Technical Tasks – Swapnil Sagar

Set 1: Monitoring System Resources for a Proxy Server

Task Description:

You are required to create a Bash script that monitors various system resources and presents them in a dashboard format. The script should refresh the data every few seconds, providing real-time insights. Additionally, it should allow users to call specific parts of the dashboard individually using command-line switches.

Answer:

To create a comprehensive Bash script that monitors system resources for a proxy server and presents the data in a dashboard format, we'll break down each requirement and write corresponding script sections. The script will use common Linux utilities like top, ps, df, netstat, vmstat, free, systemctl, and awk to gather the required information.

Script Outline

```
#!/bin/bash

# Function to display help message
show_help() {
    echo "Usage: $0 [OPTIONS]"
    echo "Options:"
    echo "  -cpu        Display CPU and load average information"
    echo "  -memory     Display memory usage information"
    echo "  -network    Display network monitoring information"
    echo "  -disk       Display disk usage information"
    echo "  -process    Display process monitoring information"
    echo "  -service    Display service status information"
    echo "  -help       Show this help message"
    exit 0
}

# Function to display top 10 most used applications by CPU and memory
top_applications() {
    echo "Top 10 Applications by CPU and Memory Usage:"
```

```
    ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head -n 11
    echo ""
}

# Function to display network monitoring information
network_monitoring() {
    echo "Network Monitoring:"
    echo "Concurrent connections: $(netstat -tun | grep ESTABLISHED | wc -l)"
    echo "Packet drops (eth0):"
    netstat -i | awk '/eth0/ {print "Dropped packets: "$4}'
    echo "Network traffic (MB in/out):"
    ifconfig eth0 | awk '/RX packets/ {print "RX MB: " $5/1048576} /TX packets/ {print "TX MB: " $5/1048576}'
    echo ""
}

# Function to display disk usage
disk_usage() {
    echo "Disk Usage:"
    df -h | awk '{if($5 >= 80) print $0; else print $0}'
    echo ""
}

# Function to display system load
system_load() {
    echo "System Load:"
    uptime | awk -F 'load average:' '{ print "Current Load Average: " $2 }'
    mpstat | awk '/all/ {printf "CPU Usage: User: %.2f%%, System: %.2f%%, Idle: %.2f%%\n", $4, $6, $12}'
    echo ""
}

# Function to display memory usage
memory_usage() {
    echo "Memory Usage:"
```

```
    free -h | awk 'NR==2{printf "Memory: Total: %s, Used: %s, Free: %s\n", $2,$3,$4}
NR==3{printf "Swap: Total: %s, Used: %s, Free: %s\n", $2,$3,$4}'
    echo ""
}

# Function to display process monitoring
process_monitoring() {
    echo "Process Monitoring:"
    echo "Number of active processes: $(ps aux | wc -l)"
    echo "Top 5 Processes by CPU and Memory Usage:"
    ps aux --sort=-%cpu | head -n 6
    ps aux --sort=-%mem | head -n 6
    echo ""
}

# Function to display service status
service_monitoring() {
    echo "Service Monitoring:"
    for service in sshd nginx apache2 iptables; do
        systemctl is-active --quiet $service && echo "$service is running" || echo "$service is
not running"
    done
    echo ""
}

# Main script logic
if [ "$#" -eq 0 ]; then
    echo "No options provided. Displaying full dashboard."
    top_applications
    network_monitoring
    disk_usage
    system_load
    memory_usage
    process_monitoring
    service_monitoring
    exit 0
```

```
fi

# Parse command-line options
while [ "$1" != "" ]; do
    case $1 in
        -cpu )      shift
                    system_load
                    ;;
        -memory )   shift
                    memory_usage
                    ;;
        -network )  shift
                    network_monitoring
                    ;;
        -disk )     shift
                    disk_usage
                    ;;
        -process )  shift
                    process_monitoring
                    ;;
        -service )  shift
                    service_monitoring
                    ;;
        -help )     shift
                    show_help
                    ;;
        * )         echo "Invalid option: $1"
                    show_help
                    ;;
    esac
    shift
done
```

