

Kathmandu University
Department of Computer Science and Engineering
Dhulikhel, Kavre



A Lab Report
on
“Operating System Lab-I”

[Code No.: COMP 307]

Submitted by

Manogya Dahal
Roll No.: 14

Submitted to
Ms. Rabina Shrestha
Department of Computer Science and Engineering

Questions

Q1: What is Linux?

Linux is an open-source operating system based on the Unix architecture. It manages hardware resources, executes commands, and provides a secure multi-tasking environment. It powers servers, desktops, embedded devices, and even supercomputers. Some of the most popular Linux distributions are ArchLinux, Ubuntu, RedHat, etc.

Q2: The Linux Hierarchical File System

Linux uses a hierarchical file structure that begins at the root directory /. Everything in Linux is a file or a directory, and all paths originate from /. Common directories include:

- /home – User home directories
- /bin – Essential command binaries
- /etc – Configuration files
- /usr – User utilities and applications
- /var – Logs, caches, temporary data

```
manogya@Arch ~ $ ls -lah
Permissions Size User Date Modified Name
lrwxrwxrwx  - root 12 Oct 22:06 bin  -> usr/bin
drwxr-xr-x  - root  4 Dec 12:33 boot
drwxr-xr-x  - root 10 Dec 12:45 dev
drwxr-xr-x  - root 10 Dec 15:55 etc
drwxr-xr-x  - root 19 Oct 2024 home
lrwxrwxrwx  - root 12 Oct 22:06 lib  -> usr/lib
lrwxrwxrwx  - root 12 Oct 22:06 lib64 -> usr/lib
drwxr-xr-x  - root  7 Apr 2024 mnt
drwxr-xr-x  - root  1 Dec 16:54 opt
dr-xr-xr-x  - root 10 Dec 12:45 proc
drwxr-xr-x  - root  6 Sep 22:15 root
drwxr-xr-x  - root 10 Dec 15:55 run
lrwxrwxrwx  - root 12 Oct 22:06 sbin -> usr/bin
drwxr-xr-x  - root 19 Oct 2024 srv
dr-xr-xr-x  - root 10 Dec 23:39 sys
drwxrwxrwt  - root 10 Dec 23:37 tmp
drwxr-xr-x  - root  4 Dec 13:48 usr
drwxr-xr-x  - root  5 Dec 13:50 var
```

Figure 1: Root Of My Linux System

Q3: Importance of Linux commands in Operating Systems

Linux commands are critical because they provide a direct and powerful interface to the operating system. They allow users and administrators to navigate the file system, manage files and directories, monitor system performance, and automate tasks through scripting. Unlike graphical interfaces, command-line commands are faster, use fewer resources, and provide more precise control. Mastering these commands enhances efficiency, troubleshooting capabilities, and overall understanding of how the OS operates, making it indispensable for system administrators, developers, and power users.

Linux Commands

1. **pwd**

The `pwd` command prints your current working directory. It tells you exactly where you are located inside the Linux file system. This is extremely useful when navigating through multiple folders, working in scripts, or verifying paths before executing commands that affect files. Since Linux uses a hierarchical file structure starting at the root `/`, `pwd` helps ensure you never get lost.

```
manogya@Arch ~/Programming/golang/Http$ pwd  
/home/manogya/Programming/golang/Http  
manogya@Arch ~/Programming/golang/Http$ []
```

2. **ls**

The `ls` command lists all files and directories in your current location. It gives a quick overview of the contents of a folder and is one of the most frequently used commands. By default, it shows only visible items (non-hidden files).

```
manogya@Arch ~/Programming/golang/Http$ ls  
cmd go.mod go.sum internal  
manogya@Arch ~/Programming/golang/Http$ []
```

3. **ls -a**

This version of `ls` displays all files, including hidden ones. Hidden files in Linux start with a dot (.), such as `.bashrc` or `.config`. These files usually store configurations and preferences.

```
manogya@Arch ~/Programming/golang/Http$ ls -a  
.git .gitignore cmd go.mod go.sum internal  
manogya@Arch ~/Programming/golang/Http$ []
```

4. **ls -l**

The `-l` option displays a long, detailed listing. It includes file permissions, owner, group, size, and last modification time. This format is essential for understanding access rights and managing file security.

```
manogya@Arch ~/Programming/golang/Http$ ls -l
drwxr-xr-x - manogya 12 Oct 11:31 cmd
.rw-r--r-- 243 manogya 27 Sep 13:11 go.mod
.rw-r--r-- 782 manogya 27 Sep 13:11 go.sum
drwxr-xr-x - manogya 12 Oct 11:32 internal
manogya@Arch ~/Programming/golang/Http$ []
```

