Department of Computer Science & Information Technology

III Year V Semester(Batch 2022-2026)

Lab Record Submission of

Linux

Subject Code – CSIT-505

Submitted to:

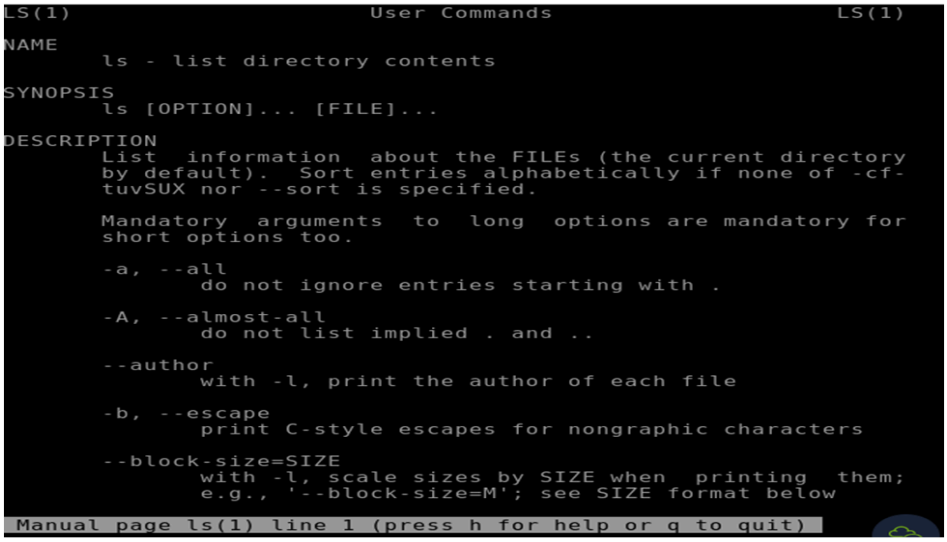
Prof.Nidhi Nigam

Submitted by:

Sneha Porwal

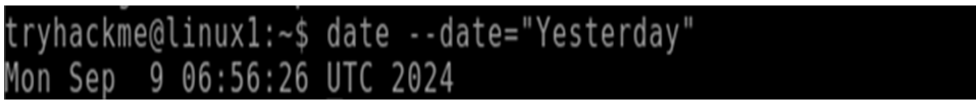
LINUX COMMANDS

1. **man command->**
   Description : Displays the manual page for a specified command.
   Syntax : man [command]
   Example: man ls



2. **whatis command ->**
   Description : Provides a brief description of a command.
   Syntax: whatis [command]
   Example: whatis pwd

3. **date command ->**
   Description : Displays or sets the system date and time.
   Syntax: date [option]...[+format]
   date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]
   Example: date -u



4. **ps command ->**
   Description : Shows a snapshot of the current processes.
   Syntax: ps[options]
   Example: ps

```
tryhackme@linux1:~$ ps
    PID TTY          TIME CMD
   1039 pts/0    00:00:00 bash
        /0       00:00:00 ps
```

5. **mkdir command ->**
   Description : Creates a new directory.
   Syntax: mkdir [directory]
   Example: mkdir GeeksForGeeks

```
tryhackme@linux1:~$ mkdir GeeksForGeeks
tryhackme@linux1:~$ pwd
/home/tryhackme
tryhackme@linux1:~$ mkdir CISCO
tryhackme@linux1:~$ ls
CISCO          access.log  folder2  folder4
GeeksForGeeks  folder1     folder3
```

6. **touch command ->**
   Description : Updates the access and modification times of a file or creates an empty file.
   Syntax: mkdir [directory]
   Example: mkdir GeeksForGeeks

```
labex:project/ $ touch test.txt
labex:project/ $ ls
test.txt
```

7. **rmdir command ->**
   Description : Removes empty directories.
   Syntax: rmdir [OPTION]... DIRECTORY...
   Example: rmdir newdocuments

```
tryhackme@linux1:~$ rmdir newdocuments
tryhackme@linux1:~$ cd newdocuments
-bash: cd: newdocuments: No such file or directory
```

8. **mov command ->**
   Description : Moves or renames files or directories.
   Syntax: mv [options(s)] [source_file_name(s)] [Destination_file_name]
   Example: mov country.txt State.txt

```
tryhackme@linux1:~$ mv country.txt State.txt
tryhackme@linux1:~$ ls
State.txt  access.log  folder1  folder2  folder3  folder4
```

9. **cd command ->**
   Description : Changes the current working directory.
   Syntax: cd [directory]
   Example: cd country.txt

```
tryhackme@linux1:~$ cd country.txt
tryhackme@linux1:~/country.txt$ cat > State.txt
Assam
```

