

# Daily System Logger Script

---

Name: RAHUL

**SAP ID:** 590029148

**Date:** 2025-11-29

---

## AIM

To create and schedule a shell script that:

- Logs system information daily
  - Stores logs in a dedicated folder
  - Rotates logs older than 7 days
  - Runs automatically using **cron**
- 

## Requirements

- Any Linux distribution (e.g., Linux Mint / Ubuntu / Pop!\_OS)
  - Text editor (Nano, Vim, VS Code, etc.)
  - Cron service enabled
- 

## Theory

System administrators often automate logs to track performance and system health. This experiment focuses on:

1. Collecting system data: processes, memory, disk usage, user info
  2. Saving daily logs with date-based filenames
  3. Deleting old logs automatically
  4. Scheduling tasks using **cron**
- 

## Exercise 1: Creating the Daily Log Script

### Task:

Create a Bash script that saves system info and rotates logs older than 7 days.

### Script (daily\_log.sh):

```
#!/bin/bash

LOG_DIR="$HOME/daily_logs"
```

```

mkdir -p "$LOG_DIR"

# Create a log file with today's date
LOG_FILE="$LOG_DIR/log_$(date +"%Y-%m-%d").txt"

# Record system information
{
    echo "==== Daily System Log ====="
    echo "Date: $(date)"
    echo "User: $USER"
    echo

    echo "==== Top 5 Running Processes (by CPU) ====="
    ps -eo pid,user,comm,%cpu,%mem --sort=-%cpu | head -n 6
    echo

    echo "==== Disk Usage ====="
    df -h
    echo

    echo "==== Memory Usage ====="
    free -h
} > "$LOG_FILE"

# Delete logs older than 7 days
find "$LOG_DIR" -type f -name "log_*.txt" -mtime +7 -exec rm {} \;

echo "Log saved to: $LOG_FILE"

```

## Output Screenshot:

```
linuxmint@DESKTOP-KSC4L9I x + v
GNU nano 7.2 daily_log.sh
#!/bin/bash

LOG_DIR="/home/linuxmint/daily_logs"
mkdir -p "$LOG_DIR"

# Create a log file with today's date
LOG_FILE="$LOG_DIR/log_$(date +%Y-%m-%d).txt"

# Record system info
{
    echo "==== Daily System Log ====="
    echo "Date: $(date)"
    echo "User: $USER"
    echo
    echo "==== Top 5 Running Processes (by CPU) ====="
    ps -eo pid,user,comm,%cpu,%mem --sort=-%cpu | head -n 6
    echo
    echo "==== Disk Usage ====="
    df -h
    echo
    echo "==== Memory Usage ====="
    free -h
} > "$LOG_FILE"

# Keep only the last 7 logs (rotate logs)
find "$LOG_DIR" -type f -name "log_*.txt" -mtime +7 -exec rm {} \;

echo "Log saved to: $LOG_FILE"
```

---

## Exercise 2: Scheduling the Script Using Cron

### Task:

Schedule the script to run daily.

### Steps:

1. Open crontab:

```
crontab -e
```

2. Add this line:

```
0 20 * * * /home/linux/daily_log.sh
```

This runs the script.

### Cron Format Reminder:

```
m h dom mon dow command
```

## Cron Screenshot:

```
linuxmint@DESKTOP-K5C4L9I x + v
GNU nano 7.2 /tmp/crontab.QSa6yh/crontab
0 20 * * * /home/linuxmint/scripts/daily_log.sh
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
```

---

## Final Output

The script successfully:

- Creates a new log file daily
- Shows running processes, memory usage, disk usage
- Deletes old logs (older than 7 days)
- Runs automatically via cron

```
linuxmint@DESKTOP-KSC4L9L: /mnt/e/linux$ nano daily_log.sh
linuxmint@DESKTOP-KSC4L9L: /mnt/e/linux$ cat ~/daily_logs/log_2025-11-12.txt
===== Daily System Log =====
Date: Wed Nov 12 20:08:01 UTC 2025
User:

===== Top 5 Running Processes (by CPU) =====
  PID USER      COMMAND      %CPU %MEM
   1 root        systemd      0.0  0.1
 308 root        Relay(309)   0.0  0.0
   44 root        systemd-journal 0.0  0.2
 181 root        wsl-pro-service 0.0  0.1
 309 linuxmi+    bash         0.0  0.0

===== Disk Usage =====
Filesystem      Size  Used Avail Use% Mounted on
none             3.9G   0    3.9G   0% /usr/lib/modules/6.6.87.2-microsoft-standard-WSL2
none             3.9G  4.0K   3.9G   1% /mnt/wsl
drivers          317G 145G  173G  46% /usr/lib/wsl/drivers
/dev/sdd         1007G 1.8G  954G   1% /
none             3.9G   76K   3.9G   1% /mnt/wslg
none             3.9G   0    3.9G   0% /usr/lib/wsl/lib
rootfs           3.9G 2.7M   3.9G   1% /init
none             3.9G 508K   3.9G   1% /run
none             3.9G   0    3.9G   0% /run/lock
none             3.9G   0    3.9G   0% /run/shm
none             3.9G   76K   3.9G   1% /mnt/wslg/versions.txt
none             3.9G   76K   3.9G   1% /mnt/wslg/doc
C:\              317G 145G  173G  46% /mnt/c
D:\              318G  34G  285G  11% /mnt/d
E:\              318G 207G  112G  65% /mnt/e
tmpfs            3.9G  16K   3.9G   1% /run/user/1000

===== Memory Usage =====
              total        used        free      shared  buff/cache   available
Mem:          7.7Gi         496Mi        7.1Gi         3.4Mi        205Mi        7.2Gi
Swap:         2.0Gi           0B         2.0Gi

linuxmint@DESKTOP-KSC4L9L: /mnt/e/linux$
```

## Conclusion

This experiment demonstrates automation using shell scripting and cron. The daily log script efficiently captures system information and maintains a clean log directory by removing older log files. Cron ensures the script runs consistently without manual intervention.