



UNIVERSIDAD TECNOLÓGICA DE PANAMÁ
FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES
DEPARTAMENTO DE ARQUITECTURA Y REDES DE COMPUTADORAS



GUÍA DE LABORATORIO N°.4

Facilitadora: Prof. Ahlam Almaeni.

Asignatura: Sistemas Operativos I

Estudiantes:

Joy Nelaton 8-902-1282

Daniel Barton 8-961-138

Daniel Downs 3-746-1308 Fecha: 12 de Junio del 2023 Grupo: IIL143

A. TÍTULO DE LA EXPERIENCIA: Comandos para administrar y gestionar fácilmente los procesos en Linux.

B. TEMAS:

✚ Administración y gestión de procesos en Linux.

C. OBJETIVO:

✚ Conocer y utilizar los comandos más importantes para la gestión de procesos en Linux.

D. RECURSOS:

✚ Un computador personal.

✚ SO Linux Fedora.

✚ Guía del laboratorio.

E. RÚBRICAS:

Criterios	2	1	0
I – Identificación del/los participante/s	100%	El 50%	Ninguno
II - Proceso – Utilizó los recursos recomendados en el enunciado o procedimiento.	100%	Más del 50%	Menos del 50%
III - Solución – Presentó los datos solicitados	100%	Entre 50% y 70%.	Menos del 50%
IV – Puntualidad en la entrega	100%	Entregó después de la fecha	No entregó
V - Formato – Siguió el formato presentado.	100%	Obvió algunos puntos	No siguió el formato

F. ENUNCIADO DE LA EXPERIENCIA O PROCEDIMIENTO:

Metodología:

✚ Desarrolle el laboratorio en **grupo de 4**.

✚ Lea detenidamente la guía completa antes de iniciarla.

✚ Desarrolle los pasos indicados en la “explicación y ejecución de los comandos”, capture la pantalla y reemplace las que aparecen en la guía.



GUÍA DE LABORATORIO N°.4

Introducción:

Es conocida la extensión que alcanza **Linux** en lo que a servidores en producción se refiere. Los **bajos consumos de recursos** y lo fácil que ha llegado a ser administrarlos desde hace unos años han ayudado a esto.

Los **procesos** juegan un papel muy importante en las distribuciones Linux, ya que son los que consumirán estos recursos hardware tan preciados en entornos de producción, **administrarlos y gestionarlos correctamente** es de vital importancia ya que estos procesos y la gestión que hace el sistema sobre ellos hacen posible mantener funcionando el servidor sin necesidad de reiniciar después de un cambio o actualización importante. Esto es uno de los puntos más importantes por los que **Linux gobierna el 90% de los servidores alrededor del mundo**.

Para esta labor contamos con **varias herramientas** a nuestra disposición, veamos algunas de ellas.

Para ver los procesos en sistemas Linux, contamos con el comando '**ps**', que listará (de múltiples formas según las opciones que le pasemos) todos los procesos que se encuentran corriendo en nuestro equipo.

Explicación y ejecución de los comandos

ps [opciones]

```
ubuntu@ubuntu18: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
ubuntu@ubuntu18:~$ ps  
  PID TTY          TIME CMD  
 1616 pts/0    00:00:00 bash  
 1742 pts/0    00:00:00 ps
```



GUÍA DE LABORATORIO N°.4

Como de costumbre, podemos **revisar el manual de ps** dentro del sistema para conocer todas las opciones posibles:

man ps

```
ubuntu@ubuntu18: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
default selection is discarded, and then the selected processes are  
added to the set of processes to be displayed. A process will thus be  
shown if it meets any of the given selection criteria.  
  
EXAMPLES  
To see every process on the system using standard syntax:  
ps -e  
ps -ef  
ps -eF  
ps -ely  
  
To see every process on the system using BSD syntax:  
ps ax  
ps axu  
  
To print a process tree:  
ps -ejH  
ps axjf  
  
To get info about threads:  
ps -eLf  
ps axms  
  
Manual page ps(1) line 52 (press h for help or q to quit)
```