10. **pwd command ->**
    Description : Prints the current working directory.
    Syntax: pwd [options]
    Example: pwd -L

```
tryhackme@linux1:~$ pwd -L
/home/tryhackme
```

11. **cat command ->**
    Description : Concatenates and displays the contents of files.
    Syntax: cat [OPTION]... [FILE]...
    Example: cat capital.txt

```
tryhackme@linux1:~/country.txt$ cat > capital.txt
Patna
Gujrat
Delhi
tryhackme@linux1:~/country.txt$ cat capital.txt
Patna
Gujrat
Delhi
```

12. **cat > command ->**
    Description : Redirects standard input to create or overwrite a file.
    Syntax: cat >[OPTION]... [FILE]...
    Example: cat > capital.txt

```
tryhackme@linux1:~/country.txt$ cat > capital.txt
Patna
Gujrat
Delhi
tryhackme@linux1:~/country.txt$ cat capital.txt
Patna
Gujrat
Delhi
```

13. **cat >> command ->**

Description : Appends standard input to an existing file.

Syntax: cat>> [OPTION]... [FILE]...

Example: cat >> t.txt

```
tryhackme@linux1:~/country.txt$ cat > t.txt
Hii
tryhackme@linux1:~/country.txt$ cat >> t.txt
Everyone
tryhackme@linux1:~/country.txt$ cat t.txt
Hii
Everyone
```

14. **cp command ->**

Description : Copies files or directories from one location to another.

Syntax: cp [OPTION]... SOURCE... DIRECTORY

Example: cp text.txt t.txt

```
tryhackme@linux1:~/Practice$ ls
t.txt
tryhackme@linux1:~/Practice$ cp t.txt t1.txt
tryhackme@linux1:~/Practice$ ls
t.txt  t1.txt
```

15. **chmod command ->**

Description : Changes file permissions.

Syntax: chmod [options] [mode] [File_name]

Example: chmod u+x t.txt

```
tryhackme@linux1:~/p$ cat t.txt
GeeksforGeeks
tryhackme@linux1:~/p$ chmod u+x t.txt
tryhackme@linux1:~/p$ ls -l
total 4
-rwxrw-r-- 1 tryhackme tryhackme 14 Sep 12 15:22 t.txt
tryhackme@linux1:~/p$ chmod u+rw t.txt
```

16. **cal command ->**

Description : Displays a calendar for a specific month or year.

Syntax: cal [[month] year]

Example: cal 04 2024

```
tryhackme@linux1:~$ cal 04 2024
     April 2024
Su Mo Tu We Th Fr Sa
    1  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
```

17. **ifconfig command - >**

Description : Configures or displays network interfaces.

Syntax: ifconfig [interface] [options]

Example: ifconfig

```
labex:project/ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.18.0.6  netmask 255.255.0.0  broadcast 172.18.255.255
        ether 02:42:ac:12:00:06  txqueuelen 0  (Ethernet)
        RX packets 35213  bytes 2738976 (2.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 30304  bytes 12911838 (12.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 6  bytes 384 (384.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6  bytes 384 (384.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

18. **echo command ->**

Description : Prints a string of text to the terminal.

Syntax:echo [option] [string]

Example: echo “Hello World!”

```
tryhackme@linux1:~$ echo "Hello World"
Hello World
tryhackme@linux1:~$
```

19. **grep command ->**

Description : Searches for patterns within files.

Syntax: grep [OPTION]... PATTERNS [FILE]...

Example: grep -l “unix” *

```
tryhackme@linux1:~$ grep -l "unix" *
grep: country.txt: Is a directory
grep: folder1: Is a directory
grep: folder2: Is a directory
grep: folder3: Is a directory
grep: folder4: Is a directory
```

20. **ls -l command ->**

Description : Lists files in a directory with detailed information.

Syntax: ls -l[OPTION]... [FILE]...

Example: ls -l country.txt

```
tryhackme@linux1:~$ ls -l country.txt
total 16
-rw-rw-r-- 1 tryhackme tryhackme 18 Sep 12 14:20 State.txt
-rw-rw-r-- 1 tryhackme tryhackme 19 Sep 12 14:21 capital.txt
-rw-rw-r-- 1 tryhackme tryhackme 15 Sep 12 14:28 state.txt
-rw-rw-r-- 1 tryhackme tryhackme 14 Sep 12 14:30 t.txt
```

21. **ls -a command ->**

Description:-Lists all files, including hidden files.

Syntax: : ls -a [OPTION]... [FILE]...

