CSCRT10: Linux Administration (Core)

UNIT I. Overview of Linux

What is Linux?

Linux is a free and open source Operating System Software developed primarily by Linus Torvald.
Linux's root in Unix: Linux was developed with the help of many unix programmers and wizards across the internet. Linux allows any one with enough knowledge to develop and change the system.
Various linux distributions are availble today. Some of them are
• Ubuntu
• Fedora
• Open SUSE
• RedHat
• Mandrake
• Debian
• Slackware
You can use linux as Server OS or as a Standalone OS on your PC. It is best suited for Server. Linux is a multi user & multi tasking OS.

Common Linux Features:

• Multiuser capability: Multiple users can access the same system resources like memory, hard disk, etc. But they have to use different terminals to operate.
• Multitasking: More than one function can be performed simultaneously by dividing the CPU time intelligently.
• Portability: Portability doesn't mean it is smaller in file size or can be carried in pen drives or memory cards. It means that it support different types of hardware.
• Security: It provides security in three ways namely authenticating (by assigning password and login ID), authorization (by assigning permission to read, write and execute) and encryption (converts file into an unreadable format).
• Live CD/USB: Almost all Linux distros provide live CD/USB so that users can run it without installing it
• Graphical User Interface (X Window system): Linux is command line based OS but it can be converted to GUI based by installing packages.
• Support's customized keyboard: As it is used worldwide, hence supports different languages keyboards.
• Application support: It has its own software repository from where users can download and install many applications.
• File System: Provides hierarchical file system in which files and directories are arranged.
• Open Source: Linux code is freely available to all and is a community based development project.

Adavntages of Linux:

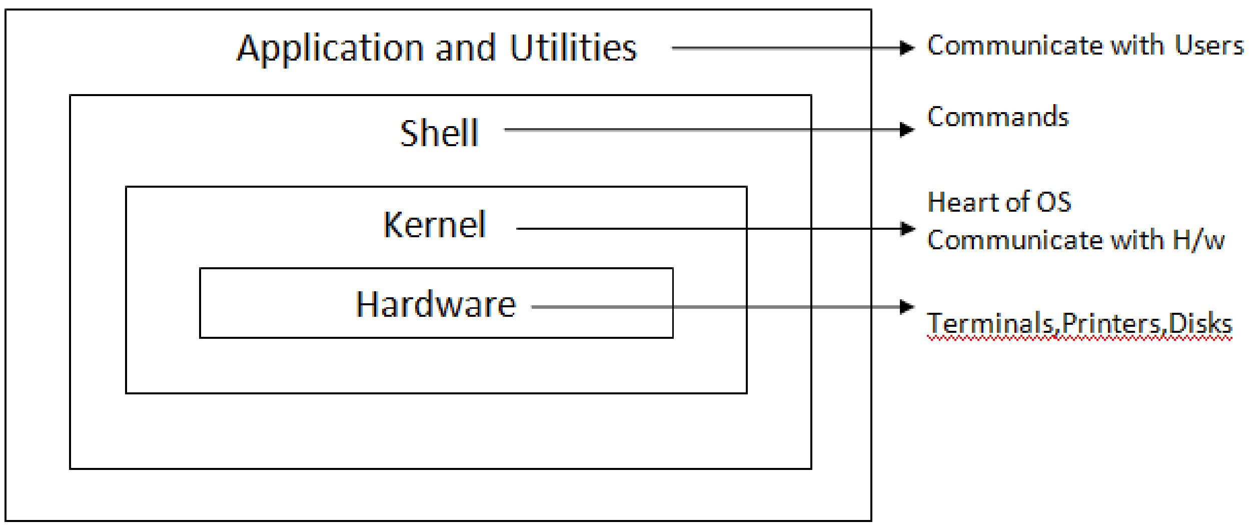
- Free – Most Linux distributions are free, users do not need to pay for a copy. Linux distributions can be freely downloaded and legally installed on as many as you want and freely (and legally) given to other people.
- Open Source – You have access to the source code and can customize Linux to be whatever you want it to be. Linux is easy to install.