GUÍA DE LABORATORIO N°.4

Siendo las más habituales:

ps aux (muestra todos los procesos del sistema)

```
ubuntu@ubuntu18: ~
Archivo Editar Ver Buscar Terminal Ayuda
ubuntu@ubuntu18:~$ ps aux
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.4  0.2 159916 9072 ?        Ss   16:57   0:01 /sbin/init splash
root           2  0.0  0.0      0     0 ?        S    16:57   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        I<   16:57   0:00 [rcu_gp]
root           4  0.0  0.0      0     0 ?        I<   16:57   0:00 [rcu_par_gp]
root           6  0.0  0.0      0     0 ?        I<   16:57   0:00 [kworker/0:0H-k
root           7  0.0  0.0      0     0 ?        I    16:57   0:00 [kworker/u4:0-e
root           8  0.0  0.0      0     0 ?        I<   16:57   0:00 [mm_percpu_wq]
root           9  0.0  0.0      0     0 ?        S    16:57   0:00 [ksoftirqd/0]
root          10  0.0  0.0      0     0 ?        I    16:57   0:00 [rcu_sched]
root          11  0.0  0.0      0     0 ?        S    16:57   0:00 [migration/0]
root          12  0.0  0.0      0     0 ?        S    16:57   0:00 [idle_inject/0]
root          13  0.0  0.0      0     0 ?        I    16:57   0:00 [kworker/0:1-cg
root          14  0.0  0.0      0     0 ?        S    16:57   0:00 [cpuhp/0]
root          15  0.0  0.0      0     0 ?        S    16:57   0:00 [cpuhp/1]
root          16  0.0  0.0      0     0 ?        S    16:57   0:00 [idle_inject/1]
root          17  0.0  0.0      0     0 ?        S    16:57   0:00 [migration/1]
root          18  0.0  0.0      0     0 ?        S    16:57   0:00 [ksoftirqd/1]
root          19  0.0  0.0      0     0 ?        I    16:57   0:00 [kworker/1:0-ev
root          20  0.0  0.0      0     0 ?        I<   16:57   0:00 [kworker/1:0H-k
root          21  0.0  0.0      0     0 ?        S    16:57   0:00 [kdevtmpfs]
root          22  0.0  0.0      0     0 ?        I<   16:57   0:00 [netns]
root          23  0.0  0.0      0     0 ?        S    16:57   0:00 [rcu_tasks_kthr
root          24  0.0  0.0      0     0 ?        S    16:57   0:00 [kauditd]
root          25  0.0  0.0      0     0 ?        S    16:57   0:00 [khungtaskd]
root          26  0.0  0.0      0     0 ?        S    16:57   0:00 [oom_reaper]
root          27  0.0  0.0      0     0 ?        I<   16:57   0:00 [writeback]
root          28  0.0  0.0      0     0 ?        S    16:57   0:00 [kcompactd0]
root          29  0.0  0.0      0     0 ?        SN   16:57   0:00 [ksmd]
root          30  0.0  0.0      0     0 ?        SN   16:57   0:00 [khugepaged]
root          34  0.0  0.0      0     0 ?        I    16:57   0:00 [kworker/1:1-ev
root          77  0.0  0.0      0     0 ?        I<   16:57   0:00 [kintegrityd]
root          78  0.0  0.0      0     0 ?        I<   16:57   0:00 [kblockd]
root          79  0.0  0.0      0     0 ?        I<   16:57   0:00 [blkcg_punt_bio
```

GUÍA DE LABORATORIO N°.4

ps axjf (que mostrará un árbol jerárquico con la ruta del programa al que pertenece el proceso)

