BCa - 2nd year Operating Systems

UNIT-IV-V 3.

Laishali

Linux

Linux operating system is one of the popular versions of the UNIX operating system, which is designed to offer a free or low cost operating system for personal computer users. It gained the reputation as a fast performing and very efficient system.This is a remarkably complete operating system, including a GUI (graphical user interface), TCP/IP, the Emacs editor,can X Window System, etc.

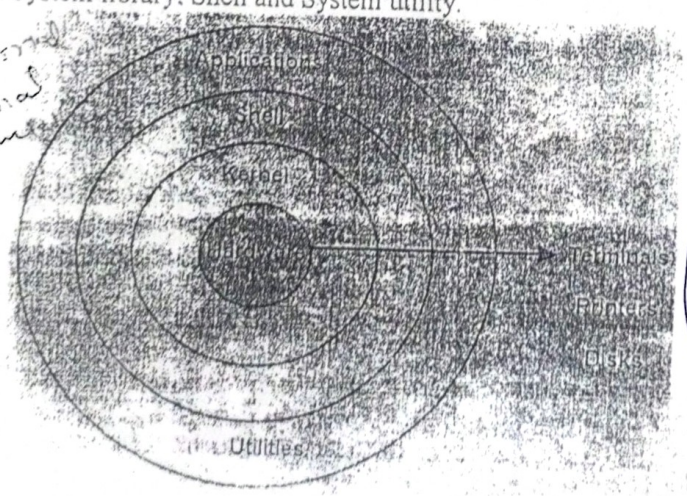
The History of LINUX Operating System

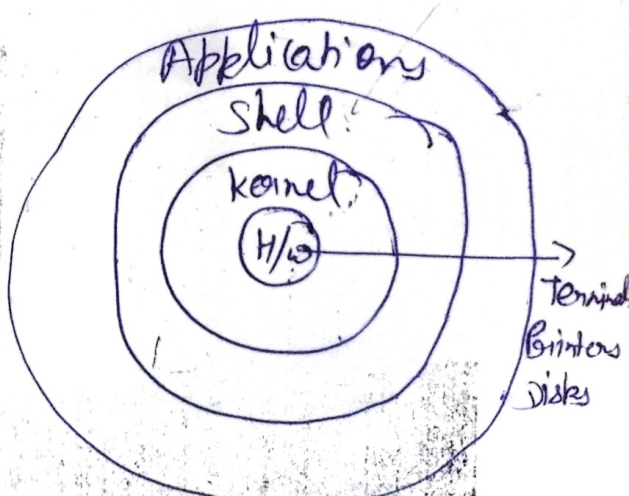
The History of Linux began in the 1991 with the beginning of a personal project by a Finland student Linus Torvalds to create a new free operating system kernel. Since then, the resulting Linux Kernel has been marked by constant growth throughout the history.

- In the year 1991, Linux was introduced by a Finland student Linus Torvalds.
- Hewlett Packard UniX(HP-UX) 8.0 was released.
- In the year 1992, Hewlett Packard 9.0 was released.
- In the year 1993, NetBSD 0.8 and FreeBSD 1.0 released.
- In the year 1994, Red Hat Linux was introduced, Caldera was founded by Bryan Sparks and Ransom Love and NetBSD1.0 Released.
- In the year 1995, FreeBSD 2.0 and HP UX 10.0 was released.
- In the year 1996, K Desktop Environment was developed by Matthias Ettrich.
- In the year 1997, HP-UX 11.0 was released.
- In the year 1998, the fifth generation of SGI Unix i.e IRIX 6.5 , Sun Solaris 7 operating system and Free BSD 3.0 was released.
- In the year 2000, the agreement of Caldera Systems with SCO server software division and the professional services division was announced.
- In the year 2001, Linus Torvalds released the Linux 2.4 version source code.
- In the year 2001, Microsoft filed a trademark suit against Lindows.com
- In the year 2004, Lindows name was changed to Linspire.
- In the year 2004, the first release of Ubuntu was released.
- In the year 2005, The project, openSUSE began a free distribution from Novell's community.
- In the year 2006, Oracle released its own distribution of Red Hat.
- In the year 2007, Dell started distributing laptops with Ubuntu pre installed in it.
- In the year 2011, Linux kernel 3.0 version was released.
- In the year 2013, Googles Linux based Android claimed 75% of the smartphone market share, in terms of the number of phones shipped.
- In the year 2014, Ubuntu claimed 22,000,000 users.

Linux System Architecture

The Linux Operating System's architecture primarily has these components: the Kernel, Hardware layer, System library, Shell and System utility.