```
tryhackme@linux1:~$ ls -a
.           .bash history  .cache         access.log  folder3
..          .bash logout   .profile       folder1     folder4
.Xauthority .bashrc        GeeksForGeeks  folder2
```

22. **ls command ->**

Description:-Lists files in a directory.

Syntax: ls [OPTION]... [FILE]...

Example: ls -l

```
tryhackme@linux1:~$ ls -l
total 84
-rw-rw-r-- 1 tryhackme tryhackme 65522 May 10  2021 access.log
drwxrwxr-x 2 tryhackme tryhackme  4096 Sep 12 14:29 country.txt
drwxr-xr-x 2 tryhackme tryhackme  4096 May 10  2021 folder1
drwxr-xr-x 2 tryhackme tryhackme  4096 May 10  2021 folder2
drwxr-xr-x 2 tryhackme tryhackme  4096 May 10  2021 folder3
drwxr-xr-x 2 tryhackme tryhackme  4096 May 10  2021 folder4
```

23. whoami command ->

Description:-Displays the current logged-in user.

Syntax: : whoami [OPTION]...

Example: whoami

```
tryhackme@linux1:~$ whoami
tryhackme
tryhackme@linux1:~$
```

24. find command ->

Description:-Searches for files in a directory hierarchy.

Syntax: find [path] [options] [expression]

Example: find –type f –name note.txt

```
tryhackme@linux1:~$ find -type f -name note.txt
./folder4/note.txt
```

25. less command ->

Description:-Views file contents one screen at a time.

Syntax: less[OPTION][FILENAME]

Example: less –N t.txt

```
tryhackme@linux1:~$ less -N t.txt
      1 Linux is a free software
t.txt (END)
```

26. more command ->

Description:-Views file contents, similar to less, but with limited backward movement.

Syntax: more [options] <file>...

Example: more –f t2.txt

```
tryhackme@linux1:~$ cat > t2.txt
HII ,Linux was developed in 1991.It ia a free and open source software.
tryhackme@linux1:~$ more -f t2.txt
HII ,Linux was developed in 1991.It ia a free and open source software.
```

27. **join command ->**

Description:-Joins lines of two files on a common field.

Syntax: **join [OPTION]... FILE1 FILE2**

Example: **join t2.txt t3.txt**

```
tryhackme@linux1:~$ join t2.txt t3.txt
101 Aayush Kumar
102 Avinash Sharma
```

28. **paste command ->**

Description:-Merges lines of files horizontally.

Syntax: **paste [OPTION]... [FILE]...**

Example: **paste t2.txt t3.txt**

```
tryhackme@linux1:~$ cat > t2.txt
101 Aayush
102 Avinash
tryhackme@linux1:~$
tryhackme@linux1:~$ cat > t3.txt
101 Kumar
102 Sharma
tryhackme@linux1:~$ paste t2.txt t3.txt
101 Aayush      101 Kumar
102 Avinash     102 Sharma
```

29. **ln command ->**

Description:-Creates a hard or symbolic link to a file.

Syntax: **ln [OPTION]... [-T] TARGET LINK_NAME (1st form)**

ln [OPTION]... TARGET... DIRECTORY (2nd form)

ln [OPTION]... -t DIRECTORY TARGET... (3rd form)

Example: **ln program.py link.py**

```
tryhackme@linux1:~/P$ cat > Program.py
print("Hllo World!")
tryhackme@linux1:~/P$ ln Program.py link.py
tryhackme@linux1:~/P$ python3 link.py
Hllo World!
```

30. **head command ->**

Description:-Displays the first few lines of a file.

Syntax: **head [OPTION]... [FILE]...**

Example: **head –n 3 State.txt**

```
tryhackme@linux1:~/country.txt$ head -n 3 State.txt
Assam
Bihar
UP
```

31. **tail command ->**

Description:-Displays the last few lines of a file.

Syntax: tail [OPTION]... [FILE]...

Example: tail –n 2 capital.txt

```
tryhackme@linux1:~/country.txt$ tail -2 capital.txt
Gujrat
Delhi
```

32. **wc command ->**

Description:-Displays word, line, character, and byte counts for a file.

Syntax: wc [OPTION]... [FILE]...

Example: wc State.txt

```
tryhackme@linux1:~/country.txt$ wc State.txt
 4  4 18 State.txt
tryhackme@linux1:~/country.txt$ wc capital.txt
 3  3 19 capital.txt
```

33. **wc -l command ->**

Description:-Displays the number of lines in a file.

Syntax: wc [OPTION]... [FILE]...

Example: wc –l State.txt

```
tryhackme@linux1:~/country.txt$ wc -l State.txt
4 State.txt
```

34. **kill command ->**

Description:-Sends a signal to terminate a process using its process ID (PID).

Syntax: kill [signal] PID

Example: kill -9 1184