Las opciones que podemos aplicar a ps no van más allá de mostrar la información de una u otra forma, más o menos extensa, o como ya sabemos, filtrar los resultados con grep. Sea cual sea el método de muestra que elijamos, siempre habrá dos constantes, el PID y el comando o nombre del programa. Aquí un ejemplo de filtrado sobre ps para obtener únicamente los procesos pertenecientes a bash.

```

ubuntu@ubuntu18: ~
Archivo Editar Ver Buscar Terminal Ayuda
ubuntu@ubuntu18:~$ ps axjf
PPID    PID    PGID    SID    TTY          TPGID STAT   UID     TIME COMMAND
0        2      0       0      ?            -1 S      0       0:00 [kthreadd]
2        3      0       0      ?            -1 I<     0       0:00 \_ [rcu_gp]
2        4      0       0      ?            -1 I<     0       0:00 \_ [rcu_par_gp]
2        6      0       0      ?            -1 I<     0       0:00 \_ [kworker/0:0H-kb]
2        7      0       0      ?            -1 I      0       0:00 \_ [kworker/u4:0-ev]
2        8      0       0      ?            -1 I<     0       0:00 \_ [mm_percpu_wq]
2        9      0       0      ?            -1 S      0       0:00 \_ [ksoftirqd/0]
2       10      0       0      ?            -1 I      0       0:00 \_ [rcu_sched]
2       11      0       0      ?            -1 S      0       0:00 \_ [migration/0]
2       12      0       0      ?            -1 S      0       0:00 \_ [idle_inject/0]
2       13      0       0      ?            -1 I      0       0:00 \_ [kworker/0:1-cgr]
2       14      0       0      ?            -1 S      0       0:00 \_ [cpuhp/0]
2       15      0       0      ?            -1 S      0       0:00 \_ [cpuhp/1]
2       16      0       0      ?            -1 S      0       0:00 \_ [idle_inject/1]
2       17      0       0      ?            -1 S      0       0:00 \_ [migration/1]
2       18      0       0      ?            -1 S      0       0:00 \_ [ksoftirqd/1]
2       19      0       0      ?            -1 I      0       0:00 \_ [kworker/1:0-eve]
2       20      0       0      ?            -1 I<     0       0:00 \_ [kworker/1:0H-kb]
2       21      0       0      ?            -1 S      0       0:00 \_ [kdevtmpfs]
2       22      0       0      ?            -1 I<     0       0:00 \_ [netns]
2       23      0       0      ?            -1 S      0       0:00 \_ [rcu_tasks_kthre]
2       24      0       0      ?            -1 S      0       0:00 \_ [kauditd]
2       25      0       0      ?            -1 S      0       0:00 \_ [khungtaskd]
2       26      0       0      ?            -1 S      0       0:00 \_ [oom_reaper]
2       27      0       0      ?            -1 I<     0       0:00 \_ [writeback]

```



UNIVERSIDAD TECNOLÓGICA DE PANAMÁ
FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES
DEPARTAMENTO DE ARQUITECTURA Y REDES DE COMPUTADORAS



GUÍA DE LABORATORIO N°.4

ps aux | grep bash

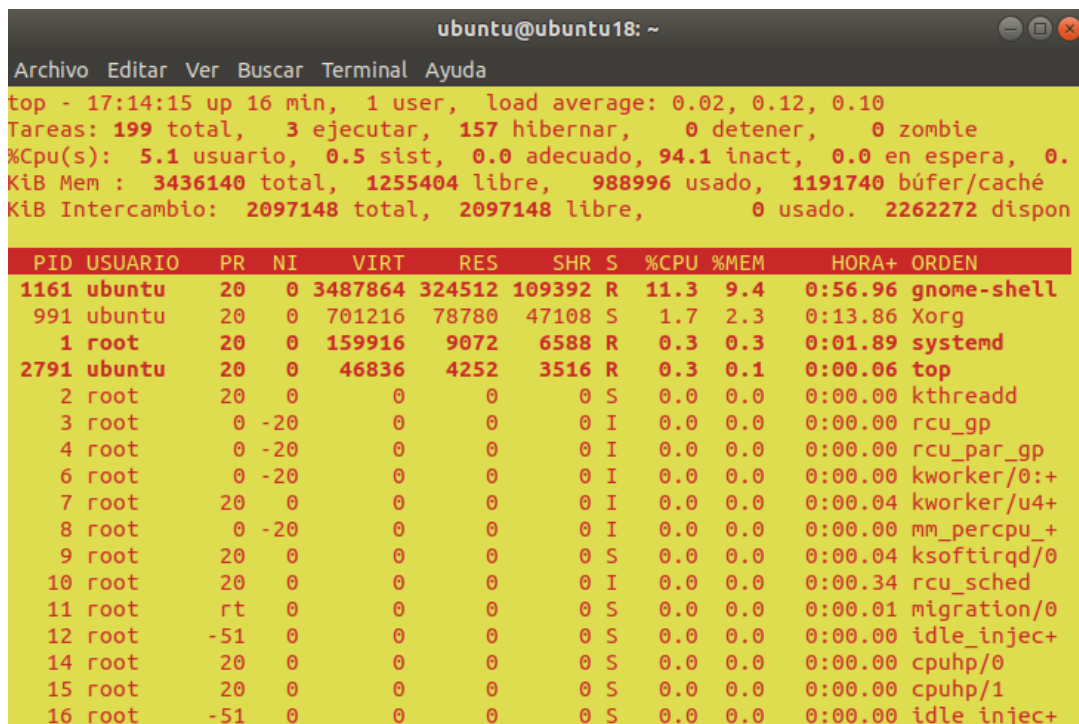
```
userlub@userlub-pc:~$ ps aux | grep bash
userlub      4196  0.0  0.0  13928  5496 pts/2    Ss   07:41   0:00  bash
userlub      8673  0.0  0.0  11740  2332 pts/2    S+   08:32   0:00  grep --color=auto bash
```