- Linux is very stable – Linux systems rarely crash, and when they do, the whole system normally does not go down.
- Linux is not easily affected by computer malware – Infection with virus, spyware, Trojans, and worms is considerably less for Linux. Also security updates normally come much quicker to Linux than other operating systems because so many people are contributing to it.
- Linux does not have a registry like windows. So it is not troubled with registry errors which can slow down a computer.
- Linux will run faster on an old computer. Linux runs great on newer computers as well.
- Linux offers a wide variety from which to choose the distribution.
- Easier to use and install.
- Easier to upgrade.

Disadvantages of Linux:

- Many Windows programs will not run in Linux. Microsoft office, Internet Explorer and many other Windows programs will not run natively in Linux.
- There is a smaller selection of peripheral hardware drivers for Linux.

Unix and Linux Architecture



The functioning of Unix and Linux are managed in 3 levels:

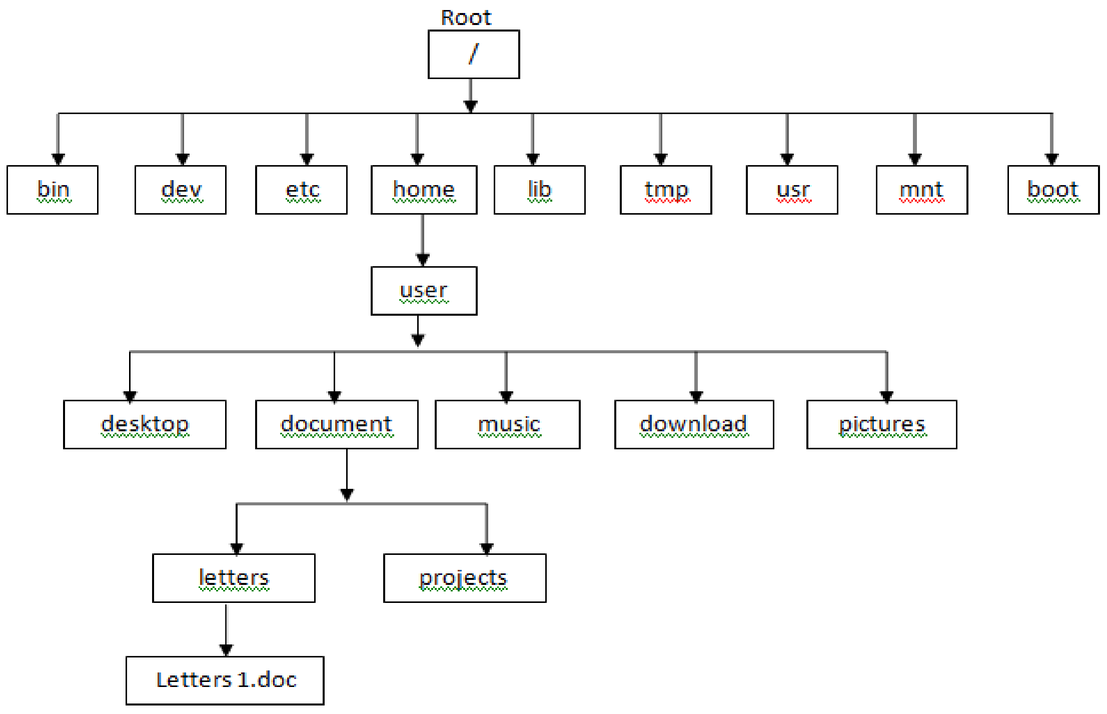
1. Application programs and other utilities are in the outer level. They speak our language.

2. Shell is in the middle layer. It is responsible for the communication between the outer and inner layers. Shell is the command interpreter. It interprets commands that we give and then conveys them to the kernel which executes them.

3. Kernel is the heart of the OS. It interacts directly with the actual h/w. it is responsible for all the major activities of the Linux OS. The Kernel has various functions like managing files, carrying out all the data transfer between the files system and the hardware. It also manages the resources of Linux OS (memory, cpu, etc.).