5. `cd`

The `cd` command lets you move between directories. It is used to navigate the Linux filesystem. You can move into subdirectories, return to the parent directory using `cd ..`, or jump to a specific absolute path.

```
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum internal
manogya@Arch ~/Programming/golang/Http$ pwd
/home/manogya/Programming/golang/Http
manogya@Arch ~/Programming/golang/Http$ cd cmd
manogya@Arch ~/Programming/golang/Http/cmd$ pwd
/home/manogya/Programming/golang/Http/cmd
manogya@Arch ~/Programming/golang/Http/cmd$ []
```

6. `mkdir`

`mkdir` creates a new directory. It is commonly used to organize files by grouping them into folders. You can also create multiple folders at once, or even nested folders using `mkdir -p`.

```
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum internal
manogya@Arch ~/Programming/golang/Http$ mkdir hello
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum hello internal
manogya@Arch ~/Programming/golang/Http$ []
```

7. `rmdir`

This command removes an empty directory. It cannot delete directories that contain files. It is useful for cleaning up folder structures or removing temporary empty folders. To remove the folder there must be a folder that is created.

```
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum hello internal
manogya@Arch ~/Programming/golang/Http$ rmdir hello
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum internal
manogya@Arch ~/Programming/golang/Http$ []
```

8. **rm**

The rm command deletes files permanently (no recycle bin). It should be used carefully because deleted files are not easily recoverable. You can also remove multiple files at once.

```
manogya@Arch ~/Programming/golang/Http$ mkdir new
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum internal new
manogya@Arch ~/Programming/golang/Http$ rm new
rm: cannot remove 'new': Is a directory
manogya@Arch ~/Programming/golang/Http$ rm -r new
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum internal
manogya@Arch ~/Programming/golang/Http$ []
```

9. **rm -r folder_name**

The -r option stands for recursive deletion. It removes a directory and everything inside it — files, subfolders, and all. This is powerful and potentially dangerous, so double-check the directory before executing.

```
manogya@Arch ~/Programming/golang/Http$ mkdir new
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum internal new
manogya@Arch ~/Programming/golang/Http$ rm new
rm: cannot remove 'new': Is a directory
manogya@Arch ~/Programming/golang/Http$ rm -r new
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum internal
manogya@Arch ~/Programming/golang/Http$ []
```

10. **touch**

touch is used to create an empty file or update the timestamp of an existing file. It is commonly used in scripting or when preparing placeholder files.

```
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum internal
manogya@Arch ~/Programming/golang/Http$ touch hello
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum hello internal
manogya@Arch ~/Programming/golang/Http$ 
```

11. **cat**

The cat command reads and displays the content of a file directly in the terminal. It is also used to combine files or create files using output redirection.

```
manogya@Arch ~/Programming/golang/Http$ cat hello
I
USE
ARCH
BTW
manogya@Arch ~/Programming/golang/Http$ 
```

12. **nano, vi, jed**

These are terminal-based text editors. nano is beginner-friendly, vi (or vim) is a powerful editor popular among developers, and jed provides a lightweight interface. They allow editing, writing, and saving files directly from the terminal.

```
1  []
~
~
~
~
NVIM v0.12.0-dev-974+g2c3929624a
~
Nvim is open source and freely distributable
https://neovim.io/#chat
~
type :help nvim<Enter>      if you are new!
type :checkhealth<Enter>      to optimize Nvim
type :q<Enter>              to exit
type :help<Enter>            for help
~
type :help news<Enter>       to see changes in v0.12
~
Help poor children in Uganda!
type :help iccf<Enter>        for information
~
[No Name]                      0,0-1          All
```

13. **cp**

cp copies files from one place to another. You can also copy directories using

the -r option. This command is essential for backups, duplication, and organizing files.

```
manogya@Arch ~/Programming/golang/Http$ cat hello
I
USE
ARCH
BTW
manogya@Arch ~/Programming/golang/Http$ cp hello helloCopy
manogya@Arch ~/Programming/golang/Http$ cat helloCopy
I
USE
ARCH
BTW
manogya@Arch ~/Programming/golang/Http$ []
```

14. mv

mv allows you to move or rename files and directories. Renaming is simply a move within the same folder. It's used for reorganizing or updating file names.

```
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum hello helloCopy internal
manogya@Arch ~/Programming/golang/Http$ mv hello IuseArchBtw
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum helloCopy internal IuseArchBtw
manogya@Arch ~/Programming/golang/Http$ mv IuseArchBtw internal
manogya@Arch ~/Programming/golang/Http$ tree internal -L 1
internal
├── headers
├── IuseArchBtw
└── request
    └── server

4 directories, 1 file
manogya@Arch ~/Programming/golang/Http$ ls
cmd go.mod go.sum helloCopy internal
manogya@Arch ~/Programming/golang/Http$ []
```

