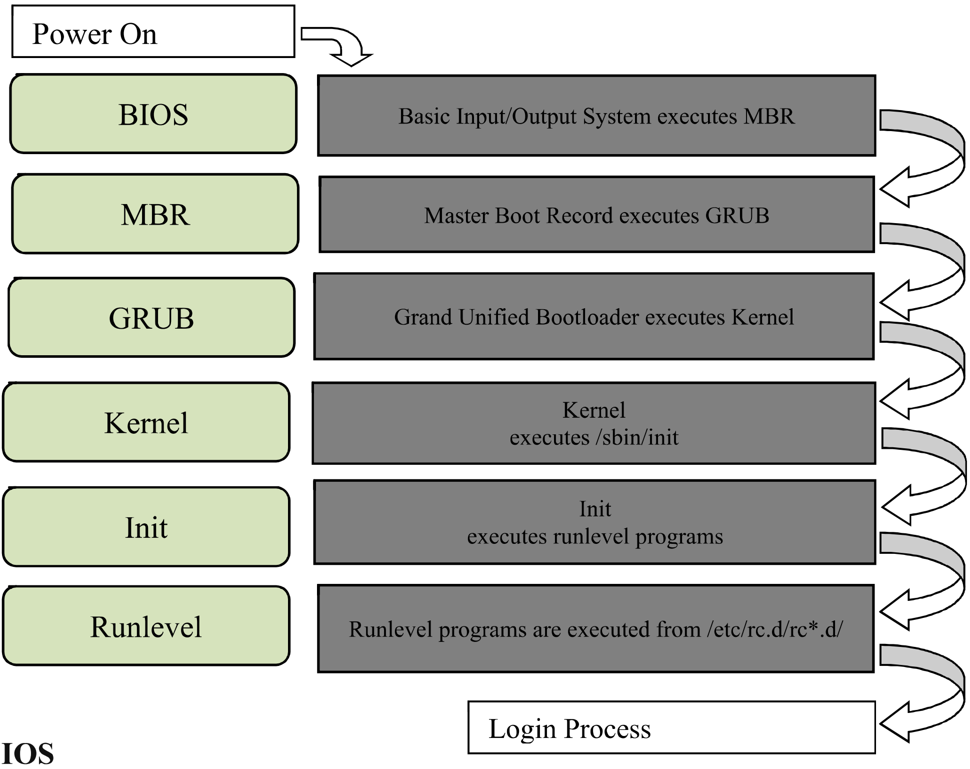
Ex. No. 1b	BOOTING PROCESS OF LINUX	Date :

Press the power button on your system, and after few moments you see the Linux login prompt. From the time you press the power button until the Linux login prompt appears, the following sequence occurs. The following are the 6 high level stages of a typical Linux boot process.


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

Step 1.BIOS

- BIOS stands for Basic Input/Output System
- Performs some system integrity checks
- Searches, loads, and executes the boot loader program.
- It looks for boot loader in floppy, CD-ROMs, or hard drive. You can press a key (typically F12 or F2, but it depends on your system) during the BIOS startup to change the boot sequence.
- Once the boot loader program is detected and loaded into the memory, BIOS gives the control to it.
- So, in simple terms BIOS loads and executes the MBR boot loader.

Step 2. MBR

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

Step 3. GRUB

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

```
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title CentOS(2.6.18-194.el5PAE)
root(hd0,0)
kernel/boot/vmlinuz-2.6.18-194.el5PAE ro root=LABEL=/
initrd /boot/initrd-2.6.18-194.el5PAE.img
```

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

Step 4. Kernel

- Mounts the root file system as specified in the "root=" in grub.conf
- Kernel executes the /sbin/init program
- Since init was the 1st program to be executed by Linux Kernel, it has the process id (PID) of 1. Do a 'ps -ef | grep init' and check the pid.
- initrd stands for Initial RAM Disk.
- initrd is used by kernel as temporary root file system until kernel is booted and the real root file system is mounted. It also contains necessary drivers compiled inside, which helps it to access the hard drive partitions, and other hardware.

Step 5. Init

- Looks at the /etc/inittab file to decide the Linux run level.
- Following are the available run levels
  - 0 – halt
  - 1 – Single user mode
  - 2 – Multiuser, without NFS
  - 3 – Full multiuser mode
  - 4 – unused
  - 5 – X11
  - 6 – reboot
- Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.

- Execute ‘grep initdefault /etc/inittab’ on your system to identify the default run level
- If you want to get into trouble, you can set the default run level to 0 or 6. Since you know what 0 and 6 means, probably you might not do that.
- Typically you would set the default run level to either 3 or 5.

Step 6. Runlevel programs

- When the Linux system is booting up, you might see various services getting started. For example, it might say “starting sendmail …. OK”. Those are the runlevel programs, executed from the run level directory as defined by your run level.
- Depending on your default init level setting, the system will execute the programs from one of the following directories.
  - Run level 0 – /etc/rc.d/rc0.d/
  - Run level 1 – /etc/rc.d/rc1.d/
  - Run level 2 – /etc/rc.d/rc2.d/
  - Run level 3 – /etc/rc.d/rc3.d/
  - Run level 4 – /etc/rc.d/rc4.d/
  - Run level 5 – /etc/rc.d/rc5.d/
  - Run level 6 – /etc/rc.d/rc6.d/
- Please note that there are also symbolic links available for these directory under /etc directly. So, /etc/rc0.d is linked to /etc/rc.d/rc0.d.
- Under the /etc/rc.d/rc*.d/ directories, you would see programs that start with S and K.
- Programs starts with S are used during startup. S for startup.
- Programs starts with K are used during shutdown. K for kill.
- There are numbers right next to S and K in the program names. Those are the sequence number in which the programs should be started or killed.
- For example, S12syslog is to start the syslog deamon, which has the sequence number of 12. S80sendmail is to start the sendmail daemon, which has the sequence number of 80. So, syslog program will be started before sendmail.

