

Informe Laboratorio 2

1. Análisis de memoria física y virtual (swap).

La **memoria virtual** es una técnica que usan los sistemas operativos para simular más memoria RAM de la que realmente tiene una computadora, utilizando una parte del disco duro como si fuera memoria principal.

El procedimiento seguido fue abrir dos terminales, en la primera para observar el htop con los procesos y en la segunda ejecutar el script "llena_memoria.py"

The screenshot shows a Linux desktop environment with three terminal windows. The top-left terminal displays the htop process monitor, showing system statistics and a list of running processes. The top-right terminal shows the execution of the 'llena_memoria.py' script, which is filling memory. The bottom terminal shows the system's memory usage increasing over time.

Memoria en reposo, porcentajes

Memoria consumida: ~800 MB
Memoria consumida: ~900 MB
Memoria consumida: ~1000 MB
Memoria consumida: ~1100 MB
Memoria consumida: ~1200 MB
Memoria consumida: ~1300 MB
Memoria consumida: ~1400 MB
TerMemoria consumida: ~1500 MB

Momento en el que se empieza a usar el swap

```
Memoria consumida: ~5400 MB
Memoria consumida: ~5500 MB
Memoria consumida: ~5600 MB
Memoria consumida: ~5700 MB
Memoria consumida: ~5800 MB
Memoria consumida: ~5900 MB
Memoria consumida: ~6000 MB
TerMemoria consumida: ~6100 MB

0[|||||] 19.2% Tasks: 122, 589 thr, 97 kthr; 2 runnin
1[|||||] 53.0% Load average: 1.40 1.09 0.60
2[|||||] 26.1% Uptime: 00:17:02
3[|||||] 9.8%

Mem[|||||] 3.52G/3.82G
Swp[|||||] 3.67G/3.82G

Main I/O
PID USER PRI NI VIRT RES SHR S CPU%MEM% TIME+ Command
4485 root 20 0 2021M 18448 9992 S 36.4 0.5 0:01.38 /usr/lib/snap
4100 root 20 0 2021M 18448 9992 S 10.7 0.5 0:01.62 /usr/lib/snap
4812 andres 20 0 6051M 3151M 2700 S 7.1 80.5 0:09.91 python3 llena
4548 andres 20 0 20424 3516 1852 R 4.3 0.1 0:18.54 htop
1964 andres 20 0 4850M 86996 28020 S 2.9 2.2 0:07.77 /usr/bin/gnom
1962 andres 20 0 4850M 86996 28020 S 1.4 2.2 0:08.17 /usr/bin/gnom
1963 andres 20 0 4850M 86996 28020 S 1.4 2.2 0:07.70 /usr/bin/gnom
1965 andres 20 0 4850M 86996 28020 S 1.4 2.2 0:07.80 /usr/bin/gnom
1926 andres 20 0 4850M 86996 28020 S 0.7 2.2 0:26.66 /usr/bin/gnom
3508 andres 20 0 2688M 52852 23180 S 0.7 1.3 0:11.64 /snap/firefox
3530 andres 20 0 2688M 52852 23180 S 0.7 1.3 0:00.18 /snap/firefox
4095 root 20 0 2021M 18448 9992 S 0.7 0.5 0:03.49 /usr/lib/snap
4096 root 20 0 2021M 18448 9992 S 0.7 0.5 0:01.15 /usr/lib/snap
```