Linux file System

The Linux file system is the structure in which all the information on the computer is stored. Linux uses hierarchical file system to organize its files. It has a root directory (/) that contains other files and directories. Each file or directory is uniquely identified by its name, directory in which it resides and a unique identifier called **“inode”**.
A full path or relative path can be used to refer a file or directory. In Linux there are several types of files. They are:
1. Ordinary files
2. Directory files and
3. Device files   (In Linux system, the devices are considered as special files.)



bin – This directory contains common linux commands such as ls, dateedtc.(binary files or executable files)

dev – It contains special files corresponding to devices such as printers,disk storage devices, terminals, modems, memory etc.

etc – It contains administrative configuration files.

Home- It contains directories assigned to each regular user with a login accout.

lib – It contains shared library files needed by applications in /bin to boot the system.
tmp – used to store all temporary files.

usr- used to store all files of user accounts.

mnt- used to store mount points.

boot – has bootable Linux kernel and boot – loader configuration files. [GRUB – Grand United Bootloader]

LINUX Commands

Syntax: $ command options arguments

The Linux command is followed by options and a list of arguments. The options can modify the behavior of a command. The arguments may be files or directories or some other data on which the command acts. Every command might not need arguments.
The options can be provided in two ways:
(i) single letter option with a - (eg: -a, -l, -abc)
(ii) full word option with -- (eg: --help)

I. Simple Commands

1. **man** – It is an online help system (manual).
 Syntax : man <commandname>
 eg: man date

2. **info** – Gives help in info format
 eg: info date

3. **date** - To display the current date and time
 eg: date
 o/p: mon Jan 7 10:55:36 IST 2019

4. **cal** – To display the calendar.

```
eg:  cal           - display the calendar for the current month.
     cal –y        - calendar is displayed for the entire current year.
     cal –y 2018   - displays last year calendar.
     cal 06 2018   - calendar of 6th month of 2018 is displayed.
```

5. **more** – the output of a command can be send to the more command, using '|' symbol .
 'more' is a filter for paging through text one screen full at a time.
 eg: cal –y | more

6. **less** – it is a more powerful command than 'more'.
 The 'less' command allows to use the page up and page down keys to scroll back and forth in the output.
 eg: cal –y | less

II. Creating and Viewing files using cat command:

1. **cat** – used to create and view files. Used to concatenate files also.

```
eg:1) cat > music          - to create a file music from keyboard.
                 artist
                 album
                 playlist
                 Ctrl + Z
eg:2) cat music                - to display the contents of the file music.
eg:3) cat music  game > new    - to concatenate multiple files (music, game)
                                 and write into a single file.
eg:4) cat *                    - to display all files.
```

III. File commands

1. **ls** – list directory contents

```
 eg:1) ls                     - displays all files and directories in the current directory.
 o/p:  pictures
       desktop
       documents              - denoted by blue color (Directories)
       downloads
       videos
       music                  - denoted by white color (Ordinary files)


eg:2)  ls  - l                - to display long list with file permission modes. Number of
links, owner, group,
                          size in bytes, date and time of last modification and file name.
 o/p:    drwxr  -xr –x    4 casp casp 4096 NOV 16 12 : 32 Desktop
       –rw –rw –r - -   1 casp casp 69 Jul 17 21 : 01 dance.pl

eg:3)  ls – R          - to list all files and directories recursively from current directories.

eg: 4) ls  - a         - In linux low level configuration file names start with a period.
                         By default   ls does not display those files. – a option is used to
                         display those   hidden files.

eg: 5)  ls – s         - to list by size, print the allocated size of each file in blocks.

eg: 6)  ls – F         -  to list files and file type indicators.

eg: 7)  ls - - hide = D *   -  do not include files beginning with D.

eg: 8)  ls - - help           - to display the help screen of ls.
                          ( - - is used to consider the following letters as a whole world)
```

2. **cp** - used to create copies of an ordinary file.

```
eg:1)   cp   file1   file2       - to copy file1 to file2
eg:2)  cp   file? /data         - to copy the file1, file2  to data directory
eg:3)  cp   file1  ~            - to copy the file to home directory
eg:4)  cp - i  * . txt   /data          - i confirm each copy
```

3. **mv** - to move or rename files

```
eg:1)   mv   file1      file3          - rename to file1 to file3
eg:2)   mv   file3      /data          - move file3 to data directory
eg:3)   mv   file1       ~             -  move to home directory
```

4. **rm**   - To remove a file or directory

```
eg:1)  rm   file1               - to remove file1
eg:2)  rm  *                    -  to remove all files.
eg:3)  rm   -  R                - To delete the entire directory tree. (recursion)
```

5. **cmp**   - To compare 2 files

```
eg:     cmp   file1   file2
```

6. **file** -   To find out the type of the file (text file, executable file, script file etc.)

```
eg:     file    file3
```

File matching meta characters

Meta characters are used to easily refer a group of files.