```
tryhackme@linux1:~$ ps
    PID TTY          TIME CMD
   1113 pts/1    00:00:00 bash
   1184 pts/1    00:00:00 bash
   1191 pts/1    00:00:00 ps
tryhackme@linux1:~$ kill -9 1184
Killed
tryhackme@linux1:~$ ps
    PID TTY          TIME CMD
   1113 pts/1    00:00:00 bash
   1192 pts/1    00:00:00 ps
```

35. **killall command ->**

Description:-Sends a signal to terminate all instances of a specific process by name**.**

Syntax: killall [signal] PID

Example: killall -l

```
tryhackme@linux1:~$ killall -l
HUP INT QUIT ILL TRAP ABRT BUS FPE KILL USR1 SEGV USR2 PIPE ALRM TERM STKFLT
CHLD CONT STOP TSTP TTIN TTOU URG XCPU XFSZ VTALRM PROF WINCH POLL PWR SYS
```

36. **rev command->**

Description:-Reverses the contents of each line in a file.

Syntax:rev [option][File…]

Example: rev t.txt , rev -V

```
tryhackme@linux1:~$ cat> t.txt
Linux is a free software
tryhackme@linux1:~$ rev t.txt
erawtfos eerf a si xuniL
tryhackme@linux1:~$ rev -V t.txt
rev from util-linux 2.34
```

38. **cmp command->**

Description:-Compares two files byte by byte and reports the first difference.

Syntax:cmp[OPTION]....FILE1 [FILE2 [SKIP1 [SKIP2]]]

Example: cmp t.txt t2.txt , cmp -b t.txt t2.txt

```
tryhackme@linux1:~$ cat > t.txt
GeeksfoGeeks is a coading and learning website.
tryhackme@linux1:~$ cat > t1.txt
Codechef is a coading website.
tryhackme@linux1:~$ cmp t.txt t1.txt
t.txt t1.txt differ: byte 1, line 1
tryhackme@linux1:~$ cmp -b t.txt t1.txt
t.txt t1.txt differ: byte 1, line 1 is 107 G 103 C
```

37. **rm command->**

Description:-Removes files or directories from the file system.

Syntax:rm[OPTION..][FILE….]

Example: rm t.txt

```
tryhackme@linux1:~/p$ ls
t.txt  t1.txt
tryhackme@linux1:~/p$ rm t.txt
tryhackme@linux1:~/p$ ls
t1.txt
```

39. **function command->**

Description:-Defines a shell function that groups several commands under a single name.

Syntax: **function name { COMMANDS ;}**

Example: **function hello**

```
tryhackme@linux1:~$ function hello
> {
> echo Hello,Welcome to Linux
> }
tryhackme@linux1:~$ hello
Hello,Welcome to Linux
```

40. **sort command->**

Description:-Sorts the lines of a file in alphabetical or numerical order.

Syntax: **sort [OPTION]... [FILE]...**

Example: **sort sortdemo.txt**

```
tryhackme@linux1:~$ cat sortdemo.txt
Aayush
Ansh
Akshay
Palak
Babita
tryhackme@linux1:~$ sort sortdemo.txt
Aayush
Akshay
Ansh
Babita
Palak
```

41. **split command->**

Description:-Splits a large file into smaller pieces based on specified criteria like number of lines.

Syntax: **split [OPTION]... [FILE [PREFIX]]**

Example: **split -l 2 course.tx**