Explanation of the Script

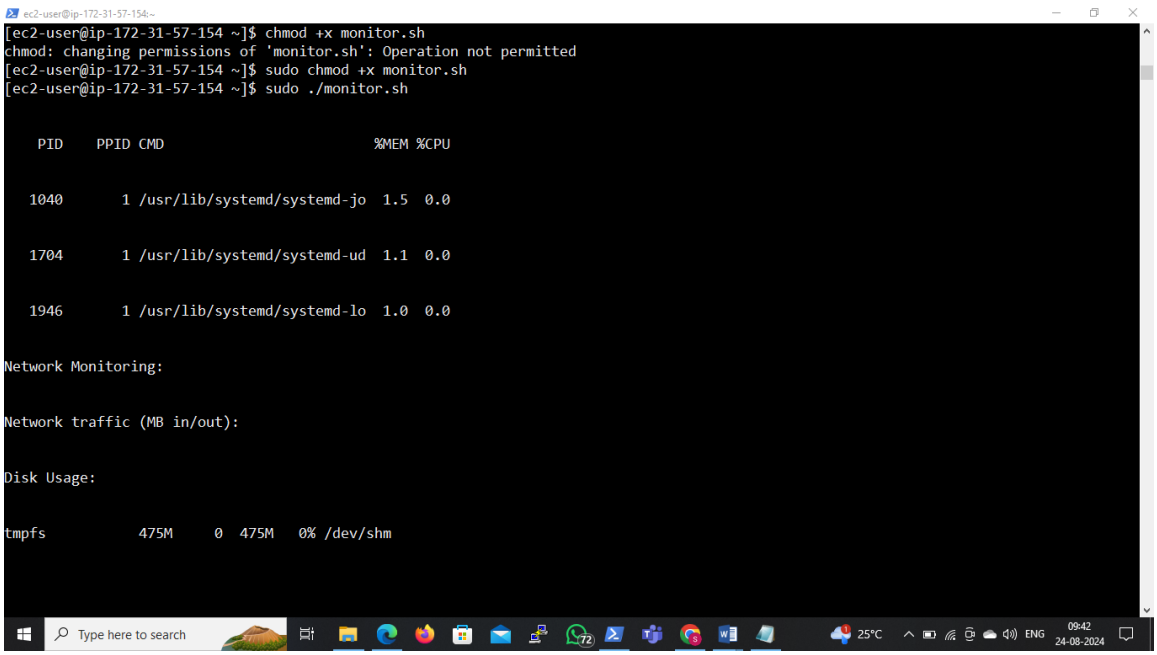
1. **Help Function (`show_help`)**: Provides usage instructions for the script and describes the available command-line options.
2. **Top Applications Function (`top_applications`)**: Uses `ps` to list the top 10 applications by CPU and memory usage.
3. **Network Monitoring Function (`network_monitoring`)**: Displays the number of concurrent connections, packet drops, and network traffic in MB.
4. **Disk Usage Function (`disk_usage`)**: Uses `df` to display disk usage and highlights partitions using more than 80% of space.
5. **System Load Function (`system_load`)**: Shows the current load average and CPU usage breakdown using `uptime` and `mpstat`.
6. **Memory Usage Function (`memory_usage`)**: Displays total, used, and free memory using `free`.
7. **Process Monitoring Function (`process_monitoring`)**: Displays the number of active processes and top 5 processes by CPU and memory usage using `ps`.
8. **Service Monitoring Function (`service_monitoring`)**: Checks the status of essential services (e.g., `sshd`, `nginx`, `iptables`) using `systemctl`.
9. **Main Script Logic**: Checks command-line arguments and calls corresponding functions. If no options are provided, it displays the full dashboard.

Instructions for Usage

1. **Add Execute Permissions (if needed):**

If the script does not have execute permissions (e.g., `-rw-r--r--`), you need to add execute permissions:

chmod +x monitor.sh



2. **Run the script with no options to display the full dashboard:**

   sudo ./monitor.sh

3. **Use specific options to view individual sections:**

   ./monitor.sh -cpu

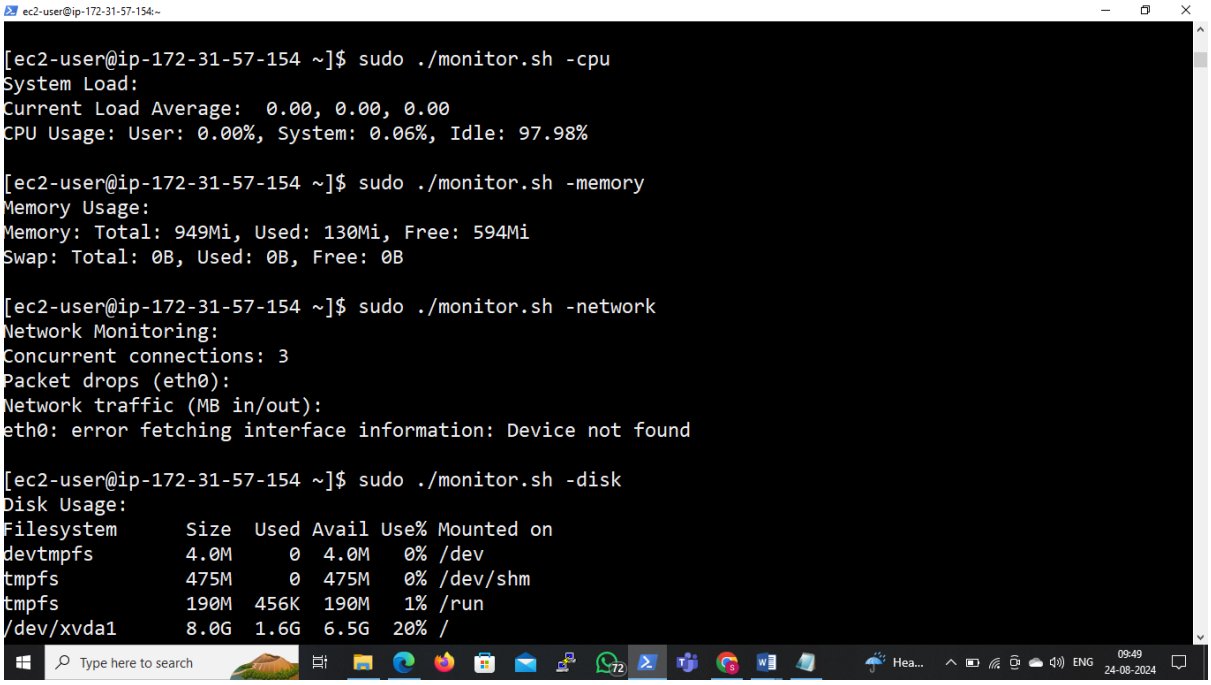
   ./monitor.sh -memory

   ./monitor.sh -network

   ./monitor.sh -disk

   ./monitor.sh -process

   ./monitor.sh -service



4. **View help message:**

   ./monitor.sh -help