*****	- Matches any number of characters
?	- Matches any one character
[…]	- Matches any one of the character can also include a range of letters or numbers separated by dash

eg:1) ls f*	- to list all files starting with the character ' f '
eg:2) ls a????	- to list all files starting with the character ' a ' and having 4 more characters
eg:3) ls [ab]*	- to list all files starting with the letter ' a ' or ' b'
eg:4) ls [a-c]*	- to list all files starting with the letter ' a ' or ' b ' or ' c '

IV. Directory commands

1. **pwd** - (Present working directory) To display the full path name of the present working directory.

eg: pwd

2. **mkdir** - (Make directory) To create a subdirectory in the current directory.

eg: mkdir stud

3. **cd** - (change directory) Change to another directory.

eg:1) cd stud

eg:2) cd .. - To change to the parent directory.

eg:3) cd ~ - To change to home directory.

4. **rmdir** - (Remove directory) To remove a directory.

eg: rmdir stud

Minimum Requirements for Linux :

Minimum Requirements for Fedora or RedHat Linux:

Processor	-	400 MHz Pentium for GUI Installation
RAM	-	1 GB (At least 2 or 3 GB is better)
DVD Drive	-	To boot up installation process
Network card	-	Wired or wireless; to add more software for software updates
Disk space	-	10 GB for installation
Special Hardware	-	AMD – V or Intel VT chip for using it as a virtualization host.

There are many linux distributors for handheld devices or old PC’s (such as Damn Small Linux, Slack ware) which require as little as 24 MB RAM and 486 processor.

Module II : Essential Linux Commands

Processes in Linux : A process is a running instance of a command. A process is identified in the system by a process id. Process id is unique for a system. Other attributes associated with the process are: - user account and group account.

File redirection metacharacters:

>	-	Used to create a file. It directs the standard output of a command to a file, deleting the existing file.
>>	-	Directs the output of the command to a file, adding the output to the end of the existing file.
<	-	Used to redirect the content from the file.

eg1: cat > sports
eg2: cat >> sports
eg3: sort < sports

Piping in Linux:

A pipe is a form of redirection. A pipe sends the output of one command/program/process to another command/program/process for further processing. The stdout of a command is redirected to stdin of another command. The pipe character is '|'.

Pipe is used to combine two or more commands. The output of one command acts as input to another command, and this command's output may act as input to the next command and so on.

The commands/ programs/ processes operate simultaneously. Data are transferred between them continuously. Pipes are unidirectional. Data flows from left to right through the pipeline. Pipes avoid the use of temporary text files.

eg: ls -l | more - Listing all files and directories and give it as input to more command.

We can do piping through several commands as follows:

command1 | command2 | command3

One advantage of pipes in Linux is that, unlike some other popular OS, there is no intermediate file involved.

V. Process Related commands

1. **ps** - Displays a list of running processes. It includes the running programs who are running them.

```
eg1:   ps
o/p            PID     TTY    TIME   COMMAND
                       1030   tty1   0.03   sh
                       1056   tty1   0.10   ps
eg2:   ps u    - shows the details of user also.
o/p:     USER          PID    %CPU %MEM    TTY  STAT   START TIME    COMMAND
          stu          1030   0.0    0.7     tty1 S+     14.50 0.00    bash
          stu          1056   0.0    0.7     tty1 R+     18.22 0.00    psu
eg3:   ps   ux                 - Displays all processes for the current user.
eg4:   ps   alus | less        -  To display all processes running for all users.
eg5:   ps  - eo   pid,  user,  command    -e lists every running processes and
                                          -o to specify column names of information
```