```
tryhackme@linux1:~/Practice$ cat course.txt
M1
M2
ADA
DSA
DBMS
OOPM
tryhackme@linux1:~/Practice$ ls
course.txt  t.txt
tryhackme@linux1:~/Practice$ split -l 2 course.txt
tryhackme@linux1:~/Practice$ ls
course.txt  t.txt  xaa  xab  xac
```

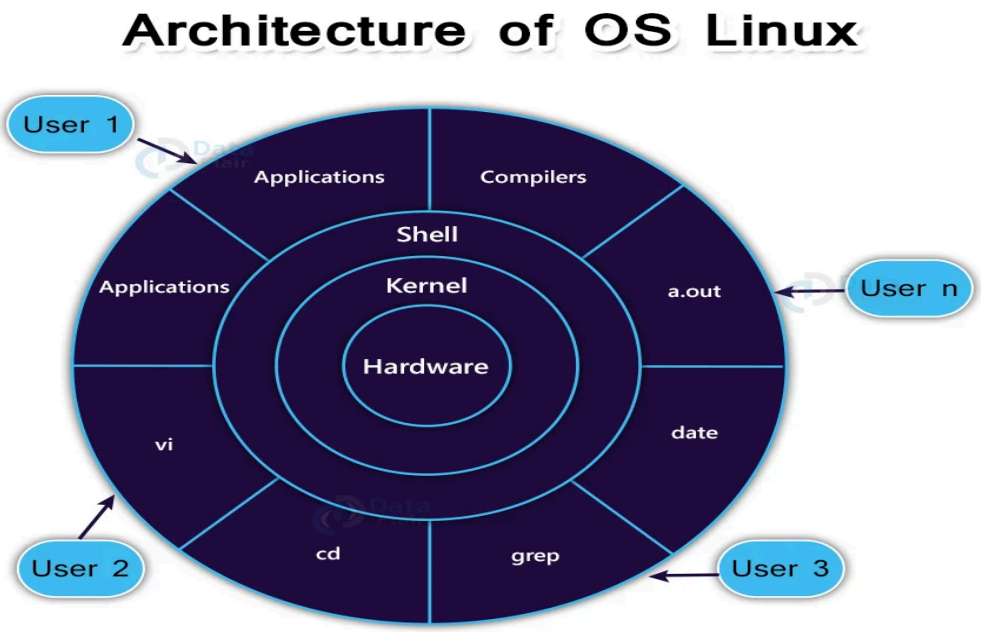
LAB-3
LINUX ARCHITECTURE

1. What is LINUX Architecture ?

The architecture of a Linux System consists of the following layers −

- **Hardware layer** − Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc).
- **Kernel** -
  - The kernel acts as a bridge between hardware and software, managing CPU, memory, and devices efficiently.
  - It provides essential services like process scheduling, memory allocation, file system handling, and device communication.
  - The kernel enforces user permissions, isolates processes, and regulates access to system resources for stability and security.

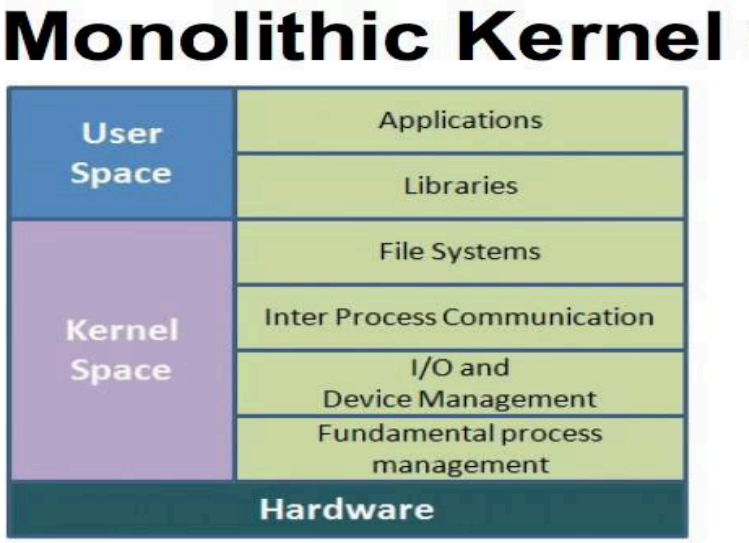
- **Shell** − An interface to kernel, hiding complexity of kernel's functions from users. The shell takes commands from the user and executes kernel's functions.
- **Utilities** − Utility programs that provide the user most of the functionalities of an operating systems



2.Types of Kernel :-

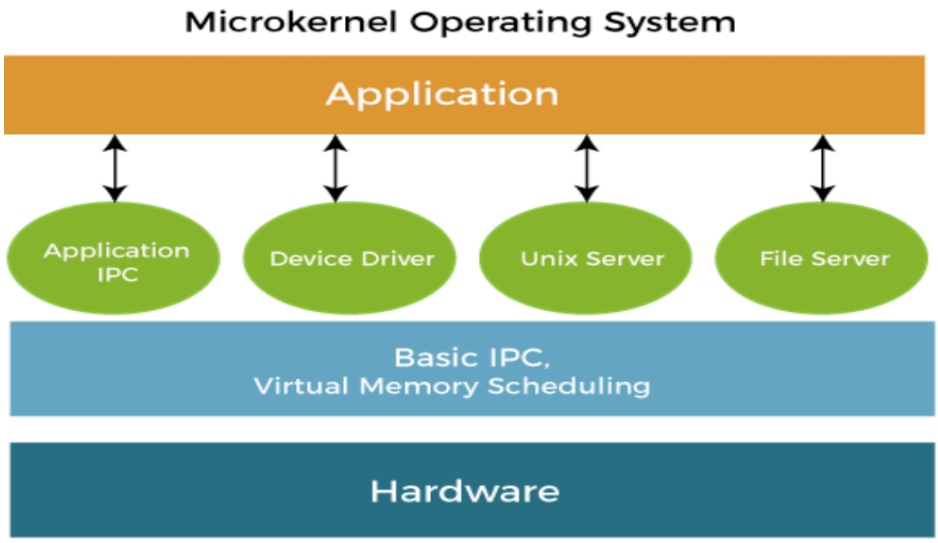
1.Monolithic Kernel

1. **Larger size:** Monolithic kernels house both kernel and user services in the same address space.
2. **Faster system calls:** They use a faster communication protocol for executing processes between hardware and software.
3. **Less flexible**: New services require reconstructing the entire kernel.
4. **Higher security risk**: If one service fails, the whole system shuts down.
5. **Less code**: Requires less source code than microkernels, leading to fewer bugs and less debugging.
6. **Example**: Linux Kernel uses a monolithic kernel architecture.



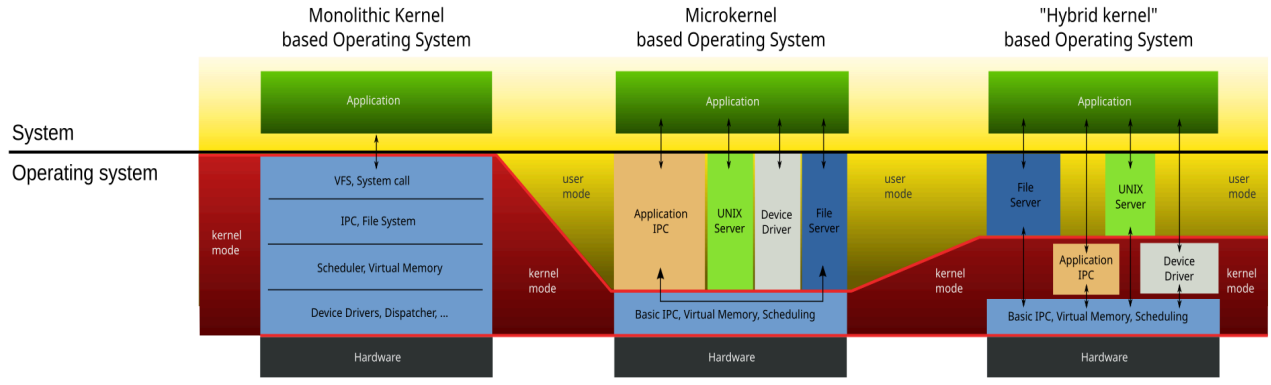
2.Micro kernel

1. **Separate address spaces**: Services run in different spaces from the kernel.
