Experiment: [Daily System Logger Script]

Name: Biswabandya Mohanty, Roll No.: 590029274 Date: 2025-11-26

AIM:

- To create a shell script that logs current system information, rotates old logs, and schedules itself to run daily.

Requirements:

- Any Linux Distro(mint)
- Any text editor (VS Code, Vim, Nano, etc.)
- Cron service for scheduling

Theory:

In system administration, automated logging is crucial for monitoring system performance, diagnosing issues, and maintaining records.
This experiment involves:

1. Logging details like username, date, processes, and disk usage.
2. Archiving old logs automatically.
3. Scheduling the script to run daily using `cron`.

Procedure & Observations

Exercise 1: Creating the Daily Log Script

Task Statement:

Write a shell script that logs system info and handles automatic rotation of old logs.

Explanation:

This script:

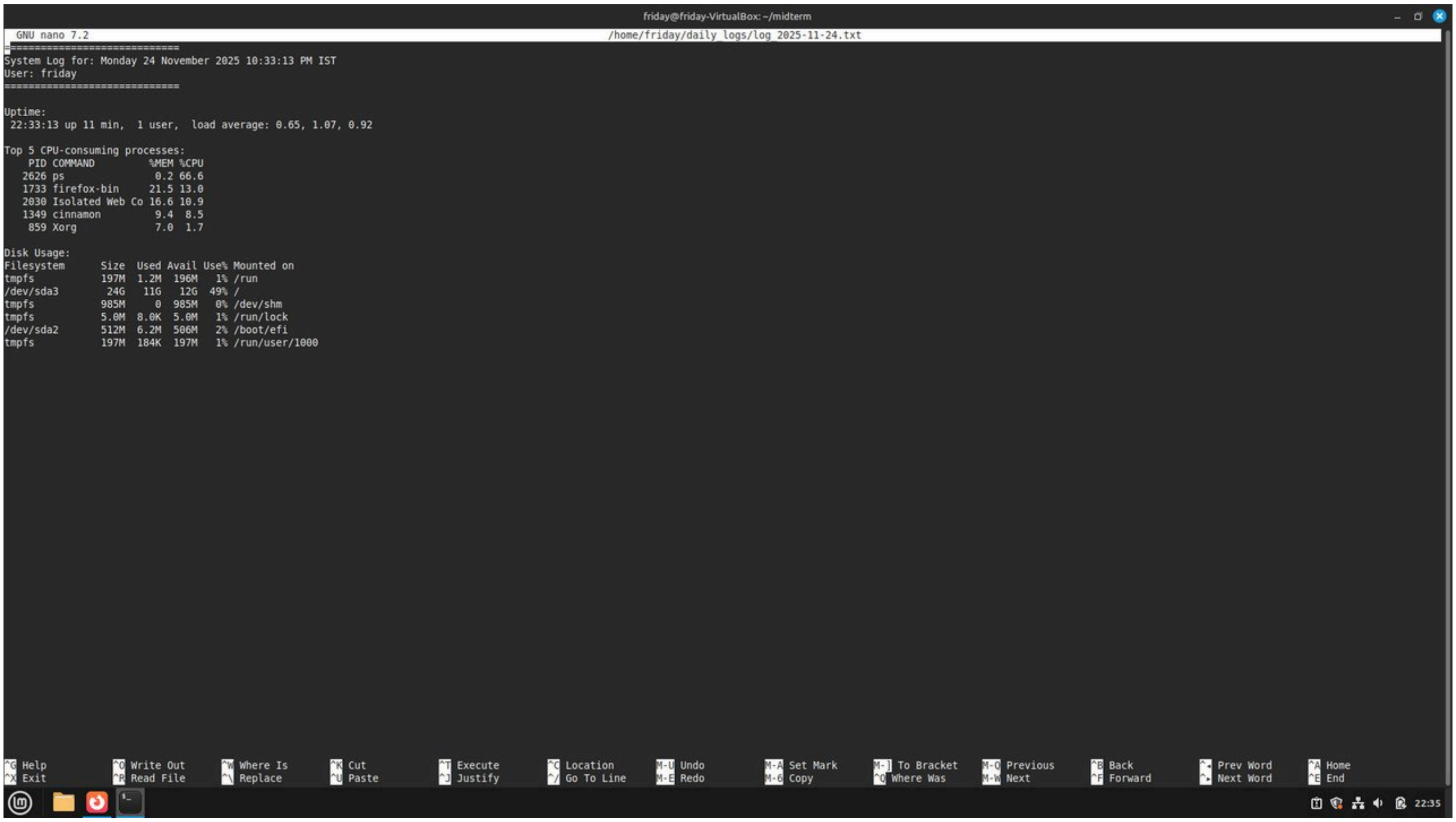
- Identifies the current user.
- Creates a directory for storing logs.