El **PID** es el **número identificador de proceso** que le asigna el sistema a cada proceso que se inicia, mientras que el **command** es el programa al cual pertenece dicho proceso.

GUÍA DE LABORATORIO N°.4

Top es otro gestor de procesos integrado en la mayoría de los sistemas Linux. Mientras que **ps** nos muestra un listado de procesos estático, es decir, nos informa de los procesos, nombres, usuarios o recursos que se están usando en el momento de la petición; **top** nos da un informe en tiempo real de los mismos.

top



```

ubuntu@ubuntu18: ~
Archivo Editar Ver Buscar Terminal Ayuda
top - 17:14:15 up 16 min,  1 user,  load average: 0.02, 0.12, 0.10
Tareas: 199 total,  3 ejecutar, 157 hibernar,  0 detener,  0 zombie
%Cpu(s):  5.1 usuario,  0.5 sist,  0.0 adecuado, 94.1 inact,  0.0 en espera,  0.
KiB Mem : 3436140 total, 1255404 libre,  988996 usado, 1191740 búfer/caché
KiB Intercambio: 2097148 total, 2097148 libre,      0 usado. 2262272 dispon

```

PID	USUARIO	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	HORA+	ORDEN
1161	ubuntu	20	0	3487864	324512	109392	R	11.3	9.4	0:56.96	gnome-shell
991	ubuntu	20	0	701216	78780	47108	S	1.7	2.3	0:13.86	Xorg
1	root	20	0	159916	9072	6588	R	0.3	0.3	0:01.89	systemd
2791	ubuntu	20	0	46836	4252	3516	R	0.3	0.1	0:00.06	top
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:+
7	root	20	0	0	0	0	I	0.0	0.0	0:00.04	kworker/u4+
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_+
9	root	20	0	0	0	0	S	0.0	0.0	0:00.04	ksoftirqd/0
10	root	20	0	0	0	0	I	0.0	0.0	0:00.34	rcu_sched
11	root	rt	0	0	0	0	S	0.0	0.0	0:00.01	migration/0
12	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_injec+
14	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
15	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/1
16	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_injec+



GUÍA DE LABORATORIO N°.4

man top

```
ubuntu@ubuntu18: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
(SHR) by the kernel.  
  
1. COMMAND-LINE Options  
The command-line syntax for top consists of:  
  
-hv|-bcHiOSs -d secs -n max -u|U user -p pid -o fld -w [cols]  
  
The typically mandatory switch ('-') and even whitespace are completely optional.  
  
-h | -v :Help/Version  
Show library version and the usage prompt, then quit.  
  
-b :Batch-mode operation  
Starts top in Batch mode, which could be useful for sending output from top to other programs or to a file. In this mode, top will not accept input and runs until the iterations limit you've set with the '-n' command-line option or until killed.  
  
-c :Command-line/Program-name toggle  
Starts top with the last remembered 'c' state reversed. Thus, if top was displaying command lines, now that field will show program names, and vice versa. See the 'c' interactive command for additional information.  
  
-d :Delay-time interval as: -d ss.t (secs.tenths)  
Specifies the delay between screen updates, and overrides the corresponding value in one's personal configuration file or the startup default. Later this can be changed with the 'd' or 's' interactive commands.
```