The kernel is the core part of the operating system, which is responsible for all the major activities of the LINUX operating system. This operating system consists of different modules and interacts directly with the underlying hardware. The kernel offers the required abstraction to hide application programs or low-level hardware details to the system. The types of Kernels are as follows:

- Monolithic Kernel
- Micro kernels
- Exo kernels
- Hybrid kernels

System libraries are special functions, that are used to implement the functionality of the operating system and do not require code access rights of kernel modules.
System Utility programs are liable to do individual, and specialized-level tasks.
Hardware layer of the LINUX operating system consists of peripheral devices such as RAM, HDD, CPU.
The shell is an interface between the user and the kernel, and it affords services of the kernel. It takes commands from the user and executes kernel's functions. The Shell is present in different types of operating systems, which are classified into two types: command line shells and graphical shells.
The command line shells provide a command line interface, while the graphical line shells provide a graphical user interface. Though both shells perform operations, but the graphical user interface shells perform slower than the command line interface shells. Types of shells are classified into four:

- Korn shell
- Bourne shell
- C shell
- POSIX shell

Features of Linux Operating System

The main features of Linux operating system are
Portable: Linux operating system can work on different types of hardwares as well as Linux kernel supports the installation of any kind of hardware platform.
Open Source: Source code of LINUX operating system is freely available and, to enhance the ability of the LINUX operating system, many teams work in collaboration.
Multiuser: Linux operating system is a multiuser system, which means, multiple users can access the system resources like RAM, Memory or Application programs at the same time.
Multiprogramming: Linux operating system is a multiprogramming system, which means multiple applications can run at the same time.
Hierarchical File System: Linux operating system affords a standard file structure in which system files or user files are arranged.
Shell: Linux operating system offers a special interpreter program, that can be used to execute commands of the OS. It can be used to do several types of operations like call application programs. and so on.
Security: Linux operating system offers user security systems using authentication features like encryption of data or password protection or controlled access to particular files.

Steps of Installation

Linux is the foundation of thousands of open source operating systems designed to replace Windows and Mac OS. It is free to download and install on any computer. Because it is open source, there are a variety of different versions, or distributions, available developed by different groups. Follow this guide for basic instructions on how to install any version of Linux.

1. **Download the Linux distribution of your choice.** If you're new to Linux, consider trying a lightweight and easy to use distribution, such as Ubuntu or Linux Mint. Linux distributions (known as "distros") are typically available for free to download in ISO format. You can find the ISO for the distribution of your choice at the distribution's

website. This format needs to be burned to a CD or USB stick before you can use it to install Linux. This will create a Live CD or Live USB.

2. **Boot into the Live CD or Live USB.** Most computers are set to boot into the hard drive first, which means you will need to change some settings to boot from your newly-burned CD or USB. Start by rebooting the computer.
   - Once the computer reboots, press the key used to enter the boot menu. The key for your system will be displayed on the same screen as the manufacturer's logo. Typical keys include F12, F2, or Del.
   - Once you're in the boot menu, select your live CD or USB. Once you've changed the settings, save and exit the BIOS setup or boot menu. Your computer will continue with the boot process.
3. **Try out the Linux distribution before installing.** Most Live CDs and USBs can launch a "live environment", giving you the ability to test it out before making the switch. You won't be able to create files, but you can navigate around the interface and decide if it's right for you.
4. **Start the installation process.** If you're trying out the distro, you can launch the installation from the application on the desktop. If you decided not to try out the distribution, you can start the installation from the boot menu. You will be asked to configure some basic options, such as language, keyboard layout, and time zone.
5. **Create a username and password.** You will need to create login information to install Linux. A password will be required to log into your account and preform administrative tasks.
6. **Set up the partition.** Linux needs to be installed on a separate partition from any other operating systems on your computer if you intend dual booting Linux with another OS. A partition is a portion of the hard drive that is formatted specifically for that operating system. You can skip this step if you don't plan on dual booting.
   - Distros such as Ubuntu will set a recommended partition automatically. You can then adjust this manually yourself. Most Linux installations require at least 20 GB, so be sure to set aside enough room for both the Linux operating system and any other programs you may install and files you may create.
   - If the installation process does not give you automatic partitions, make sure that the partition you create is formatted as Ext4. If the copy of Linux you are installing is the only operating system on the computer, you will most likely have to manually set your partition size.
7. **Boot into Linux.** Once the installation is finished, your computer will reboot. You will see a new screen when your computer boots up called "GNU GRUB". This is a boot loader that handles Linux installations. Pick your new Linux distro from the list. This screen may not show up if you only have one operating system on your computer. If this screen isn't being presented to you automatically, then you can get it back by hitting shift right after the manufacturer splash screen. If you install multiple distros on your computer, they will all be listed here.
8. **Check your hardware.** Most hardware should work out of the box with your Linux distro, though you may need to download some additional drivers to get everything working.
   - Some hardware requires proprietary drivers to work correctly in Linux. This is most common with graphics cards. There is typically an open source driver that will work, but to get the most out of your graphics cards you will need to download the proprietary drivers from the manufacturer.
   - In Ubuntu, you can download proprietary drivers through the System Settings menu. Select the Additional Drivers option, and then select the graphics driver from the list. Other distros have specific methods for obtaining extra drivers.
   - You can find other drivers from this list as well, such as Wi-Fi drivers.
9. **Start using Linux.** Once your installation is complete and you've verified that your hardware is working, you're ready to start using Linux. Most distros come with

several popular programs installed, and you can download many more from their respective file repositories.

