Linux Scenario-Based Practice

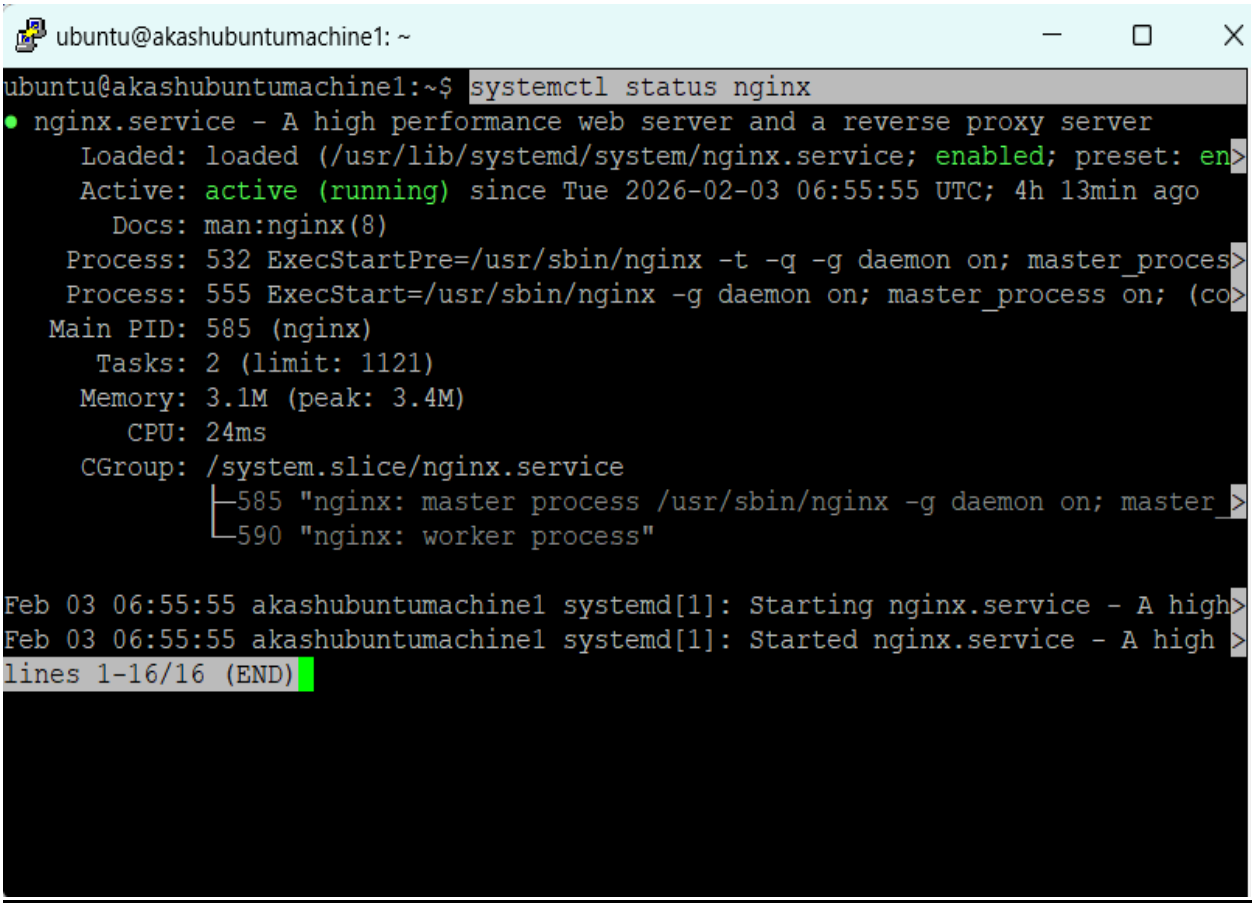
SOLVED EXAMPLE: Understanding How to Approach Scenarios

Example Scenario: Check if a service is running

1-Question: How do you check if the 'nginx' service is running?

