eh1act06 – System resource monitoring. Benchmarking and stressing the system

GENERAL CONDITIONS

1- Deadline: **09-05-2025**
2- Teacher will check that your operating system is working properly

DOCUMENTATION

1- Basic ideas

a) What is a system resource?
Any component of a computer system that can be used by the operating system and applications to perform tasks. These resources are finite and must be managed efficiently to keep the system running properly.

b) What are the resources typically monitored?
- CPU: Usage % and load average
- RAM: total, used and free
- Swap: total, used and free
- Disk space: total, used, free, read and write speed
- Network: Bandwith usage
- Processes: Top consumers of CPU and RAM
- System Uptime: How long the system has been running

c) What means "system resource monitoring"?
System resource monitoring means observing, measuring, and analyzing how a computer system uses its resources (like CPU, memory, disk, network, etc.) in real time or over a period of time.

d) What is the porpouse of system resource monitoring?
- Detect performance issues (e.g., high CPU usage).
- Prevent system crashes (e.g., from full memory or disk).
- Identify bottlenecks that slow down applications.
- Optimize resource usage to improve efficiency.

e) Command-Line tools for monitoring system resources
- **top** and **htop** : Live view of CPU, memory, and process usage.
- **free -h** : Shows used and available RAM and swap space in human-readable format.
- **df -th** : Reports disk space usage of mounted filesystems.
- **vnstat** : Real-time bandwidth usage by connection
- **hwinfo** : It is a command-line utility that collects and prints information about the system hardware.

f) What means "benchmarking a system"?
Benchmarking a system means measuring its performance under specific workloads or tests, to evaluate how well the system performs in terms of: CPU, RAM, network bandwidth, disk performance, etc...

f) What means "streassing the system"?
Stressing a system means deliberately putting high load on system resources like CPU, memory, disk, or network to test system stability under pressure, benchmark performance, simulate high usage scenarios or identify hardware/software bottlenecks or weaknesses.

g) Command-Line tools to stress and benchmark Linux systems
- **stress** : → command-line tool to stress a Linux system.
- **sysbench** → command-line tool to test CPU, memory and storage perfomance.

2- Installing system resource monitoring software on your system

- **top**, **uptime**, **free** and **df** were installed during the Linux installation process.
- **sudo aptitude -y install htop** → this command installs **htop** on your system.
- **sudo aptitude -y install hwinfo** → this command installs **hwinfo** on your system.
- **sudo aptitude -y install vnstat** → this command installs **vnstat** on your system.
- **sudo aptitude -y install sysbench** → this command installs **sysbench** on your system.
- **sudo aptitude -y install stress** → this command installs **stress** on your system.

PRACTICAL EXERCISE

Part 1 – System resource monitoring

1- Install **htop**, **hwinfo**, **vnstat**, **stress** and **sysbench** on your system.

2- With the help of **hwinfo** show:
- Capacity of your hard drive
- CPU manufacturer and model
- Network card manufacturer, model and bandwidth.

3- With the help of **free** show:
- Total and available physical memory installed on your machine that can be used by the operating system and applications.
- Extra available memory via hard drive SWAP space.

4- With the help of **htop** show:
- Tasks running on your system (running or sleeping)
- % of CPU usage for cores 0 and 1

5- With the help of **df** show total, used and available disk space of your linux system partition in human-readable format.

6- With the help of **vnstat** capture network traffic on **enp0s3** while you are surfing on internet with the help of your web browser. Afterwards, show:
- Total amount of received and transmitted in KiB.
- Average receiving and transmitting speed in Kbits/s.
- Peak receiving and transmitting speed in Kbits/s.

Part 2 – Benchmarking the system

1- With the help of **sysbench**, test performance of your CPU. Check events per second and latency (min, avg, max). Afterwards, compare your results with the following results:

```
CPU speed:
    events per second:  1398.62

General statistics:
    total time:                          10.0001s
    total number of events:              13988

Latency (ms):
         min:                                    0.70
         avg:                                    0.71
         max:                                    3.29
         95th percentile:                        0.75
         sum:                                 9997.24

Threads fairness:
    events (avg/stddev):           13988.0000/0.00
    execution time (avg/stddev):   9.9972/0.00
```

Is your virtual machine's CPU performance better o worse than the computer where these results were taken?. Why?.

2- With the help of **sysbench**, test performance of your memory. Check **Total operations** and l**atency** (min, avg, max). Afterwards, compare your results with the following results:

```
Total operations: 78784605 (7877331.85 per second)

76938.09 MiB transferred (7692.71 MiB/sec)


General statistics:
    total time:                          10.0001s
    total number of events:              78784605

Latency (ms):
         min:                                    0.00
         avg:                                    0.00
         max:                                    1.01
         95th percentile:                        0.00
         sum:                                 4658.47

Threads fairness:
    events (avg/stddev):           78784605.0000/0.00
    execution time (avg/stddev):   4.6585/0.00
```

Is your virtual machine's Memory performance better o worse than the computer where these results were taken?. Why?.

Part 3 – Stressing the system

1- With the help of **stress**, run a test to stress CPUs of your virtual machine running t**wo stressing processes** for **60 second**s. Open another terminal and run **htop** while the test is running. Check CPU usage for core 0 and 1. What happens?.

2- With the help of **stress**, run a test to stress the memory of your virtual machine running **two stressing processes** for **60 seconds**. Open another terminal and run **htop** while the test is running. Check the total memory usage. What happens?.

Part 4 - Checking your practical exercise

a) One random question among of part **1**. One single opportunity to properly answering the question.

b) One random question of part **2** with a small difference in the command you have to run. One single opportunity to properly answering the question.

c) One random question among of part **3** with a small difference in the command you have to run.. One single opportunity to properly answering the question.