Shell and Kernel

Both the Shell and the Kernel are the Parts of Linux Operating System. These Both Parts are used for performing any Operation on the System. When a user gives his Command for Performing Any Operation, then the Request will goes to the Shell Parts, The Shell Parts is also called as the Interpreter which translates the Human Program into the Machine Language and then the Request will be transferred to the Kernel. So that Shell is just called as the interpreter of the Commands which converts the Request of the user into the Machine Language.

1. Kernel is also called as the **heart of the Operating System** and the Every **Operation is performed by using the Kernel**, When the Kernel Receives the Request from the Shell then this will Process the Request and Display the Results on the Screen. The various Types of Operations those are Performed by the Kernel are as followings:-
2. It Controls the State the Process Means it checks whether the **Process is running or Process is waiting** for the Request of the user.
3. **Provides the Memory for the Processes those are Running on the System** Means Kernel Runs the Allocation and De-allocation Process , First When we Request for the service then the Kernel will Provides the Memory to the Process and after that he also Release the Memory which is Given to a Process.
4. The Kernel also Maintains a **Time table for all the Processes** those are Running Means the Kernel also **Prepare the Schedule Time means this will Provide the Time to various Process of the CPU** and the Kernel also Puts the Waiting and Suspended Jobs into the different Memory Area.
5. When a **Kernel determines that the Logical Memory doesn't fit to Store the Programs. Then he uses the Concept of the Physical Memory which Will Stores the Programs into Temporary Manner. Means the Physical Memory of the System can be used as Tempora y Memory.**
6. **Kernel also maintains all the files those are Stored into the Computer System and the Kernel Also Stores all the Files into the System as no one can read or Write the Files without any Permissions. So that the Kernel System also Provides** us the Facility to use the Passwords and also all the Files are Stored into the Particular Manner.

Shells

There are several different shells available, each with pros and cons. While tcsh (descended from C-shell/csh) and Bash-shell (bash) are the most common shells, the choice of shell is entirely up to user preference and availability on the system. To determine which shell you are currently using, type the *echo* command followed by the system environment variable *$SHELL*.

```
$ echo $SHELL
/bin/bash
```

Here, *echo* is the command entered through the shell, and *$SHELL* is a command argument

Job Control in the shell

In addition to starting commands, the shell provides basic job control functions for processes. For [illegible] options, it can be used to see and control processes [illegible]. Job control allows the user to stop, suspend, and resume jobs from within the shell. This is useful if you have a program which runs over a long period of time or which does not complete due to a bug or other problems. From within a shell session, the *ps* command will show the currently running processes in your shell session.

```
$ ps
 PID TTY          TIME CMD
4621 pts/7    00:00:00 bash
32273 pts/7   00:00:00 ps
```

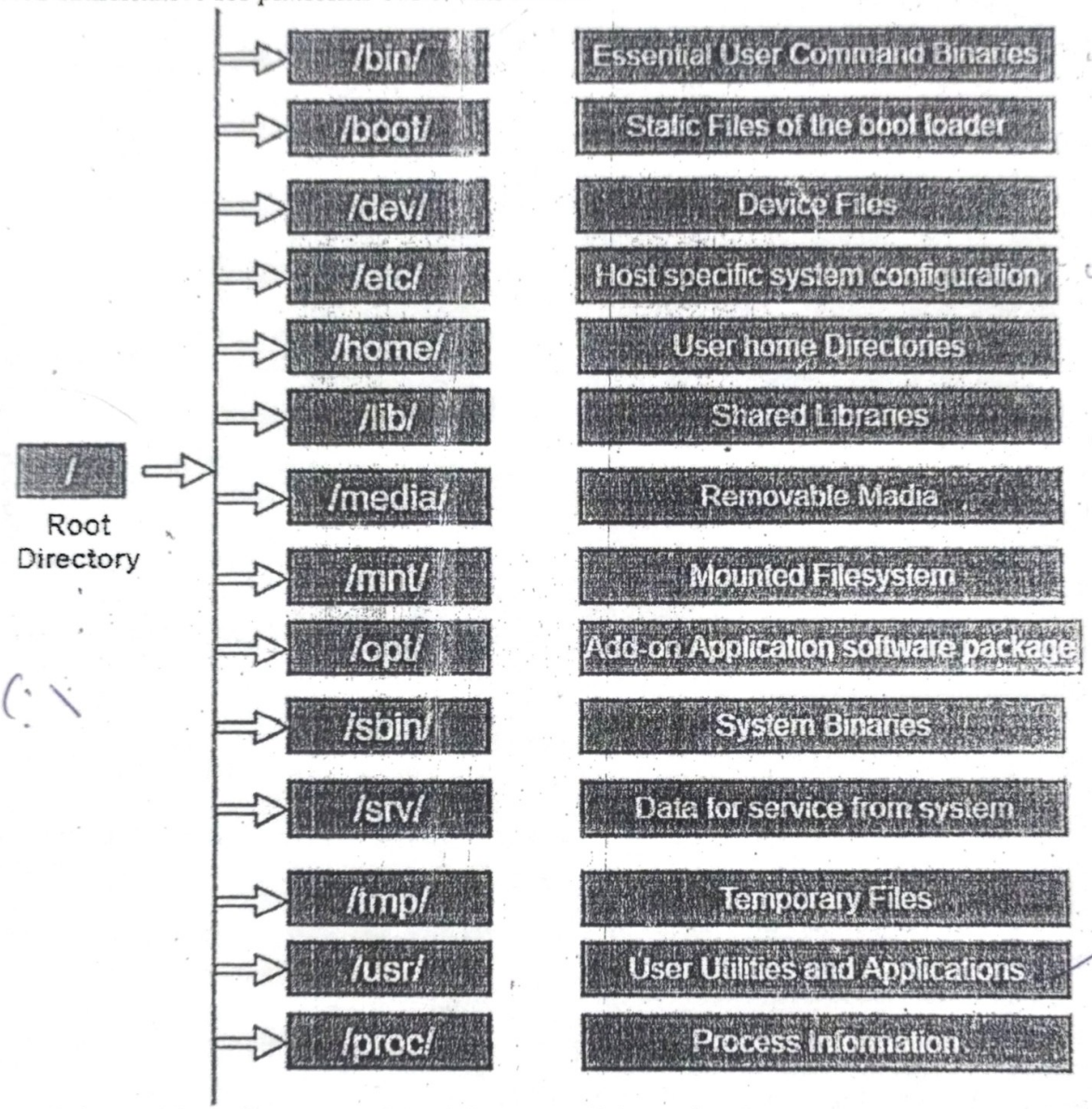
Directory Structure (Linux File Hierarchy Structure)