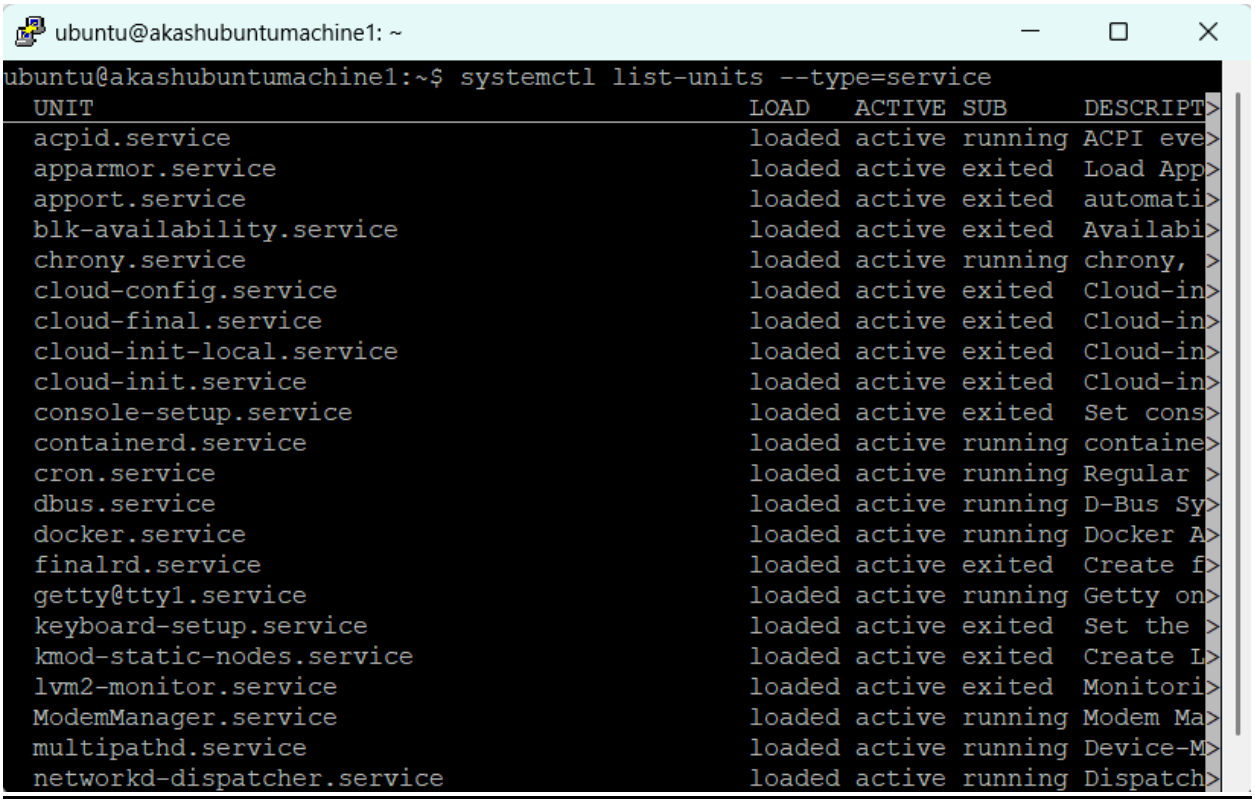
Step 1: Check service status

#systemctl status nginx

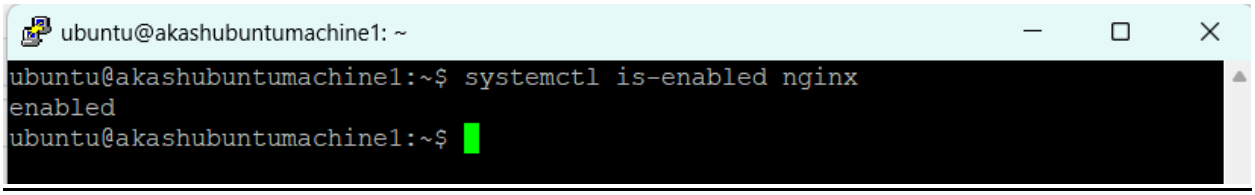


Step 2: If service is not found, list all services

systemctl list-units --type=service



Step 3: Check if service is enabled on boot



Scenario 1: Service Not Starting

A web application service called '**myapp**' failed to start after a server rebootWhat commands would you run to diagnose the issue?

Step 1:
#systemctl status myapp

Why: Check whether the service is running, failed, or inactive after reboot.

Step 2:
#journalctl -u myapp -n 50

Why: View the last 50 log entries for *myapp* to find the exact error message.

Step 3:
#systemctl is-enabled myapp

Why: Verify if the service is enabled to start automatically at boot.

Step 4:
#systemctl restart myapp

Why: Try restarting the service after checking logs to see if the issue persists.

(Optional but professional bonus step):
#ps -ef | grep myapp

Why: Confirm whether the myapp process is actually running at the OS level.

Scenario 2: High CPU Usage

Your manager reports that the application server is slow.You SSH into the server. What commands would you run to identify.

which process is using high CPU?

Step 1:
top

Why: Display live CPU usage and quickly identify which process is consuming the most CPU.

Step 2:
ps aux --sort=-%cpu | head -10

Why: List the top 10 processes sorted by highest CPU usage for clear identification.

Step 3:
htop

Why: Provide an interactive and more readable view of CPU usage and running processes (if installed).

Step 4:
ps -p <PID> -o pid,ppid,cmd,%cpu,%mem

Why: Inspect the specific high-CPU process in detail using its Process ID (PID).

Scenario 3: Finding Service Logs

A developer asks: "Where are the logs for the 'docker' service?"The service is managed by systemd.

What commands would you use?

Step 1:
#systemctl status docker

Why: Shows the current status of Docker and the most recent log messages.

Step 2:
#journalctl -u docker

Why: Display all logs for the docker service from the systemd journal.

Step 3:
#journalctl -u docker -n 50

Why: Show only the last 50 log entries for quick troubleshooting.

Step 4 (live logs):
#journalctl -u docker -f

Why: Follow Docker logs in real time (like tail -f).

Scenario : 4File Permissions Issue

A script at /home/user/backup.sh is not executing.When you run it: ./backup.sh

You get: "Permission denied"

What commands would you use to fix this?

Step 1:
ls -l /home/user/backup.sh

Why: Check the current file permissions to confirm the script is not executable.

Step 2:
#chmod +x /home/user/backup.sh

Why: Add execute permission so the script can be run.

Step 3:
ls -l /home/user/backup.sh

Why: Verify that execute (x) permission is now set.

Step 4:
cd /home/user/backup.sh

Why: Run the script again to confirm the issue is resolved**.**

Why This Matters for DevOps

Understanding the file system is critical for:

- Knowing where to find logs, configs, and binaries
- Troubleshooting deployment issues
- Writing automation scripts that work across systems

Scenario-based practice prepares you for:

- Real production incidents
- DevOps interviews
- On-call troubleshooting under pressure

These are questions you **will** face in interviews and during real incidents.