GUÍA DE LABORATORIO N°.4

Aquí, como vemos en su manual, podemos controlar más aspectos, como los de los siguientes ejemplos entre otros:

top -d 5 (Donde 5 es el número de segundos a transcurrir entre cada muestreo)

top -o %CPU (Donde %CPU es el valor por el que vamos a **ordenar los procesos**)

```

ubuntu@ubuntu18: ~
Archivo Editar Ver Buscar Terminal Ayuda
top - 17:18:55 up 21 min,  1 user,  load average: 0.08, 0.13, 0.10
Tareas: 206 total,  1 ejecutar, 159 hibernar,   0 detener,   0 zombie
%Cpu(s): 11.0 usuario,  0.4 sist,  0.0 adecuado, 88.5 inact,  0.0 en espera,  0.0 hardw int,  0.1 softw
KiB Mem : 3436140 total, 1241640 libre,  996976 usado, 1197524 búfer/caché
KiB Intercambio: 2097148 total, 2097148 libre,   0 usado. 2251984 dispon Mem

  PID USUARIO    PR  NI   VIRT   RES   SHR S  %CPU %MEM    HORA+ ORDEN
1161 ubuntu      20   0 3493704 331216 110848 S   23.0  9.6   1:19.06 gnome-shell
 991 ubuntu      20   0 705160  82664  48392 S    1.8  2.4   0:19.20 Xorg
    1 root         20   0 159916   9072   6588 S    0.4  0.3   0:02.11 systemd
3252 ubuntu      20   0 46872  4176  3456 R    0.4  0.1   0:00.08 top
 338 root        -51   0      0      0      0 S    0.2  0.0   0:00.18 irq/18-vmwgfx
 797 root         20   0 299252   9964   6860 S    0.2  0.3   0:00.16 polkitd
1533 ubuntu      20   0 1335156 165640 37220 S    0.2  4.8   0:05.58 gnome-software
3018 root         20   0      0      0      0 I    0.2  0.0   0:00.02 kworker/u4:1-ev
    2 root         20   0      0      0      0 S    0.0  0.0   0:00.00 kthreadd
    3 root          0 -20      0      0      0 I    0.0  0.0   0:00.00 rcu_gp
    4 root          0 -20      0      0      0 I    0.0  0.0   0:00.00 rcu_par_gp
    6 root          0 -20      0      0      0 I    0.0  0.0   0:00.00 kworker/0:0H-kb
    7 root         20   0      0      0      0 I    0.0  0.0   0:00.06 kworker/u4:0-ne
    8 root          0 -20      0      0      0 I    0.0  0.0   0:00.00 mm_percpu_wq
    9 root         20   0      0      0      0 S    0.0  0.0   0:00.04 ksoftirqd/0
   10 root         20   0      0      0      0 I    0.0  0.0   0:00.43 rcu_sched
   11 root         rt    0      0      0      0 S    0.0  0.0   0:00.01 migration/0
  
```

GUÍA DE LABORATORIO N°.4

```
ubuntu@ubuntu18: ~
Archivo Editar Ver Buscar Terminal Ayuda
ubuntu@ubuntu18:~$ top -o %CPU

top - 17:23:43 up 26 min, 1 user, load average: 0.06, 0.08, 0.08
Tareas: 202 total, 2 ejecutar, 159 hibernar, 0 detener, 0 zombie
%Cpu(s): 2.3 usuario, 0.3 sist, 0.0 adecuado, 97.3 inact, 0.0 en espera, 0.
KiB Mem : 3436140 total, 1240488 libre, 997100 usado, 1198552 búfer/caché
KiB Intercambio: 2097148 total, 2097148 libre, 0 usado. 2253140 dispon

  PID USUARIO PR NI VIRT RES SHR S %CPU %MEM HORA+ ORDEN
1161 ubuntu 20 0 3493304 329824 109504 S 5.3 9.6 1:26.14 gnome-shell
3347 ubuntu 20 0 46836 4272 3532 R 0.7 0.1 0:00.19 top
1519 root 20 0 0 0 0 I 0.3 0.0 0:00.31 kworker/1:++
1 root 20 0 159916 9072 6588 S 0.0 0.3 0:02.21 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kthreadd
3 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 rcu_gp
4 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 rcu_par_gp
6 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/0:++
7 root 20 0 0 0 0 I 0.0 0.0 0:00.07 kworker/u4+
8 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 mm_percpu_+
9 root 20 0 0 0 0 S 0.0 0.0 0:00.05 ksoftirqd/0
10 root 20 0 0 0 0 R 0.0 0.0 0:00.48 rcu_sched
11 root rt 0 0 0 0 S 0.0 0.0 0:00.01 migration/0
12 root -51 0 0 0 0 S 0.0 0.0 0:00.00 idle_injec+
14 root 20 0 0 0 0 S 0.0 0.0 0:00.00 cpuhp/0
```