The Linux File Hierarchy Structure or the File system Hierarchy Standard (FHS) defines the directory structure and directory contents in Unix-like operating systems. It is maintained by the Linux Foundation.

In the FHS, all files and directories appear under the root directory /, even if they are stored on different physical or virtual devices.

Some of these directories only exist on a particular system if certain subsystems, such as the X Window System, are installed.

Most of these directories exist in all UNIX operating systems and are generally used in much the same way; however, the descriptions here are those used specifically for the FHS, and are not considered authoritative for platforms other than Linux.



1. / (Root): Primary hierarchy root and root directory of the entire file system hierarchy.
Every single file and directory starts from the root directory
Only root user has the right to write under this directory
/root is root user's home directory, which is not same as /
2. /bin: Essential command binaries that need to be available in single user mode; for all users, e.g., cat, ls, cp.
Contains binary executables
Common linux commands you need to use in single-user modes are located under this directory.
Commands used by all the users of the system are located here e.g. ps, ls, ping, grep, cp
3. /boot: Boot loader files, e.g., kernels, initrd.
Kernel initrd, vmlinux, grub files are located under /boot
Example: initrd.img-2.6.32-24-generic, vmlinuz-2.6.32-24-generic

4. /dev: Essential device files, e.g., /dev/null.
These include terminal devices, usb, or any device attached to the system.
Example: /dev/tty1, /dev/usbmon0
5. /etc: Host-specific system-wide configuration files.
Contains configuration files required by all programs.
This also contains startup and shutdown shell scripts used to start/stop individual programs.
Example: /etc/resolv.conf, /etc/logrotate.conf.
6. /home: Users' home directories, containing saved files, personal settings, etc.
Home directories for all users to store their personal files.
Example: /home/kishlay, /home/kv
7. /lib: Libraries essential for the binaries in /bin/ and /sbin/.
Library filenames are either ld* or lib*.so.* loader, library, software
Example: ld-2.11.1.so, libncurses.so.5.7
8. /media: Mount points for removable media such as CD-ROMs (appeared in FHS-2.3).
Temporary mount directory for removable devices.
Examples, /media/cdrom for CD-ROM; /media floppy for floppy drives; /media/cdrecorder for CD writer
9. /mnt: Temporarily mounted filesystems.
Temporary mount directory where sysadmins can mount filesystems.
10. /opt: Optional application software packages.
Contains add-on applications from individual vendors.
Add-on applications should be installed under either /opt/ or /opt/ sub-directory.
11. /sbin: Essential system binaries, e.g., fsck, init, route.
Just like /bin, /sbin also contains binary executables.
The Linux commands located under this directory are used typically by system aministrator, for system maintenance purpose.
Example: iptables, reboot, fdisk, ifconfig, swapon
12. /srv: Site-specific data served by this system, such as data and scripts for web servers, data offered by FTP servers, and repositories for version control systems.
srv stands for service.
Contains server specific services related data.
Example, /srv/cvs contains CVS related data.
13. /tmp: Temporary files. Often not preserved between system reboots, and may be severely size restricted.
Directory that contains temporary files created by system and users.
Files under this directory are deleted when system is rebooted.
14. /usr: Secondary hierarchy for read-only user data; contains the majority of (multi-)user utilities and applications.
Contains binaries, libraries, documentation, and source-code for second level programs.
/usr/bin contains binary files for user programs. If you can't find a user binary under /bin, look under /usr/bin. For example: at, awk, cc, less, scp
/usr/sbin contains binary files for system administrators. If you can't find a system binary under /sbin, look under /usr/sbin. For example: atd, cron, sshd, useradd, userdel
/usr/lib contains libraries for /usr/bin and /usr/sbin
/usr/local contains users programs that you install from source. For example, when you install apache from source, it goes under /usr/local/apache2
/usr/src holds the Linux kernel sources, header-files and documentation.
15. /proc: Virtual file system providing process and kernel information as files. In Linux, corresponds to a procfs mount. Generally automatically generated and populated by the system, on the fly.
Contains information about system process.
This is a pseudo file system contains information about running process. For example: /proc/{pid} directory contains information about the process with that particular pid.
This is a virtual filesystem with text information about system resources. For example: /proc/uptime

Linux Commands

Miscellaneous Commands

Manual command
man
This is help command, and will explains you about online manual pages you can also use man in conjunction with any command to learn more about that command for example.