2. **kill** - The KILL command is usually used to terminate a process to kill the process you must be the owner or have a super user status.
eg: kill 1056
kill command internally sends a signal: SIGKILL - default signal to terminate a process
SIGOUT - To continue the stopped process
SIGSTOP- To pause the process
eg1: kill SIGKILL 1056
eg2: killall – kill process based on name
eg3: killall file1

3. **who** - Gives information about the user logged in.
eg1: who
o/p : casp tty 7 2019-01-14 14:21:0
eg2: who - u - Display information about idle time and pid.
o/p: casp tty 7 2019-01-14 14:21 00:29 2255

4. **find** - used to search your file system for files
eg1: find
eg2: find / etc - To search the files in the etc directory (/ etc - directory) .
eg3: find / etc - name passwd - used to find files by name
eg4: find / usr - size + 10 M - to find files > 10 MB in the usr directory
eg5: find / usr - usr - used to find files by user.
eg6: find / usr - perm - used to find files by permission.
eg7: find / usr - exec - used to find files and execute command.

5. **sort** - Sorts the specified file line by line in character order.
eg1: sort file1 - file1 is sorted and displayed
eg2: sort game > newgame - the game file is sorted and placed in newgame file.
eg3: sort file1 file2 > allfiles - sorts the contents of file1 and file2 and redirects to the file allfiles.
eg4: sort file * > allfiles - sorts the contents of file1, file2, etc to the file allfiles.
eg5: sort - r music - sorts the music file in reverse order.

6. **touch** - Touch command is used to change file time stamps. It update the access and modification times of each file to the current time.
– a option is used to change the access time
– m option is used to change the modification time
eg1: touch - a file1
eg2: touch - m file1

<u>VI. File Processing Commands</u>
1. wc - to count the number of characters, words and lines in the file.
eg1: wc good o/p 2 9 48 good
eg2: wc - l good - to count the number of lines in the file. o/p 2 good
eg3: wc - w good - to count the number of words in the file. o/p 9 good
eg4: wc - c good - to count the number of characters in the file. o/p 48 good

2. **cut** - it is used to cutting out vertically from each line of a file.
eg1: cut - c 5 good
eg2: cut - c 2 -5 good
eg3: cut - c -5 good
eg4: cut - d ‘ . ‘ - f1 s4c - cut the first field from the file s4c with delimiter dot(.)
eg5: cut - d ‘ . ‘ - f2 sports

3. **paste** - Merge the corresponding lines from two files.
eg: paste s4c good

VII. Managing Background and Foreground Processes

In Linux we can move active programs between the background and foreground. Thus we can select one program we want to deal at the moment. To stop a running command and put it in the background, press 'Ctrl + Z'. The command ' fg' is used to bring a process into foreground. The command 'jobs' is used to check which commands are running in the background. To place a program in the background, type an '&' at the end of the command line at the time of running the program.

```
eg1:    ls  - R &
        find  / usr > temp/file  &
        vi  &
eg2:    jobs
            [ 1 ]   Exit   ls  -  R &
            [ 2 ]   Exit   find  / usr > /temp/file  &
            [ 3 ]  - Stopped   vi   &
```

Exit means the command is completed, Stopped means the command is paused and running means the command is executing.

The numbers [1], [2] and [3] give the job number. The – sign next to the number shows that it was placed in the background just before the most recent background job. The + sign next to the number shows that it was most recently placed in the background.

The find command finds all files in the usr directory and redirects to the file named ' f ' in the tmp directory.

VIII. Mathematical Commands

1. **expr**: to evaluate the expression. The operators used ' | ', '& ',' < ',' <= ',' = ' ,' != ', ' > ', ' >= ', ' + ', ' – ', ' * ', ' / ', ' % '.
eg: expr 5 + 6 o/p : 11

2. **factor**: Used to print the prime factors of the given number.
eg: factor 100 o/p: 2 2 5 5

IX. vi editor

vi is a text editor in Linux. There are other editors namely geany, emacs, nano, pico joe, gedit , vim (vi-improved), etc. The vi editor is used to create and edit text or file. To open a file called demo, type the command;

```
vi demo
```

The bottom line of vi editor keeps the information about editing.

vi works in 3 modes:

(i) **input mode :**	It is used to enter text into the file.
(ii) **command mode** :	The vi editor opens in this mode. It is used to pass command that make action on the text. Press Esc key to return from input mode to command mode.
(iii) **Esc mode :**	It is used to save a file or switch to another file. Last Line mode is invoked by typing colon (:) while vi is in command mode. This mode is also called **last line mode** or **Ex mode**(Extended)

1. **Commands for Adding Text in vi** - To get input mode from command mode, type a input command letter [a, A, i, I, o, O]. After finishing the input of text, press the Esc key to return to command mode.

i -	To insert text at the left of the cursor
I -	To insert text at the beginning of the line.
a -	To add text at the right of the cursor.
A -	To add text at the end of the line.
o -	To open a line below the current line.
O -	To open a line above the current line.

2. **Commands for Moving around in the text** - Arrow keys moves the cursor left, right, up and down.

h	:	move left by one character
i	:	move right by one character
k	:	move up by one character
j	:	move down by one character
w	:	moves the cursor to the beginning of the next word
b	:	moves the cursor to the beginning of the previous word

3. **Commands for Deleting**

x	:	deletes the character under the cursor
X	:	deletes the character before the cursor
2x	:	delete 2 characters
D	:	delete from the cursor to the end of the line
dd	:	delete the current line
3dd	:	delete 3 lines
dw	:	delete the current word after the cursor position
db	:	delete the current word before the current cursor position
c$	:	changes the characters from the current character to the end of the current line and goes to the input mode.
C0	:	changes from the previous character to the beginning of character and goes to the input mode.
cc	:	erases the line and goes to the input mode.

4. **Commands for Copying and Pasting text**

yy	:	copies the current line into the buffer.
p	:	puts the copied text to the left of the cursor.
P	:	puts the buffered text to the right of the cursor.

5. **Repeating Commands**
After delete, change our page text you can repeat the action by typing period.

6. **Saving or Quitting Commands**

:w	-	saves the current file but does not exit from vi.
:wq	-	saves and quits
:q	-	quit
:q!	-	quits the current file and does not save the changes.
zz	-	saves the current changes to the file and exit.

<u>**7. commands for Skipping around in the file**</u>

ctrl + f	-	page forward, one page at a time.
ctrl +bf	-	page backward, one page at a time.
ctrl + d	-	page forward, half page at a time.

ctrl + u	-	page backward, half page at a time.
ctrl + g	-	goes to the last line of the page.
1g	-	goes to the first line of the page.

8. **Do and Undo commands**

u	-	undo the previous change
ctrl + R	-	to redo a command

9. **Searching for text**

/hello	-	searches forward the word hello
?good bye	-	searches backward for the word good bye

X. Process Scheduling Commands

1. at 2. batch 3. crontab 4. nohup (no hang up)

1. **at** - It schedules a command to run once at a later time. The command can be anything like a message or complex script.

Steps: - Run the 'at' command in the command line with a scheduled time.

- At the special prompt type the command.
- Press Ctrl + D

Syntax: at time date

```
eg:  at   11am   May 14
         at  >  echo  "Welcome"
         at  > ^ D
```

The at command puts jobs on 'queue a ' by default.

2. **batch** - The batch command is a variation of at command. It executes commands when the system load level drops below a specified value (1.5 or 0.8). If the system is not busy, the kernel gives control to such a program. The batch command will delay the start of the job until there is idle time. The batch command puts jobs on 'queue b' by default.

3. **crontab** - The crontab is a table containing a list of commands that are scheduled to run at regular time intervals on the computer. The crontab command opens the crontab for editing and let us add, remove or modify scheduled tasks. cron is a daemon which reads the crontab and executes the commands at the right time. Daemon is a background process that perform a specific function or task. corntab is a list of command that you want to run on a regular schedule.

Format: minute hour day month weekday [username] command

```
Eg:    30   .08  10  05     *  /home/user1/file1
        59     5  .21  04     wed  /home/user1/file1
```

crontab command examples

eg1: crontab	-e	to edit crontab
eg2: crontab	-l	to list contents
eg3: crontab	-r	to remove contents

4. **nohup** (no hang up) - Prefixing a command with nohup prevents the command from being aborted automatically when you logout or exit the shell.

```
eg:    nohup  abcd  -sh  $
```

****