`top -u ubuntu` (Donde **ubuntu** es el usuario del cual queremos mostrar los procesos).

```
ubuntu@ubuntu18: ~
Archivo Editar Ver Buscar Terminal Ayuda
top - 17:26:13 up 28 min, 1 user, load average: 0.18, 0.10, 0.09
Tareas: 203 total, 2 ejecutar, 157 hibernar, 0 detener, 0 zombie
%Cpu(s): 10.0 usuario, 1.3 sist, 0.0 adecuado, 88.7 inact, 0.0 en espera, 0.0 hardw int, 0.0 softw
KiB Mem : 3436140 total, 1247596 libre, 987844 usado, 1200700 búfer/caché
KiB Intercambio: 2097148 total, 2097148 libre, 0 usado. 2260940 dispon Mem

  PID USUARIO PR NI VIRT RES SHR S %CPU %MEM HORA+ ORDEN
1161 ubuntu 20 0 3496696 333380 110848 S 18.3 9.7 1:38.33 gnome-shell
991 ubuntu 20 0 705160 82660 48388 S 1.6 2.4 0:25.04 Xorg
3455 ubuntu 20 0 46840 4236 3488 R 0.7 0.1 0:00.34 top
940 ubuntu 20 0 77024 8172 6712 S 0.0 0.2 0:00.12 systemd
941 ubuntu 20 0 114060 2676 60 S 0.0 0.1 0:00.00 (sd-pam)
984 ubuntu 20 0 284140 7460 6508 S 0.0 0.2 0:00.11 gnome-keyring-d
989 ubuntu 20 0 207716 6156 5528 S 0.0 0.2 0:00.00 gdm-x-session
1027 ubuntu 20 0 51080 5504 3800 S 0.0 0.2 0:01.26 dbus-daemon
1030 ubuntu 20 0 555464 14976 12492 S 0.0 0.4 0:00.19 gnome-session-b
1121 ubuntu 20 0 11316 320 0 S 0.0 0.0 0:00.00 ssh-agent
```

GUÍA DE LABORATORIO N°.4

Otro gestor de procesos muy interesante y usado es ' htop ', que nos mostrará sin salir de la terminal (si es que lo ejecutamos desde ésta...) algo similar a top, pero donde mediante las teclas de función del teclado, accederemos a menús de configuración al estilo de las aplicaciones DOS (qué tiempos...).

Htop

```

userlub@userlub-pc: ~
File Edit View Search Terminal Help

 0[|||||||] 29.1% Tasks: 110, 518 thr; 2 running
 1[|||||||] 63.2% Load average: 1.20 1.16 0.52
 2[|||||] 15.2% Uptime: 00:03:41
 3[|||||||] 30.2%
Mem[|||||||] 2.21G/15.5G
Swp[ ] 0K/512M

  PID USER   PRI  NI  VIRT  RES  SHR S  CPU% MEM%   TIME+  Command
 2934 userlub 20    0  976M 147M 52372 R 93.5 0.9   0:03.25 /usr/bin/perl /
1796 userlub 20    0 2794M 350M 114M S 13.3 2.2   0:36.60 /snap/firefox/2
 655 root    20    0  609M  99M 68492 S 11.3 0.6   0:07.30 /usr/lib/xorg/X
1247 userlub 20    0 12.0G 398M 174M S  6.6 2.5   2:01.23 /snap/firefox/2
1006 userlub  9   -11 1125M 22544 17548 S  2.7 0.1   0:04.21 /usr/bin/pulsea
2133 userlub 20    0 2832M 348M 98784 S  2.7 2.2   0:16.61 /snap/firefox/2
2396 userlub 20    0 2902M 438M 105M S  2.0 2.8   0:31.03 /snap/firefox/2
2932 userlub 20    0 14068 4984 3448 R  2.0 0.0   0:00.37 htop
1065 userlub -6    0 1125M 22544 17548 S  1.3 0.1   0:02.35 /usr/bin/pulsea
1076 userlub 20    0  385M 31784 22832 S  1.3 0.2   0:00.32 /usr/bin/openbo
1100 userlub 20    0 1129M 116M 92808 S  1.3 0.7   0:01.81 /usr/bin/lxqt-p
1915 userlub 20    0  226M 36880 27988 S  1.3 0.2   0:01.79 /snap/firefox/2
1951 userlub 20    0 2794M 350M 114M S  1.3 2.2   0:01.57 /snap/firefox/2
1952 userlub 20    0 12.0G 398M 174M S  1.3 2.5   0:01.52 /snap/firefox/2
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice -F8Nice +F9Kill F10Quit
  