- man ls will explain about the ls command and how you can use it.
- man -k pattern command will search for the pattern in given command.

Banner command
banner prints characters in a sort of ascii art poster, for example to print wait in big letters. typing banner wait at UNIX command line will look as follows.

```
=  #  ##    #   #####
=  # # #    #     #
=  # #  #   #     #
= ## # ######   #     #
## ## #  #    #     #
=  # #  #    #     #
```

Cal command
cal command will print the calendar on current month by default. If you want to print calendar of august of 1965. That's eighth month of 1965.
cal 8 1965 will print following results.

```
    August 1965
S  M Tu  W Th  F  S
    2  3  4  5  6  7
8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

Clear command
clear command clears the screen and puts cursor at beginning of first line.

Tty command
Tty command will display your terminal. Syntax is
tty options
Options

- -l will print the synchronous line number.
- -s will return only the codes: 0 (a terminal), 1 (not a terminal), 2 (invalid options) (good for scripts)

File Management commands

Pwd command
pwd command will print your home directory on screen, pwd means print working directory.
/u0/ssb/sandeep
is output for the command when pwd is used in /u0/ssb/sandeep directory.

ls command
ls command is most widely used command and it displays the contents of directory.
Options
- ls will list all the files in your home directory, this command has many options.
- ls -l will list all the file names, permissions, group, etc in long format.
- ls -a will list all the files including hidden files that start with . .
- ls -lt will list all files names based on the time of creation, newer files bring first.
- ls –Fx will list files and directory names will be followed by slash.
- ls –R will lists all the files and files in the all the directories, recursively.
- ls -R | more will list all the files and files in all the directories. one page at a time.

– pipe

mkdir command
mkdir sandeep will create new directory, i.e. here **sandeep** directory is created.

cd command
cd sandeep will change directory from current directory to **sandeep** directory. Use pwd to check your current directory and **ls** to see if **sandeep** directory is there or not. You can then use **cd sandeep** to change the directory to this new directory.

cat command
cat cal.txt
cat command displays the contents of a file here cal.txt on screen (or standard out).

head command
head filename by default will display the first 10 lines of a file. If you want first 50 lines you can use head -50 filename or for 37 lines head -37 filename and so forth.

tail command
tail filename by default will display the last 10 lines of a file. If you want last 50 lines then you can use **tail -50 filename**.

more command
more command will display a page at a time and then wait for input which is spacebar. For example if you have a file which is 500 lines and you want to read it all. So you can use
more filename

wc command
wc command counts the characters, words or lines in a file depending upon the option.
Options
wc -l filename will print total number of lines in a file.
wc -w filename will print total number of words in a file.
wc -c filename will print total number of characters in a file.

cp command
cp command copies a file. If you want to copy a file named oldfile in a current directory to a file named newfile in a current directory.
cp oldfile newfile
If you want to copy oldfile to other directory for example **/tmp** then **cp oldfile /tmp/newfile**. Useful options available with **cp** are **-p** and **-r** , **-p** options preserves the modification time and permissions, **-r** recursively copy a directory and its files, duplicating the tree structure.

mv command
mv command is used to move a file from one directory to another directory or to rename a file.
Some examples:
mv oldfile newfile will rename oldfile to newfile.
mv -i oldfile newfile for confirmation prompt.
mv -f oldfile newfile will force the rename even if target file exists.
mv * /usr/bajwa/ will move all the files in current directory to /usr/bajwa directory.

rm command
To delete files use rm command.
Options:

- rm oldfile will delete file named oldfile.
- rm -f option will remove write-protected files without prompting.
- rm -r option will delete the entire directory as well as all the subdirectories, it is very dangerous command.

rmdir command
rmdir command will remove directory or directories if a directory is empty.
Options:

- rm -r directory_name will remove all files even if directory is not empty.
- rmdir sandeep use it to remove sandeep directory.
- rmdir -p will remove directories and any parent directories that are empty.
- rmdir -s will suppress standard error messages caused by -p.

Comparison and Searching Commands
diff command
diff command will compare the two files and print out the differences between. in this example, there are two ascii text files. File one and file two.

Contents of file one are
This is first file
this is second line
this is third line
this is different as;lkdjf
this is not different

file two contains
This is first file
this is second line
this is third line
this is different xxxxxxxas;lkdjf
this is not different

diff fileone filetwo will give following output
4c4
< this is different as;lkdjf

> this is different xxxxxxxas;lkdjf

cmp command
cmp command compares the two files. For example, there are two different files fileone and filetwo.
cmp fileone filetwo will give
fileone filetwo differ: char 80, line 4

If cmp command is run on similar files nothing is returned.
-s command can be used to return exit codes. i.e. return 0 if files are identical, 1 if files are different. 2 if files are inaccessible.
This following command prints a message 'no changes' if files are same
cmp -s fileone file1 && echo 'no changes' .
no changes

grep Command global regular expression
grep command is the most useful search command. You can use it to find processes running on system, to find a pattern in a file, etc. It can be used to search one or more files to match an expression.
It can also be used in conjunction with other commands as in this following example, output of ps command is passed to grep command, here it means search all processes in system and find the pattern sleep.
e.g. -ef | grep sleep will display all the sleep processes running in the system as follows.