2. **Message passing**: Communication is done via message passing between processes.
3. **Flexibility**: Easy to add new services by modifying user space.
4. **Security**: Isolated services enhance security.
5. **Fault tolerance:** Kernel remains unaffected by service failures.
6. **Example**: MINIX 3 is a microkernel.



3.Hybrid Kernel

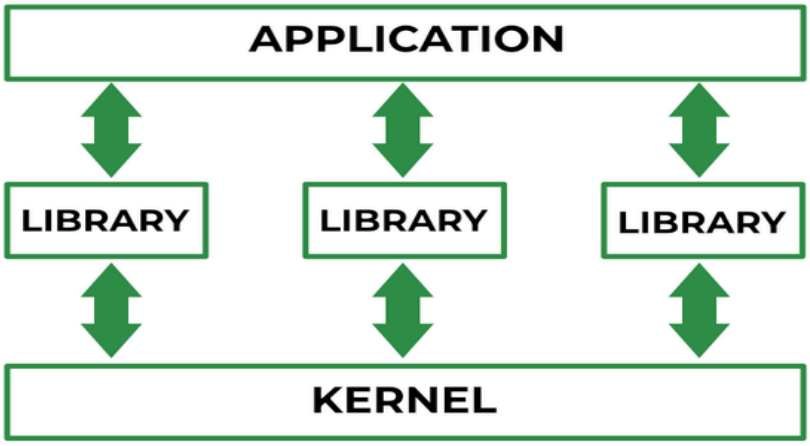
1. **Developed by Apple**: XNU kernel was developed in 1996, combining Mach and BSD kernels.
2. **Hybrid design**: Combines features of both monolithic and microkernel architectures for increased modularity and memory protection.
3. **Improved performance**: Hybrid kernels include extra code in kernel space to boost performance.
4. **Faster development:** They enable quicker development for third-party software.
5. **Examples**: Windows NT and macOS (since version X) use hybrid kernels.



Hybrid Kernel

4.Exo kernel

1. **Direct hardware access:** Exposes hardware resources directly to applications.
2. **Custom management:** Applications implement their own abstractions and management policies.
3. **Efficiency**: Provides better resource utilization and performance.
4. **Library OSes**: Offers different APIs to applications for flexibility.
5. **Performance Tuning**: Enable performance tuning without kernel intervention.
6. **Security:** Enhance security through application-enforced access controls.



Exo Kernel

3. What is Shell and Types of Shell ?

- A shell is a program through which users can interact with the Operating System.
- It interprets user commands and translates them into actions performed by the OS.
- Shells provide an interface for executing commands, managing files, and running applications.

Types of Shell:-

1. **C shell (csh):-**

    - Developed in the late 1970s at UC Berkeley, the C Shell (csh) aimed to improve user experience with syntax resembling the C programming language.
    - It offers enhanced scripting features like job control and command history for interactive use.