```

GUÍA DE LABORATORIO N°.4

man htop

```
userlub@userlub-pc: ~  
File Edit View Search Terminal Help  
HTOP(1) User Commands HTOP(1)  
  
NAME  
    htop - interactive process viewer  
  
SYNOPSIS  
    htop [-dCFhpustvH]  
  
DESCRIPTION  
    htop is a cross-platform ncurses-based process viewer.  
  
    It is similar to top, but allows you to scroll vertically and horizon-  
tally, and interact using a pointing device (mouse). You can observe  
all processes running on the system, along with their command line ar-  
guments, as well as view them in a tree format, select multiple pro-  
cesses and acting on them all at once.  
  
    Tasks related to processes (killing, renicing) can be done without en-  
tering their PIDs.  
  
COMMAND-LINE OPTIONS  
    Mandatory arguments to long options are mandatory for short options  
too.  
  
    -d --delay=DELAY  
        Delay between updates, in tenths of seconds. If the delay value  
is less than 1 it is increased to 1, i.e. 1/10 second. If the  
delay value is greater than 100, it is decreased to 100, i.e. 10  
seconds.  
  
    -C --no-color --no-colour  
        Start htop in monochrome mode  
  
    -F --filter=FILTER
```

En **htop**, al tratarse de una aplicación en sí donde ya podremos configurar algunos de sus aspectos y criterios de orden, hay poco que configurar, no obstante, tal y como podemos leer en su manual, podemos hacer que inicie en modo monocromo, predefinir el delay o intervalo de refresco, etc...

Los sistemas Linux vienen con la herramienta **KILL** instalada, que usaremos para detener los procesos que necesitemos. Por defecto el comando kill envía una señal denominada TERM a un proceso que le pasaremos mediante su **PID** como argumento. Esta señal TERM pedirá a dicho proceso que termine, permitiéndole gestionar su función de cierre, completando las tareas necesarias y limpiando la información que ha cargado en memoria.

kill [PID del proceso]

GUÍA DE LABORATORIO N°.4

```
userlub@userlub-pc:~$ kill 4131
bash: kill: (4131) - Operation not permitted
userlub@userlub-pc:~$ top -u root -o %MEM

top - 07:43:31 up 15 min,  2 users,  load average: 0.20, 0.32, 0.40
Tasks: 227 total,  1 running, 226 sleeping,  0 stopped,  0 zombie
%Cpu(s):  8.5 us,  1.5 sy,  0.0 ni, 90.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem : 15884.3 total, 11295.9 free,  2455.1 used,  2133.4 buff/cache
MiB Swap:  512.0 total,  512.0 free,  0.0 used. 12801.5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
655	root	20	0	704428	121040	83904	S	7.0	0.7	0:18.98	Xorg
554	root	20	0	1171068	51080	21020	S	0.0	0.3	0:03.74	snappd
376	root	19	-1	89164	47572	46324	S	0.0	0.3	0:00.52	systemd-journal
4131	root	20	0	285484	42828	34544	S	0.3	0.3	0:00.96	gpartedbin

En la captura de aquí arriba vemos cómo nos ha dado un error que aprovecho para recalcar un punto muy importante en la seguridad de los sistemas Linux, sistemas verdaderamente multiusuario y bien definidos, donde como vemos, no permite eliminar o cancelar procesos de otros usuarios.

En el caso de encontrarnos ante un proceso que “no quiere cerrarse” por la vía diplomática que le ofrecemos con TERM, pasaremos a eliminar dicho proceso por la fuerza ejecutando el comando kill con el siguiente argumento, **pasando a root** previamente para no recibir el error que acabamos de comentar:

kill -KILL [PID del proceso]