15. echo

Prints text or variable values to the terminal. It is commonly used in scripts to produce messages, debug values, or write text into files using redirection.

```
manogya@Arch ~/Programming/golang/Http$ echo $USER
manogya
manogya@Arch ~/Programming/golang/Http$ echo "I use Arch BTW"
I use Arch BTW
manogya@Arch ~/Programming/golang/Http$ []
```

16. uname -a

Shows complete system information, including kernel version, machine architec-

ture, hostname, and operating system. Useful for debugging or checking system specs.

```
manogya@Arch ~/Programming/golang/Http$ uname  
Linux  
manogya@Arch ~/Programming/golang/Http$ uname -a  
Linux Arch 6.17.9-arch1-1 #1 SMP PREEMPT_DYNAMIC Mon, 24 Nov 2025 15:21:09 +0000 x86_64 GNU/Linux  
manogya@Arch ~/Programming/golang/Http$ 
```

17. **df -h**

Displays disk usage in human-readable format (MB/GB). It shows total size, used space, available space, and mounted filesystems. Handy for monitoring storage.

```
manogya@Arch ~/Programming/golang/Http$ df -h  
Filesystem      Size  Used Avail Use% Mounted on  
dev            7.4G   0    7.4G  0% /dev  
run            7.5G  1.9M  7.5G  1% /run  
efivarsfs     148K  106K  38K  74% /sys/firmware/efi/efivars  
/dev/nvme0n1p6 300G  50G  250G 17% /  
tmpfs          7.5G  4.0K  7.5G  1% /dev/shm  
tmpfs          1.0M   0    1.0M  0% /run/credentials/systemd-journald.service  
/dev/nvme0n1p6 300G  50G  250G 17% /root  
/dev/nvme0n1p6 300G  50G  250G 17% /home  
/dev/nvme0n1p6 300G  50G  250G 17% /srv  
/dev/nvme0n1p6 300G  50G  250G 17% /tmp  
/dev/nvme0n1p6 300G  50G  250G 17% /var/cache  
/dev/nvme0n1p6 300G  50G  250G 17% /var/log  
/dev/nvme0n1p5 1022M 300K 1022M  1% /boot/efi  
tmpfs          1.0M   0    1.0M  0% /run/credentials/getty@tty1.service  
tmpfs          1.5G  160K  1.5G  1% /run/user/1000  
manogya@Arch ~/Programming/golang/Http$ 
```

18. **ps -u \$USER**

Lists all currently running processes for your user. It displays process IDs, CPU usage, memory usage, and command names. Very useful for identifying unnecessary or stuck processes.

```
manogya@Arch ~/Programming/golang/Http$ ps -u $USER
  PID TTY      TIME CMD
 798 ?        0:00:00 systemd
 800 ?        0:00:00 (sd-pam)
 808 ?        0:00:00 dbus-broker-lau
 809 ?        0:00:00 dbus-broker
 810 ?        0:01:10 pipewire
 812 ?        0:01:09 wireplumber
 813 ?        0:01:24 pipewire-pulse
 815 ?        0:00:00 gnome-keyring-d
 817 ?        0:00:00 tmux: server
 847 tty2     0:00:00 ly-dm
 883 tty2     0:29:52 Xorg
1078 tty2     0:00:00 bash
1168 pts/1    0:00:00 zsh
1170 ?        0:00:02 xcape
1174 tty2     0:00:27 slstatus
1187 tty2     0:00:14 flameshot
1198 pts/2    0:00:00 zsh
1243 pts/3    0:00:00 zsh
1245 tty2     0:00:12 dunst
1279 pts/4    0:00:00 zsh
1304 pts/5    0:00:00 zsh
1316 tty2     0:00:13 kdeconnect-indi
1346 tty2     0:00:07 dwm
1350 pts/6    0:00:00 zsh
1413 pts/7    0:00:00 zsh
1464 pts/8    0:00:00 zsh
1496 ?        0:08:13 picom
1515 pts/9    0:00:00 zsh
1538 pts/10   0:00:00 zsh
1610 pts/11   0:00:00 zsh
1663 pts/12   0:00:00 zsh
1701 pts/13   0:00:00 zsh
1852 ?        0:00:00 xdg-permission-
1853 ?        0:00:00 f-disk
```