```
Command full-path name is /bin/csh,

Non-root user default prompt is hostname %,

Root user default prompt is hostname #.
```

2. The Bourne Shell(sh):-

- The original Unix shell, known for its simplicity and portability.
- It serves as the foundation for many other shells and is widely used for scripting.

```
Command full-path name is /bin/sh and /sbin/sh,

Non-root user default prompt is $,

Root user default prompt is #.
```

3. The Korn Shell(ksh):-

- Created by David Korn, ksh combines features from both sh and csh.
- It introduces advanced features like associative arrays and improved string manipulation, making it a powerful choice for scripting and interactive use.

```
Command full-path name is /bin/ksh,Non-root user
default prompt is $,


Root user default prompt is #.
```

4. . GNU Bourne-Again Shell:-

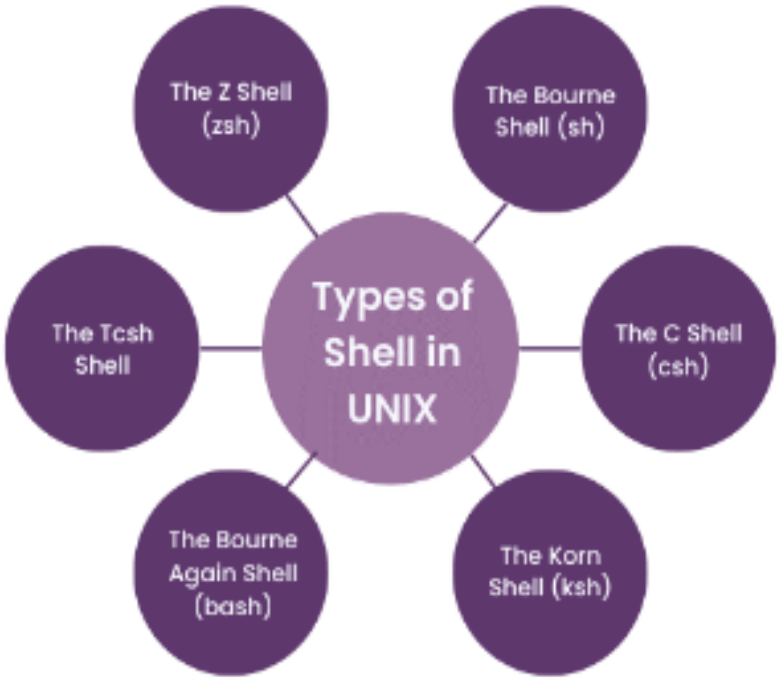
- An enhanced version of the Bourne Shell developed as part of the GNU project.
- It incorporates features from both ksh and csh, making it user-friendly and versatile, and is the default shell for most Linux distributions.

Command full-path name is /bin/bash,

Default prompt for a non-root user is bash-g.gg$

(g.ggindicates the shell version number like bash-3.50$),

Root user default prompt is bash-g.gg#.



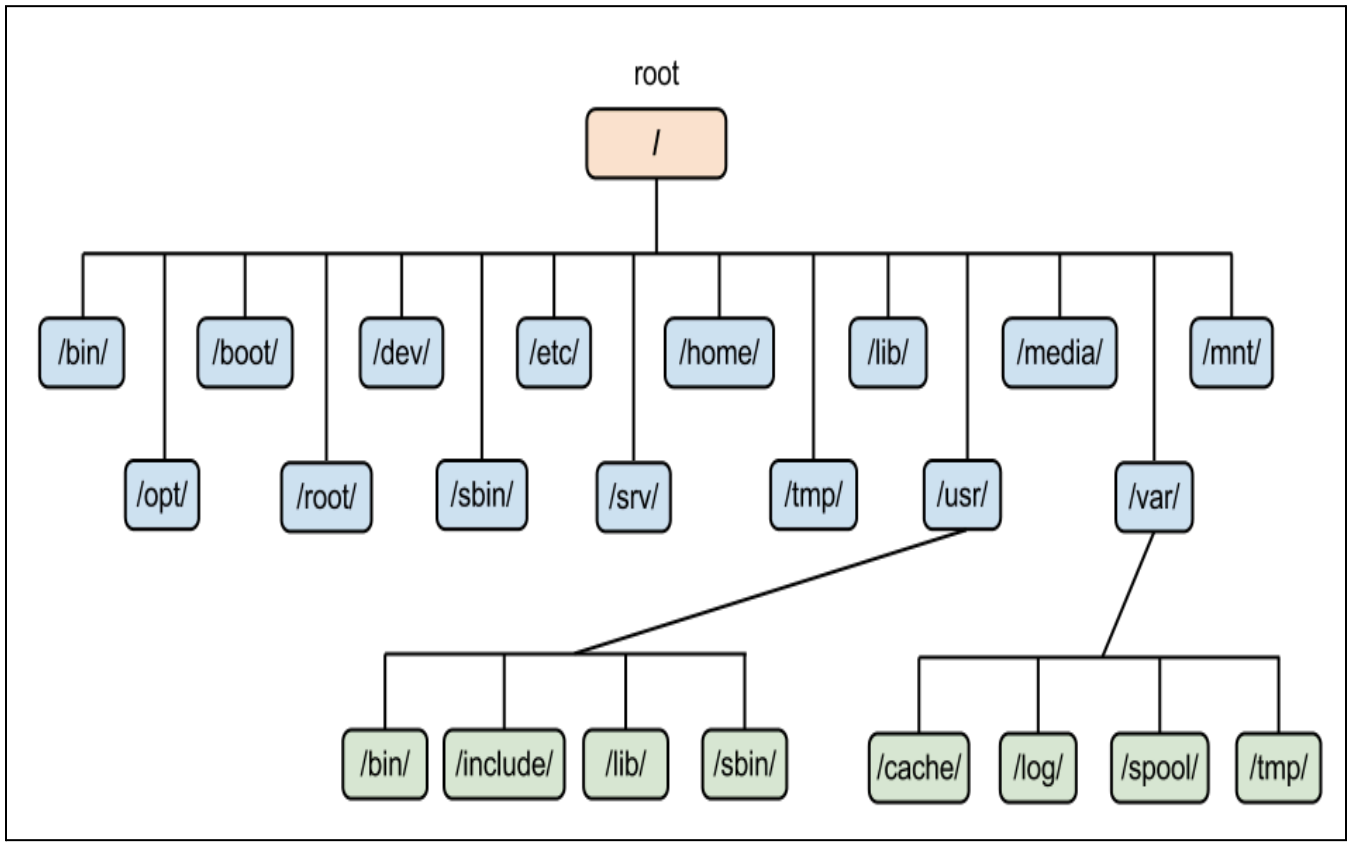
LAB-4
FILE SYSTEM

Linux file system:-

The **Linux file system** is a structured way to store and manage files on a disk drive or partition. Each partition can have a specific file system format (like **EXT4**, **XFS**, or **BTRFS**). It organizes and stores files on hard disks or other non-volatile memory, managing file names, sizes, creation dates, and more. The system ensures that data is systematically stored and easily accessible.

The Role of the Linux File System :

- ★ The data is organized and can be easilylocated
- ★ The data can be easily retrieved at any later point in time.
- ★ The integrity of the data is preserved.



Hierarchical Structure of the Linux FIle System