- Saves daily logs with timestamps.
- Archives logs older than 7 days.
- Can be scheduled using a cron job.

Command(s):

```bash
#!/bin/bash
SYS_D="$HOME/daily_logs"
ARCHIVE_DIR="$SYS_D/archive"
mkdir -p "$ARCHIVE_DIR"


LOG_FILE="$SYS_D/log_$(date +%Y-%m-%d).txt"


{
  echo "============================="
  echo "System Log for: $(date)"
  echo "User: $(whoami)"
  echo "============================="
  echo
  echo "Uptime:"
  uptime
  echo
  echo "Top 5 CPU-consuming processes:"
  ps -eo pid,comm,%mem,%cpu --sort=-%cpu | head -n 6
  echo
  echo "Disk Usage:"
  df -h
} > "$LOG_FILE"


find "$SYS_D" -name "log_*.txt" -mtime +7 -exec mv {} "$ARCHIVE_DIR" \;


if [ "$(date +%u)" -eq 7 ]; then
  tar -czf "$ARCHIVE_DIR/weeklylogs_$(date +%Y-%m-%d).tar.gz" -C "$ARCHIVE_DIR" .
fi
```

Output Example:



friday@friday-VirtualBox: ~/midterm
GNU nano 7.2
/home/friday/daily_logs/log_2025-11-24.txt

```
=============================
System Log for: Monday 24 November 2025 10:33:13 PM IST
User: friday
=============================

Uptime:
 22:33:13 up 11 min,  1 user,  load average: 0.65, 1.07, 0.92

Top 5 CPU-consuming processes:
    PID COMMAND         %MEM %CPU
   2626 ps               0.2 66.6
   1733 firefox-bin     21.5 13.0
   2030 Isolated Web Co 16.6 10.9
   1349 cinnamon         9.4  8.5
    859 Xorg             7.0  1.7

Disk Usage:
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           197M  1.2M  196M   1% /run
/dev/sda3        24G   11G   12G  49% /
tmpfs           985M     0  985M   0% /dev/shm
tmpfs           5.0M  8.0K  5.0M   1% /run/lock
/dev/sda2       512M  6.2M  506M   2% /boot/efi
tmpfs           197M  184K  197M   1% /run/user/1000
```

Exercise 2: Scheduling the Script

Task Statement:

Schedule the above script to run daily using cron.

Explanation:

Use crontab to automate the script execution at a fixed time every day.

Command(s):

```
#!/bin/bash
crontab -l+%Y-%m-%d
-name "log_*.txt" -mtime +7 to identify old files.
 df -h.
 tar -czf weeklylogs_$(date +%Y-%m-%d).tar.gz.

Store archives in ~/daily_logs/archive.

## Scheduling(cron job)
Using crontab -e to schedule the script to run everyday at a fixed time.
eg.
```

0 20 * * * /home/biswa/mid/mid.sh

```
<p align="center">
<img src="/img/mid2.jpg" width="900">
</p>


## Result:
The script successfully logs daily system information, archives logs older than 7 days, and sche
```