Memoria y swap llenos

```
Memoria consumida: ~5700 MB
Memoria consumida: ~5800 MB
Memoria consumida: ~5900 MB
Memoria consumida: ~6000 MB
Memoria consumida: ~6100 MB
Memoria consumida: ~6200 MB
Memoria consumida: ~6300 MB
TerTerminado (killed)
andres@andres-VirtualBox: ~/Escritorio$

0[ ] 0.0% Tasks: 121, 590 thr, 93 kthr; 1 runnin
1[ ] 0.0% Load average: 0.25 0.76 0.56
2[ ] 2.7% Uptime: 00:20:12
3[ ] 0.0%

Mem[ ] 437M/3.82G
Swp[ ] 901M/3.82G

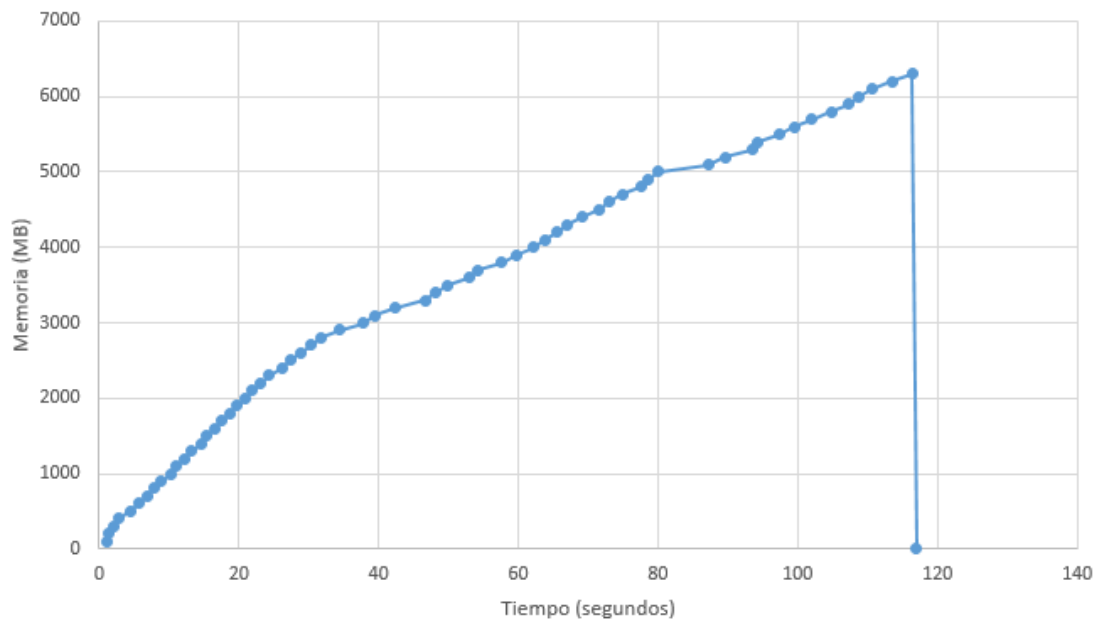
Main I/O
PID USER PRI NI VIRT RES SHR S CPU%MEM% TIME+ Command
4548 andres 20 0 20424 3244 1580 R 5.2 0.1 0:24.74 htop
1973 andres 20 0 4850M 71612 21596 S 1.3 1.8 0:02.49 /usr/bin/gnom
1926 andres 20 0 4850M 71612 21596 S 0.7 1.8 0:27.27 /usr/bin/gnom
2860 andres 20 0 10.8G 79988 45644 S 0.7 2.0 0:00.50 /snap/firefox
3508 andres 20 0 2688M 70308 37136 S 0.7 1.8 0:12.03 /snap/firefox
4521 andres 20 0 556M 13888 9524 S 0.7 0.3 0:04.35 /usr/libexec/
1 root 20 0 23064 7124 4308 S 0.0 0.2 0:02.15 /sbin/init sp
265 root 19 -1 50888 10048 9408 S 0.0 0.3 0:00.52 /usr/lib/syst
333 root 20 0 30532 1336 1208 S 0.0 0.0 0:00.25 /usr/lib/syst
432 systemd-oo 20 0 17556 2536 2152 S 0.0 0.1 0:01.13 /usr/lib/syst
439 systemd-re 20 0 21708 5424 3760 S 0.0 0.1 0:00.21 /usr/lib/syst
441 systemd-ti 20 0 91044 2504 2376 S 0.0 0.1 0:00.05 /usr/lib/syst
786 systemd-ti 20 0 91044 2504 2376 S 0.0 0.1 0:00.00 /usr/lib/syst
```

programa terminado de manera abrupta

Cuando el sistema se quedó sin memoria física y swap, no falló. En su lugar, el kernel de Linux activó su mecanismo de protección 'Out-of-Memory (OOM) Killer', que identificó a nuestro script como el mayor consumidor de memoria y lo terminó forzosamente. Esto se evidencia con el mensaje 'Terminado (killed)' y es una estrategia para asegurar la estabilidad del sistema.

Memoria vs Tiempo

Memoria en función del tiempo



La línea final representa el (OOM) killer de Linux.

OOM Killer (Out-of-Memory Killer). Cuando el sistema se queda sin memoria RAM y sin swap, en lugar de colapsar por completo, el kernel de Linux elige un proceso (generalmente el que más memoria consume) y lo "mata" para liberar memoria y mantener el sistema funcionando.

2. Impacto de la Caché de Disco en el Rendimiento

La **caché de página (page cache)** en Linux es una parte de la memoria RAM que el SO utiliza para almacenar temporalmente el contenido de archivos y datos del disco que han sido leídos o escritos recientemente.

Para el experimento, primeramente limpiamos la caché del SO, luego ejecutamos nuestro archivos y lo ejecutamos por segunda vez para observar la diferencia de tiempo.

```
andres@andres-VirtualBox: ~/Escritorio
andres@andres-VirtualBox:~/Escritorio$ sync && sudo sh -c 'echo 3 > /proc/sys/vm/drop_caches'
andres@andres-VirtualBox:~/Escritorio$ time cat archivo_prueba > /dev/null
real    0m0,998s
user    0m0,007s
sys     0m0,763s
andres@andres-VirtualBox:~/Escritorio$ time cat archivo_prueba > /dev/null
real    0m0,045s
user    0m0,004s
sys     0m0,041s
andres@andres-VirtualBox:~/Escritorio$
```

comparacion sin caché, con caché

La diferencia de tiempo entre la lectura en frío y la lectura en caliente es notable. La primera lectura fue lenta porque el SO tuvo que acceder al dispositivo de almacenamiento físico (disco duro). Durante esta lectura, el kernel copió el contenido del archivo en la RAM (caché de página). La segunda lectura fue casi instantánea porque el SO sirvió el archivo directamente desde esta caché en la RAM, evitando por completo el acceso al disco.