Root Directory (`/`):

- The root directory is at the top of the hierarchy. All other directories are branches from this root. It contains several subdirectories, each with a specific purpose.

Main Directories:

1. **/bin**: Contains essential binary executables (programs) needed by all users. Commands like `ls`, `cp`, and `mv` are stored here.
2. **/boot**: Contains files required for booting the system, including the Linux kernel.
3. **/dev**: Contains device files representing hardware devices. For example, disk drives, terminals, etc.
4. **/etc**: Contains configuration files for system-wide settings. System administrators often modify files here.
5. **/home**: Contains personal directories for users on the system. For example, `/home/user1` would be the directory for user1.
6. **/lib**: Stores essential shared libraries needed by the binaries in `/bin` and `/sbin`. It's similar to DLLs in Windows.
7. **/media**: Used for mounting removable media like USB drives, CDs, etc.
8. **/mnt**: Temporary mount points for filesystems.
9. **/opt**: Used for installing optional or third-party software.
10. **/root**: Home directory for the root user (system administrator).
11. **/sbin**: Contains system binaries used mainly for system administration tasks.
12. **/srv**: Contains files for services provided by the system, such as web servers.
13. **/tmp**: Temporary files created by users or applications are stored here.
14. **/usr**: Contains user programs, libraries, documentation, and more. It is often considered a secondary hierarchy. Subdirectories include:
    - **/usr/bin**: Non-essential binaries for users.
    - **/usr/include**: Header files for C programming.
    - **/usr/lib**: Libraries for programs in `/usr/bin`.
    - **/usr/sbin**: Non-essential system binaries.
15. **/var**: Stores variable files like logs, databases, mail, etc.
    - **/var/cache**: Cached data.
    - **/var/log**: System logs.
    - **/var/spool**: Data waiting to be processed, such as print jobs.