



UNIVERSITY INSTITUTE *of*
COMPUTING
Asia's Fastest Growing University



Project:
System Health Checker (Shell Script)

Course Code- 24CAP-607

LINUX ADMINISTRATION LAB



MASTERS IN COMPUTER APPLICATION

Submitted By

Name- Radheshyam Singh
UID-24MCA20360
Branch- MCA
Section- 6(A)

Submitted To

Prof. (Dr.) Arun Kumar Singh

Project: System Health Checker (Shell Script)

This shell script will check and report the health of your Linux system, including CPU usage, memory usage, disk usage, and running processes. It's a great way to get hands-on with Linux command-line tools and scripting.

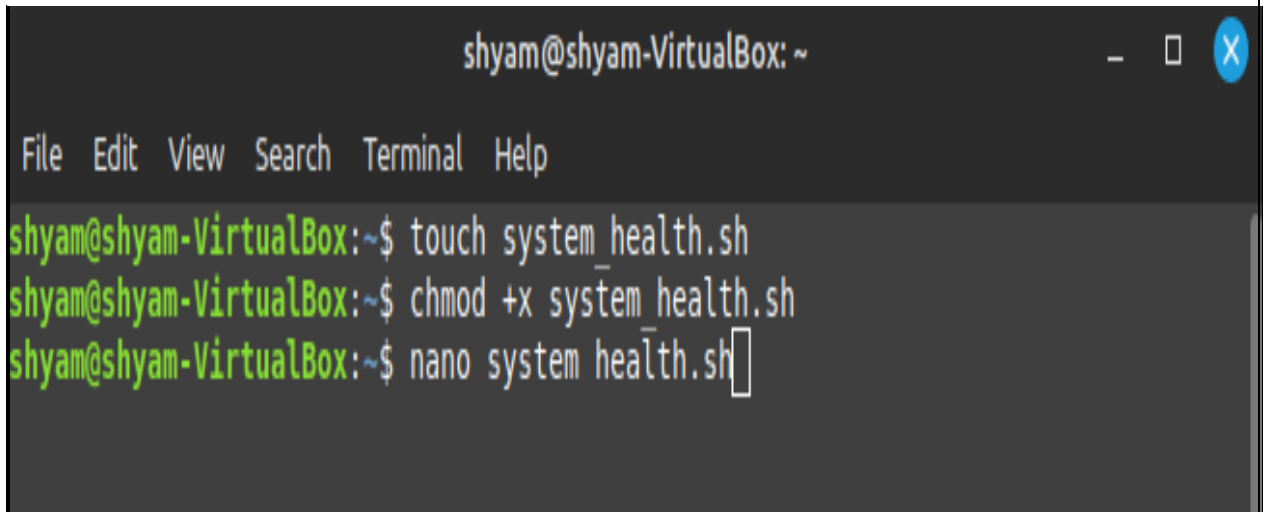
Step 1: Plan the Script

The script will need to:

1. Check CPU usage.
2. Check memory usage.
3. Check disk usage.
4. Display a list of the top 5 processes by CPU usage.
5. Display the current system uptime.

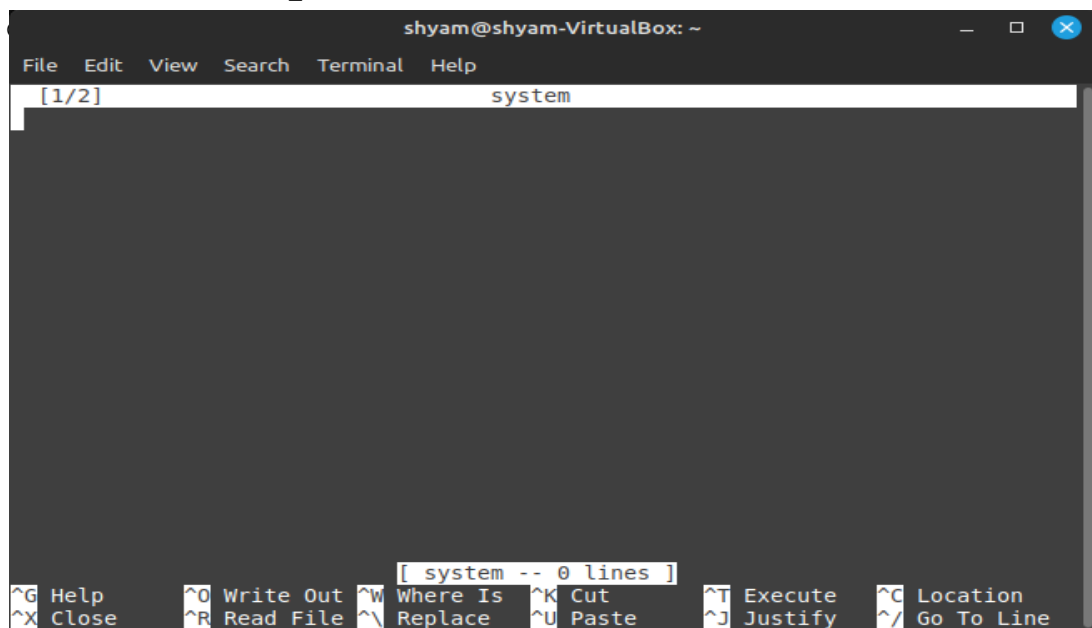
Step 2: Create the Script

1. **Create a new file for the script:** Open a terminal and create a new shell script file:

A screenshot of a terminal window titled 'shyam@shyam-VirtualBox: ~'. The terminal has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The prompt is 'shyam@shyam-VirtualBox:~\$'. Three commands are entered: 'touch system health.sh', 'chmod +x system health.sh', and 'nano system health.sh'. The cursor is at the end of the third command.

```
shyam@shyam-VirtualBox: ~  
File Edit View Search Terminal Help  
shyam@shyam-VirtualBox:~$ touch system health.sh  
shyam@shyam-VirtualBox:~$ chmod +x system health.sh  
shyam@shyam-VirtualBox:~$ nano system health.sh
```

2. **Write the script:** Open the file with your preferred text editor (nano, vim,



Now, write the following code in the script:

```
#!/bin/bash

# System Health Checker Script
# This script will check CPU, Memory, Disk Usage and Top Processes

echo "----- System Health Checker -----"
echo

# Check CPU usage
echo "CPU Usage:"
top -bn1 | grep "Cpu(s)" | sed "s/./, *\([0-9.]*\)%* id.*/\1/" | awk '{print "CPU Usage: "
echo

# Check Memory usage
echo "Memory Usage:"
free -h | grep Mem | awk '{print "Used: " $3 " / Total: " $2 " (" $3/$2*100 "%)"}'
echo

# Check Disk usage
echo "Disk Usage:"
df -h | grep '^/dev' | awk '{print $1 " - " $5 " used, " $3 " free of " $2 " total"}'
echo

# Display Top 5 processes by CPU usage
echo "Top 5 processes by CPU usage:"
ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%cpu | head -n 6
echo

# Display System Uptime
echo "System Uptime:"
uptime
```

Explanation of the code:

- `top -bn1` runs the `top` command in batch mode, providing the CPU usage.
- `free -h` shows memory usage in a human-readable format.
- `df -h` shows disk usage in a human-readable format.
- `ps -eo pid,ppid,cmd,%mem,%cpu` shows running processes with CPU and memory usage, sorted by CPU usage.
- `uptime` displays the system uptime.

3. **Save and close** the editor (in nano, press CTRL + X, then Y to confirm and Enter to save).

Step 3: Run the Script

In your terminal, navigate to the folder where the script is saved (if you're not already there). Then run the script:

```
bash
Copy code
./system_health.sh
```

It will display the system health check results including CPU usage, memory usage, disk usage, top processes by CPU usage, and the system uptime.

Step 4: Customize and Extend (Optional)

You can enhance the script by adding more checks or alerts. For example:

- Set thresholds for CPU, memory, or disk usage, and have the script alert you if they exceed certain values.
- Schedule this script to run periodically using `cron` to keep track of system health over time.
- Send an email or Slack notification if a problem is detected.

Step 5: Automate with Cron

To make the health checker run periodically (e.g., every hour), you can add it to `cron`. Run the following command to edit the crontab:

bash

```
CPU_THRESHOLD=80
cpu_usage=$(top -bn1 | grep "Cpu(s)" | sed "s/.*, *\[0-9.\]* id.*\/1/" | awk '{print $1}' | tr -d ' ' | fold -w 64 | paste | sed 's/^[ \t]*//g' | bc)
if (( $(echo "$cpu_usage > $CPU_THRESHOLD" | bc -l) )); then
    echo "Warning: CPU usage is above threshold ($cpu_usage%)"
fi
```

crontab -e

Add a line like this to run the script every hour:

bash

```
if (( $(echo "$cpu_usage > $CPU_THRESHOLD" | bc -l) )); then
    echo "CPU usage is high: $cpu_usage%" | mail -s "System Alert:"
fi
```

```
0 * * * * /path/to/system_health.sh >> /path/to/logfile.log
2>&1
```

```
LOG_FILE="/path/to/system_health.log"
echo "$(date) - CPU Usage: $cpu_usage%" >> $LOG_FILE
```

This will run the script every hour and log the output to `logfile.log`.

Conclusion

This mini project is a great way to practice your skills with shell scripting and Linux commands. You can expand it as needed to meet your specific use case, like monitoring specific services, adding alerts, or scheduling automated health checks. Enjoy coding!