```
ops 12964 25853  0 16:12:24 ttyAE/AAES  0:00 sleep 60
dxi 12974 15640  0 16:12:25 ttyAH/AAHP  0:00 sleep 60
ops 12941 25688  0 16:12:21 ttyAE/AAEt  0:00 sleep 60
ops 12847 25812  0 16:11:59 ttyAH/AAH6  0:00 sleep 60
ops 12894 25834  0 16:12:12 ttyAE/AAEX  0:00 sleep 60
dxi 13067 27253  2 16:12:48 ttyAE/ABEY  0:00 sleep 1
ops 13046 25761  0 16:12:44 ttyAE/AAE0  0:00 sleep 60
dxi 12956 13078  0 16:12:23 ttyAG/AAG+  0:00 sleep 60
ops 12965 25737  0 16:12:24 ttyAE/AAEp  0:00 sleep 60
ops 12989 25778  0 16:12:28 ttyAH/AAHv  0:00 sleep 60
ssb 13069 26758  2 16:12:49 ttyAH/AAHs  0:00 grep sleep
pjk 27049  3353  0 15:20:23 ?          0:00 sleep 3600
```

Creating a file
$ cat > Mamta
saving a file
ctrl + D
:w ← Save changes after editing
:q ← quit

Options:
-b option will precede each line with its block number.
-c option will only print the count of matched lines.
-i ignores uppercase and lowercase distinctions.
-l lists filenames but not matched lines.
Other associated commands with grep are **egrep** and **fgrep**. **egrep** typically runs faster.

VI editor
The VI stands for Visual editor: another text editor in Linux. This is a standard editor in many Linux/Unix environments. This is the default editor that comes with many Linux distributions. It might be possible that it is the only text editor available with your distro.
You can open a file with vi for editing using the following:
$ vi hello.txt
The vi editor has 3 modes in which it performs its functions. The default is COMMAND mode, in which tasks like copy, paste, undo etc can be performed. You can change a mode from command mode only (and come back to it). The second mode is the INSERT mode, in which whatever key you type is treated as a character and will be loaded into the file buffer. To enter this mode, press 'i' when in command mode.
The final mode is EX mode or last line mode. The changes made in the buffer can be saved or discarded in this mode.

```
Hello world

.
This file is edited using vi editor.
~
~
~
```

"hello.txt" 2 lines, 50 characters

Some additional useful commands

alias command

The 'alias' is another name for a command. If no argument is given, it shows current aliases. Aliases can be used for short names of commands. For example, you might use the clear command frequently. You can create an alias for it:

$ alias c="clear"

Next time you enter 'c ' on command line, your screen will get clear. Current aliases can be checked with 'alias' command:

```
$ alias
alias alert='notify-send --urgency=low -i "$([ $? = 0 ] && echo terminal || echo error)"
"$(history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[;&|]\s*alert$//'\'')"'
alias c='clear'
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alF'
alias ls='ls --color=auto'
```

w command

'w' command is used to check which users are logged in to the system, and what command they are executing at that particular time:

```
$ w
10:06:56 up 57 min, 3 users, load average: 0.04, 0.06, 0.09
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
root tty1 10:06 28.00s 1.02s 0.67s pager -s
raghu tty7 :0 09:19 57:33 1:22 0.20s gnome-session --session=classic-gnome
raghu pts/0 :0.0 09:34 0.00s 0.78s 0.00s w
```

It also shows the uptime, number of users logged in and load average of the system (in the first line of output above).

last command

Displays information about the users who logged in and out of the system. The output of the last command can be very large, so the following output has been filtered (through head) to display the top 10 lines only:

```
$ last | head
root tty1 Mon Jul 9 10:06 still logged in
root tty1 Mon Jul 9 10:06 - 10:06 (00:00)
raghu pts/1 :0.0 Mon Jul 9 10:05 - 10:06 (00:00)
raghu pts/0 :0.0 Mon Jul 9 09:34 still logged in
raghu tty7 :0 Mon Jul 9 09:19 still logged in
reboot system boot 2.6.38-13-generi Mon Jul 9 09:09 - 10:12 (01:02)
raghu tty7 :0 Sun Jul 8 23:36 - 00:30 (00:54)
reboot system boot 2.6.38-13-generi Sun Jul 8 23:36 - 00:30 (00:54)
raghu tty7 :0 Sun Jul 8 21:07 - down (01:06)
reboot system boot 2.6.38-13-generi Sun Jul 8 21:07 - 22:14 (01:07)
```

A similar command is 'lastb' that shows the last unsuccessful login attempts. But this command must be run as root otherwise you would get an error saying permission denied.

