

Day 4 – Phase 4: Process and Network Monitoring

Boss's Request: Monitor the system while simulating sensor activity.

Tasks:

- Run a background task to simulate sensor polling.

```
GNU nano 7.2 sensor_poll.sh *
#!/bin/bash

while true
do
    # Generate a random sensor value (0-100)
    value=$((RANDOM % 101))
    echo "Sensor reading: $value"

    # Wait 5 seconds before polling again
    sleep 5
done
```

```
salma2002@MSI:~$ nano sensor_poll.sh
salma2002@MSI:~$ chmod +x sensor_poll.sh
salma2002@MSI:~$ ./sensor_poll.sh &
[1] 1613
salma2002@MSI:~$ Sensor reading: 66
Sensor reading: 74
Sensor reading: 93
Sensor reading: 86
```

- List processes and filter for the background task.

```
Sensor reading: 55
Sensor reading: 21
Sensor reading: 27
Sensor reading: 84
ps aux | grep senps aux | grep sensor_poll
salma20+ 1613 0.0 0.0 4752 3168 pts/0 S 18:01 0:00 /bin/bash ./sensor_poll.sh
salma20+ 1633 0.0 0.0 4092 2016 pts/0 S+ 18:02 0:00 grep --color=auto sensor_poll
salma2002@MSI:~$ Sensor reading: 64
salma2002@MSI:~$ Sensor reading: 58
Sensor reading: 56
Sensor reading: 32
```

- Check network states (established connections).

```
Sensor reading: 70
netstat -ant | grep ESTABLISHED
salma2002@MSI:~$ Sensor reading: 72
Sensor reading: 47
```

- Try foreground and background switching.

```
Sensor reading: 30
Sensor reading: 37
fg %1
./sensor_poll.sh
Sensor reading: 78
Sensor reading: 54
Sensor reading: 59
```

- Kill a process if needed.

```
salma2002@MSI:~$ kill -9 1613
salma2002@MSI:~$ jobs -l
[1]+  1613 Killed                  ./sensor_poll.sh
salma2002@MSI:~$
```

Open-Ended Questions:

- What happens step by step when you type a command in bash (e.g., ls) until you see the output?
- Explain the types of processes in Linux: daemon, zombie, orphan. How can you detect them?
- Why do we need Inter-Process Communication (IPC)? List some IPC mechanisms and real-life examples.