Login Process

1. Users enter their username and password
2. The operating system confirms your name and password.
3. A "shell" is created for you based on your entry in the "/etc/passwd" file
4. You are "placed" in your "home"directory.
5. Start-up information is read from the file named "/etc/profile". This file is known as the system login file. When every user logs in, they read the information in this file.
6. Additional information is read from the file named ".profile" that is located in your "home" directory. This file is known as your personal login file.

Ex. No. 2a	**BASIC LINUX COMMANDS**	**Date :**

a) Basics

1. *echo* SRM → to display the string SRM

2. *clear* → to clear the screen

3. *date* → to display the current date and time

4. *cal* 2003 → to display the calendar for the year 2003
   cal 6 2003 → to display the calendar for the June-2003

5. *passwd* → to change password

b) Working with Files

1. *ls* → list files in the present working directory
   ls –l → list files with detailed information (long list)
   ls –a → list all files including the hidden files

2. cat > f1 → to create a file (Press ^d to finish typing)

3. *cat* f1 → display the content of the file f1

4. *wc* f1 → list no. of characters, words & lines of a file f1
   wc –c f1 → list only no. of characters of file f1
   wc –w f1 → list only no. of words of file f1
   wc –l f1 → list only no. of lines of file f1

5. *cp* f1 f2 → copy file f1 into f2

6. *mv* f1 f2 → rename file f1 as f2

7. *rm* f1 → remove the file f1

8. *head* –5 f1 → list first 5 lines of the file f1
   tail –5 f1 → list last 5 lines of the file f1

c) Working with Directories

1. *mkdir* elias → to create the directory elias
2. *cd* elias → to change the directory as elias
3. *rmdir* elias → to remove the directory elias

4. *pwd* → to display the path of the present working directory

5. *cd* → to go to the home directory
   cd .. → to go to the parent directory
   cd - → to go to the previous working directory
   cd / → to go to the root directory

d) File name substitution

1. *ls f?* → list files start with 'f' and followed by any one character
2. *ls* *.c → list files with extension 'c'
3. *ls* [gpy]et → list files whose first letter is any one of the character g, p or y and followed by the word et
4. *ls* [a-d,l-m]ring → list files whose first letter is any one of the character from a to d and l to m and followed by the word ring.

e) I/O Redirection

1. Input redirection
wc –l < ex1 → To find the number of lines of the file 'ex1'

2. Output redirection
who > f2 → the output of 'who' will be redirected to file f2

3. *cat* >> f1 → to append more into the file f1

f) Piping

Syntax : Command1 | command2

Output of the command1 is transferred to the command2 as input. Finally output of the command2 will be displayed on the monitor.

ex. *cat* f1 | *more* → list the contents of file f1 screen by screen

head –6 f1 |*tail* –2 → prints the 5th & 6th lines of the file f1.

g) Environment variables

1. *echo $HOME* → display the path of the home directory
2. *echo $PS1* → display the prompt string $
3. *echo $PS2* → display the second prompt string (> symbol by default)
4. *echo $LOGNAME* → login name
5. *echo $PATH* → list of pathname where the OS searches for an executable file

h) File Permission

-- chmod command is used to change the access permission of a file.

Method-1

Syntax : *chmod* [ugo] [+/-] [rwxa] filename

u : user, g : group, o : others
+ : Add permission - : Remove the permission
r : read, w : write, x : execute, a : all permissions

ex. *chmod* ug+rw f1
adding 'read & write' permissions of file f1 to both user and group members.

Method-2

Syntax : *chmod* octnum file1

The 3 digit octal number represents as follows

first digit	-- file permissions for the user
second digit	-- file permissions for the group
third digit	-- file permissions for others

Each digit is specified as the sum of following
4 – read permission, 2 – write permission, 1 – execute permission

ex. *chmod* 754 f1

it change the file permission for the file as follows

- read, write & execute permissions for the user ie; 4+2+1 = 7
- read, & execute permissions for the group members ie; 4+0+1 = 5
- only read permission for others ie; 4+0+0 = 4

Verified by

Faculty In-charge Sign :	**Date :**

Ex. No. 2b	FILTERS and ADMIN COMMANDS	Date :

FILTERS

1. cut

- Used to cut characters or fileds from a file/input

Syntax : **cut** **-c**chars filename
-ffieldnos filename

- By default, tab is the filed separator(delimiter). If the fileds of the files are separated by any other character, we need to specify explicitly by **–d** option

cut **-d**delimitchar **-f**fileds filname

2. grep

- Used to search one or more files for a particular pattern.

Syntax : **grep** pattern filename(s)

- Lines that contain the *pattern* in the file(s) get displayed
- pattern can be any regular expressions
- More than one files can be searched for a pattern

-v	option displays the lines that do not contain the *pattern*
-l	list only name of the files that contain the *pattern*
-n	displays also the line number along with the lines that matches the *pattern*

3. sort

- Used to sort the file in order

Syntax : **sort** filename

- Sorts the data as text by default
- Sorts by the first filed by default

-r	option sorts the file in descending order
-u	eliminates duplicate lines
-o filename	writes sorted data into the file *fname*
-tdchar	sorts the file in which fileds are separated by *dchar*
-n	sorts the data as number
+1n	skip first filed and sort the file by second filed numerically

4. Uniq

- Displays unique lines of a sorted file

Syntax : **uniq** filename