```
$ lastb
raghu tty2 Mon Jul 9 10:16 - 10:16 (00:00)
```

UNKNOWN tty2 Mon Jul 9 10:15 - 10:15 (00:00)
ubuntu tty8 :1 Mon Jul 2 10:23 - 10:23 (00:00)
btmp begins Mon Jul 2 10:23:54 2012

du command
The du command determines disk usage of a file. If the argument given to it is a directory, then it will list disk usage of all the files and directories recursively under that directory:
$ du /etc/passwd
4 /etc/passwd
$ du hello/
52 hello/HelloApp
4 hello/orb.db/logs
20 hello/orb.db
108 hello/

df command
The df reports file system usage. For example:
$ df
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/sda7 10079084 7372872 2194212 78% /
none 1522384 768 1521616 1% /dev
none 1529012 252 1528760 1% /dev/shm
none 1529012 108 1528904 1% /var/run
none 1529012 4 1529008 1% /var/lock
/dev/sda8 5039616 3758824 1024792 79% /home
/dev/sda2 209715196 196519248 13195948 94% /media/Data

fdisk command
The fdisk is a tool for getting partition information, and for adding and removing partitions. The fdisk tool requires super user privileges. To list all the partitions of all the hard drives available:
$ fdisk -l
Disk /dev/sda: 320.1 GB, 320072933376 bytes
255 heads, 63 sectors/track, 38913 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x396f396f
Device Boot Start End Blocks Id System
/dev/sda1 1 2611 20971520 7 HPFS/NTFS
/dev/sda2 2611 28720 209715200 7 HPFS/NTFS
/dev/sda3 * 28720 38914 81882113 5 Extended
/dev/sda5 28720 33942 41943040 7 HPFS/NTFS
/dev/sda6 33942 34464 4194304 7 HPFS/NTFS
/dev/sda7 34464 35739 10240000 83 Linux
/dev/sda8 35739 36376 5120000 83 Linux
/dev/sda9 36376 36886 4096000 82 Linux swap / Solaris
/dev/sda10 36887 38276 11164672 83 Linux
/dev/sda11 38277 38914 5117952 83 Linux
The fdisk is an interactive tool to edit the partition table. It takes a device (hard disk) as an argument, whose partition table needs to be edited.
$ fdisk /dev/sda
WARNING: DOS-compatible mode is deprecated. It's strongly recommended to switch off the mode (command 'c') and change display units to sectors (command 'u').

Command (m for help): m

Command action
a toggle a bootable flag
b edit bsd disklabel
c toggle the dos compatibility flag
d delete a partition
l list known partition types
m print this menu
n add a new partition
o create a new empty DOS partition table
p print the partition table
q quit without saving changes
s create a new empty Sun disklabel
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)

Pressing 'm' at the **fdisk** prompt prints out the help shown above that lists all the commands available for **fdisk**. A new partition can be created with **'n'** and an existing partition can be deleted with the **'d'** command. When you are done editing the partitions, press **'w'** to write the changes to the disk, and finally, hit **'q'** to quit from **fdisk** (q does not save changes).

netstat command

The 'netstat' is a command used to check the network statistics of the system. It will list the current network connections, routing table information, interface statistics, masquerade connections and a lot more information.

```
$ netstat | head
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags Type State I-Node Path
unix 13 [ ] DGRAM 8498 /dev/log
unix 2 [ ] DGRAM 6824 @/org/kernel/udev/udevd
unix 3 [ ] STREAM CONNECTED 56738 /var/run/dbus/system_bus_socket
unix 3 [ ] STREAM CONNECTED 56113
unix 3 [ ] STREAM CONNECTED 29138
unix 3 [ ] STREAM CONNECTED 29137
```

history command

This command shows the commands you have entered on your terminal so far.

Loops in Shell Script

A loop is a powerful programming tool that enables you to execute a set of commands repeatedly. The following types of loops available to shell programmers –

- The while loop
- The for loop
- The until loop

For loop

The for loop is the first of the three shell looping constructs. This loop allows for specification of a list of values. A list of commands is executed for each value in the list.

The syntax for this loop is:

```
for NAME [in LIST ];
do
COMMANDS;
done
```

The return status is the exit status of the last command that executes. If no commands are executed because LIST does not expand to any items, the return status is zero.

NAME can be any variable name, although i is used very often. LIST can be any list of words, strings or numbers, which can be literal or generated by any command. The COMMANDS to execute can also be any operating system commands, script, program or shell statement. The first time through the loop, NAME is set to the first item in LIST. The second time, its value is set to the second item in the list, and so on. The loop terminates when NAME has taken on each of the values from LIST and no items are left in LIST.

Example #1: Write a Shell Script to print from 1 to 5 using for loop

In the following example we will use for loop to print from 1 to 5.

