Assignment 2: Basics of Linux and Open-Source Tools

Name: Purnima Singh
Roll Number: 2501730358

Introduction

This assignment is about learning the basics of Linux, using shell commands, and automating tasks with scripting. Linux is super important because lots of servers and software actually run on Linux, and understanding how it works is essential for any computer science student. Shell scripting and open-source tools also make things way more efficient, so I think this assignment is a good way to get practical skills early on.

Linux Installation Documentation

Because I have a MacBook, I didn't need to install Ubuntu using VirtualBox or WSL. macOS is based on Unix and supports most standard Linux shell commands directly in its Terminal app. So, all shell commands and scripts were executed on my Mac without needing to set up a virtual machine. This made the process simpler and let me try everything out right away.

Hardware Details:

- **Device:** MacBook Pro (Apple M4)
- **CPU:** Apple Silicon M4
- **RAM:** 24GB
- **Disk Space Available:** 512GB

Installation Steps:

1. **Opened Terminal:** Used the built-in macOS Terminal application (zsh/shell).
2. **Verified Linux Tools:** Checked basic commands (`ls`, `pwd`, `chmod`, etc.) work perfectly fine on macOS.
3. **No VM Required:** Skipped VirtualBox/Ubuntu ISO since everything was natively available.
4. **Ready for Practice:** Started directly with running shell commands and writing scripts.

Screenshots of my Terminal setup and hardware configuration are included in my evidence folder.

Shell Commands Table

I used several basic and useful shell commands. Here is a summary table with syntax, what they do, and why I used them:

Command	**Syntax**	**Description / Use**
ls	ls	List files and directories in current location
cd	cd /Users	Change directory to /Users
pwd	pwd	Print the working directory path
tree	tree	Show directory structure in tree form (requires install)
mkdir	mkdir docs	Make new directory called 'docs'
touch	touch hello.txt	Create empty file called hello.txt
cp	cp a.txt b.txt	Copy file from a.txt to b.txt
mv	mv old.txt new.txt	Rename (or move) old.txt to new.txt
rm	rm bye.txt	Delete file bye.txt
chmod	chmod 755 script.sh	Change permissions for script.sh
chown	chown user file	Change owner of file to user
ps	ps aux	List all active processes
top	top	Show real-time system processes and usage
kill	kill 1234	Stop process with ID 1234
ping	ping google.com	Test network connection to google.com
ifconfig	ifconfig	Display IP/network interfaces (macOS version)
ip	ip addr	Show network addresses (only on Linux, not macOS)
netstat	netstat -an	Show network connections and listening ports
cat	cat file.txt	Display contents of file.txt
echo	echo "Hello"	Print text to terminal
find	find . -name file.txt	Search for file.txt in current folder and subfolders

Table 1: Summary of Linux Shell Commands Used

All commands above were executed in the Mac's Terminal app. Some networking commands like ip are not by default available on macOS, but most standard Linux commands work the same way. Screenshots of sample outputs are kept for evidence. I found commands like chmod, chown, ps, and kill super useful for managing files, permissions, and processes, especially as a beginner.

Shell Scripts

Here are my three shell scripts with a short explanation and sample code:

1. Backup a Directory Script

#!/bin/bash

Purpose: Backup a specified directory with timestamp

Author: Purnima Singh

src_dir="$HOME/Documents''backup_{d}ir =''$HOME/Backups"
timestamp=$(date +' mkdir - p''$backup_dir"
cp -r "$src_{d}ir''''$backup_dir/backup_${timestamp}"

echo "Backup of $src_dir completed to {timestamp}"

Explanation:
This script takes the Documents folder and copies it to Backups with the current date and time in the folder name, so it doesn't overwrite older backups. I ran this script in the macOS terminal and it worked fine.

---

2. CPU and Memory Monitoring Script

#!/bin/bash

Purpose: Log CPU and memory usage at intervals

Author: Purnima Singh

logfile="$HOME/cpu_{m}em_{l}og.txt''whiletruedotop - l1|head - n10 >>''$logfile"
echo "------" >> "$logfile"
sleep 60
done

Explanation:
This watches CPU and memory usage every minute and adds results to a log file. On macOS,

the top command uses different flags (like -l 1) but the idea is the same. Seeing live resource usage was cool and helped me understand actual system performance.

---

3. Automated Download Script

#!/bin/bash

Purpose: Download a file from internet to a folder

Author: Purnima Singh

download_dir="
$HOME/Downloads''url = ''https://example.com/sample.pdf''mkdir - p''$
download_dir"
curl -o "$download_{d}ir/sample.pdf''''$url"

echo "Downloaded $url to $download_dir"

Explanation:
This script downloads a file using curl (which comes built-in on macOS) and saves it in Downloads. I added a mkdir command so the script always works. If I run it in my Mac's terminal, the file appears instantly.

Reflection

Challenges I Faced

1. **Installation:**
    - Using macOS made setup and getting started much easier because I didn't need to install a whole new OS or VM.
    - I did have to look up how certain commands differ between Linux and macOS; sometimes the flags are not quite the same, and some networking tools like ip didn't work.
    - The Terminal environment was really welcoming since most commands just worked.
2. **Shell Scripting Errors:**
    - Forgot to add execute permissions (chmod +x) sometimes—this is important on any Unix system!
    - Debugged infinite loop in the CPU/mem script (needed sleep 60).
3. **Command Syntax:**
    - Mixed up arguments for cp and mv at first.
    - Having real error messages in Terminal helped me learn what to fix, which I think is the real value of hands-on work.

Learning Outcomes

- Got hands-on with Linux and Unix basics right on my Mac, without messing with VMs.
- Using the terminal makes me feel confident I could use Linux on a server if needed.
- Shell scripting and automating tasks feels powerful and helps save a lot of time.
- I learned how cross-platform shell commands work—the similarities and some small differences between Linux and macOS.
- Pushing code and scripts to GitHub lets me organize, share, and document my work for future learning.

Applying Skills in Real World

If I work on cloud servers, remote deployments, or DevOps, these skills will be directly useful. I now know how to automate backups, monitor system health, and interact with files using command-line tools.

Conclusion

This assignment gave me a beginner-friendly dive into Linux and open-source tools, but on my MacBook. Using the Mac's Terminal was super easy, and I feel a lot more confident with shell commands and scripting. Solving problems on my own was sometimes tricky, but also rewarding. These skills will really help me when I have to use actual Linux servers in higher semesters or in real jobs.