19. **top**

Displays real-time system activity. It shows active processes, CPU load, memory usage, and system uptime. It's one of the most important performance-monitoring commands.

```
top - 08:45:29 up 1 day, 6:36, 4 users, load average: 0.04, 0.02, 0.00
Tasks: 76 total, 1 running, 56 sleeping, 19 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 99.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7792.2 total, 3174.3 free, 1307.5 used, 3510.4 buff/cache
MiB Swap: 2048.0 total, 2048.0 free, 0.0 used, 5484.7 avail Mem

      PID USER      PR  NI  VIRT  RES  SHR  S %CPU %MEM TIME+ COMMAND
 273 mysql    20   0 234624 396456 36992 S  0.6  5.0 13:23.52 mysqld
 820 root     20   0 553728 117108 43528 S  0.6  1.5 10:28:73 mongod
  1 root     20   0 167852 119708 8704 S  0.0  0.0 0:00:00 init[systemd]
  2 root     20   0 167852 1970 1920 S  0.0  0.0 0:00:16 init[systemd](De
  6 root     20   0 3120 1792 1792 S  0.0  0.0 0:00:00 init
 58 root     20   0 49404 15360 14464 S  0.0  0.2 0:02.79 systemd-journal
 76 root     20   0 24904 5760 4608 S  0.0  0.1 0:11.86 systemd-udevd
178 root     20   0 1639736 30720 22408 S  0.0  0.5 0:09:21 bettercap
179 root     20   0 1639736 1920 1920 S  0.0  0.0 0:00:00 bettercap
188 message+ 20   0 8894 3968 3456 S  0.0  0.0 0:00:05 dbus-daemon
189 root     20   0 475808 20352 18304 S  0.0  0.3 0:00:08 nix-daemon
193 redis    20   0 60780 11392 8576 S  0.0  0.1 3:24.09 redis-server
210 root     20   0 16724 17424 6528 S  0.0  0.1 0:00:76 systemd-logind
214 root     20   0 394608 119708 9504 S  0.0  0.1 0:18:73 udisksd
220 root     20   0 167852 1536 1536 S  0.0  0.0 0:00:00 init[systemd]
240 root     20   0 5868 1920 1792 S  0.0  0.0 0:00:00 getty
265 root     20   0 4668 960 640 S  0.0  0.0 0:00:00 in.tftpd
271 root     20   0 6572 4756 3512 S  0.0  0.1 0:05:54 apache2
335 polkitd   20   0 234488 7024 6384 S  0.0  0.1 0:00:04 polkitd
368 postgres   20   0 216948 29072 27024 S  0.0  0.0 0:00:00 postgres
392 postgres   20   0 216932 3580 3148 S  0.0  0.1 0:00:06 postgres
393 postgres   20   0 216948 7172 4736 S  0.0  0.1 0:01:03 postgres
400 postgres   20   0 216868 10372 7936 S  0.0  0.1 0:01:06 postgres
401 postgres   20   0 218396 8092 6400 S  0.0  0.1 0:00:57 postgres
402 postgres   20   0 218378 8324 5632 S  0.0  0.1 0:00:57 postgres
701 debconf- 20   0 30532 17920 1792 S  0.0  0.2 0:07:07 debconf
752 root     20   0 3136 1194 3076 S  0.0  0.0 0:00:00 Relay(753)
754 root     20   0 5804 3456 3072 S  0.0  0.0 0:00:00 login
762 dragon   20   0 19200 10496 8704 S  0.0  0.1 0:00:20 systemd
763 dragon   20   0 168480 4708 1536 S  0.0  0.1 0:00:00 (sd-pam)
793 dragon   20   0 8874 4228 3956 S  0.0  0.1 0:00:01 zsh
795 dragon   20   0 15548 5968 3584 S  0.0  0.0 0:00:00 zsh[spido]
1065 dragon  20   0 7888 3840 3584 S  0.0  0.0 0:00:02 dbus-daemon
1068 dragon  20   0 311032 7296 6656 S  0.0  0.1 0:00:01 at-spi-bus-laun
1074 dragon  20   0 7784 3840 3584 S  0.0  0.0 0:00:00 dbus-daemon
```

20. **chmod**

Modifies file permissions. Permissions control who can read, write, or execute a file. chmod is essential for running scripts, securing files, and managing access rights.

```
manogya@Arch ~/Programming/golang/Http$ ls -lah helloCopy
Permissions Size User Date Modified Name
.rwXr-xr-x 16 manogya 11 Dec 00:14 helloCopy
manogya@Arch ~/Programming/golang/Http$ chmod -x helloCopy
manogya@Arch ~/Programming/golang/Http$ ls -lah helloCopy
Permissions Size User Date Modified Name
.rw-r--r-- 16 manogya 11 Dec 00:14 helloCopy
manogya@Arch ~/Programming/golang/Http$ 
```