```
for i in 1 2 3 4 5
do
 echo Si
done
```

Brace expansion

We use the brace expansion {m..n} to generate string in shell script.

Example:

{1..5} will give 1 2 3 4 5

{a..f} will give a b c d e f

{Z..T} will give Z Y X W V U T

{-5..5} will give -5 -4 -3 -2 -1 0 1 2 3 4 5

{A,B,C,D} will give A B C D

{A,B,C{1..3},D} will give A B C1 C2 C3 D

Example #2: Write a Shell Script to print from 1 to 10 using brace expansion and for loop

```
for i in {1..10}
do
  echo Si
done
```

Where, {1..10} will expand to 1 2 3 4 5 6 7 8 9 10.

Example #3: Write a Shell Script to print from A to Z using for loop

```
for ch in {A..Z}
do
  echo $ch
done
```

Example #4: Write a Shell Script to list all the files in the current directory
For this example we will use the * which is a special character and it helps to list all the files in the current directory.

```
for f in *
do
  echo Sf
done
```

while Loop
The while loop enables you to execute a set of commands repeatedly until some condition occurs. It is usually used when you need to manipulate the value of a variable repeatedly.

Syntax

```
while command
do
   Statement(s) to be executed if command is true
done
```

Here the Shell command is evaluated. If the resulting value is true, given statement(s) are executed. If command is false then no statement will be executed and the program will jump to the next line after the done statement.

Example
The following example uses the while loop to display the numbers zero to nine −

```
a=0

while [ $a -lt 10 ]
do
   echo $a
   a=`expr $a + 1`
done
```

Upon execution, you will receive the following result −

```
0
1
2
3
4
5
6
```

7
8
9

Each time this loop executes, the variable a is checked to see whether it has a value that is less than 10. If the value of a is less than 10, this test condition has an exit status of 0. In this case, the current value of a is displayed and later a is incremented by 1.

Until Loop

The while loop is perfect for a situation where you need to execute a set of commands while some condition is true. Sometimes you need to execute a set of commands until a condition is true.

Syntax

```
until command
do
   Statement(s) to be executed until command is true
done
```

Here the Shell command is evaluated. If the resulting value is **false**, given statement(s) are executed. **If the command is true then no statement will be executed** and the program jumps to the next line after the done statement.

Example

The following example uses the until loop to display the numbers zero to nine −

```
a=0

until [ ! $a -lt 10 ]
do
   echo $a
   a=`expr $a + 1`
done
```

Upon execution, you will receive the following result −

```
0
1
2
3
4
5
6
7
8
9
```

Installing Packages

All software on a Linux system is divided into packages that can be installed, uninstalled, upgraded, queried, and verified. CentOS/RHEL uses the Red Hat Package Manager (RPM) to facilitate the installation, upgrade and removal of software packages.
The rpm utility provides many useful options for querying and verifying packages, as well as installing, upgrading, and removing packages. The following provides examples of these options.

Query Packages

1. **Listing all installed packages**

To list all installed packages, use the following command:

```
# rpm -qa | more
NetworkManager-team-1.8.0-9.el7.x86_64
pyxattr-0.5.1-5.el7.x86_64
HPOvXpl-11.14.014-1.x86_64
bind-utils-9.9.4-51.el7.x86_64
pyOpenSSL-0.13.1-3.el7.x86_64
....
```

The format of rpm package names is name-version-release.architecture. The example shows packages for version 7 of EnterPrise Linux (el7) with architectures of either:

- **x86_64: Any AMD64 or Intel 64 CPUs**
- **noarch: Any CPU architecture**
- **i686: 32-bit OS**

2. **Display Package Information**

To display detailed package information (of the bash package, for example), enter:

```
# rpm -ql bash
/etc/skel/.bash_logout
/etc/skel/.bash_profile
/etc/skel/.bashrc
/usr/bin/alias
/usr/bin/bash
/usr/bin/bashbug
/usr/bin/bashbug-64
/usr/bin/bg
```

3. **Find the package from a file**

To perform a reverse search, that is to determine what package a specific file (/etc/hosts, for example) belongs to, enter:

```
# rpm -qf /etc/hosts
setup-2.8.71-7.el7.noarch
```

4. **Find configuration files of a package**

To list configuration files associated with a package (the bash package, for example), enter:

```
# rpm -qc bash
/etc/skel/.bash_logout
/etc/skel/.bash_profile
/etc/skel/.bashrc
```

Installing and Updating Packages

1. **Installing or Upgrading packages**

Using the **rpm -U package_name** command upgrades installed packages, as well as installs new packages. For example, to install or upgrade the rsync package:

rpm -Uvh rsync-3.0.9-18.el7.x86_64.rpm

The -v (verbose) option displays more information

The -h (hash) option displays progress.

2. **Installing a New Kernel**

When installing a new kernel, use the **–i** option so as not to upgrade the current kernel, for example:

 # rpm –ivh kernel-3.10.0-229.el7.x86_64.rpm

Removing Packages

To remove a package (the **rsync** package, for example), enter:

 # rpm –e rsync