```
userlub@userlub-pc:~$ sudo su
[sudo] password for userlub:
(base) root@userlub-pc:/home/userlub# kill -KILL 4131
(base) root@userlub-pc:/home/userlub#
```

Con este último comando, no estamos mandando al proceso ninguna señal, directamente estamos diciéndole al kernel del sistema que descarte y cierre dicho proceso.

Estas señales también pueden ser identificadas con números. Por ejemplo, en los ejemplos anteriores **TERM** puede ser pasada al proceso mediante “-15” y **-KILL** es el equivalente a pasar “-9”. Es decir, el resultado de los siguientes comandos será el mismo:

kill -9 [PID del proceso]

kill -KILL [PID del proceso]

```
top - 07:53:39 up 25 min,  1 user,  load average: 0.75, 0.57, 0.46
Tasks: 229 total,  1 running, 228 sleeping,  0 stopped,  0 zombie
%Cpu(s):  4.7 us,  1.2 sy,  0.0 ni, 94.0 id,  0.1 wa,  0.0 hi,  0.1 si,  0.0 st
MiB Mem : 15884.3 total, 10475.9 free,  2913.7 used,  2494.7 buff/cache
MiB Swap:  512.0 total,  512.0 free,  0.0 used. 12306.1 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2396	userlub	20	0	3015076	525320	108356	S	1.7	3.2	1:53.26	Isolated Web Co
5166	userlub	20	0	3428100	473248	176788	S	0.7	2.9	0:04.86	PacketTracer

GUÍA DE LABORATORIO N°.4

```
userlub@userlub-pc:~$ kill -9 5166
userlub@userlub-pc:~$ top -u userlub -o %MEM

top - 07:56:45 up 28 min, 1 user, load average: 0.29, 0.52, 0.47
Tasks: 215 total, 3 running, 212 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4.7 us, 1.3 sy, 0.0 ni, 93.8 id, 0.1 wa, 0.0 hi, 0.1 si, 0.0 st
MiB Mem : 15884.3 total, 10829.5 free, 2570.4 used, 2484.3 buff/cache
MiB Swap: 512.0 total, 512.0 free, 0.0 used. 12667.1 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2396	userlub	20	0	3026652	542680	108856	S	1.7	3.3	2:17.68	Isolated Web Co
1247	userlub	20	0	11.9g	446440	178464	R	5.3	2.7	4:15.53	firefox
1796	userlub	20	0	2963792	424080	117688	S	11.3	2.6	3:27.91	Isolated Web Co
2133	userlub	20	0	2908156	346040	99408	S	0.0	2.1	0:24.74	Isolated Web Co
2907	userlub	20	0	2732636	253148	101772	S	0.0	1.6	0:20.71	Isolated Web Co

El comando kill además de para finalizar procesos, también podemos usarlo para reiniciar ciertos servicios. Uno de los que más necesita reiniciarse suele ser Apache, sobre todo si aún estamos con la configuración base, para ir viendo que todo funciona correctamente.

Al igual que Apache, multitud de servicios necesitan ser reiniciados, y la mayoría de ellos responde al argumento '**HUP**' (**Hang up**) de kill. Mediante el siguiente comando, el servicio perteneciente a Apache, se reiniciará y volverá a cargar el fichero de configuración, permitiéndonos ver si los cambios han surtido efecto y volviendo a dar servicio a los usuarios.

kill -HUP [PID de Apache]

Como vimos anteriormente, HUP también tiene su respectiva nomenclatura en numeración, siendo el equivalente al comando anterior, la siguiente línea:

kill -1 [PID de Apache]

```
userlub@userlub-pc:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-06-09 08:01:06 EST; 10min ago
     Docs: https://httpd.apache.org/docs/2.4/
    Main PID: 6149 (apache2)
      Tasks: 55 (limit: 18914)
    Memory: 5.1M
       CPU: 53ms
    CGroup: /system.slice/apache2.service
            └─6149 /usr/sbin/apache2 -k start
              └─6151 /usr/sbin/apache2 -k start
                └─6152 /usr/sbin/apache2 -k start

jun 09 08:01:06 userlub-pc systemd[1]: Starting The Apache HTTP Server...
jun 09 08:01:06 userlub-pc apachectl[6145]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, u
jun 09 08:01:06 userlub-pc systemd[1]: Started The Apache HTTP Server.
lines 1-16/16 (END)
```

GUÍA DE LABORATORIO N°.4

```
userlub@userlub-pc:~$ kill -HUP 6149
bash: kill: (6149) - Operation not permitted
userlub@userlub-pc:~$ sudo su
(base) root@userlub-pc:/home/userlub# kill -HUP 6149
(base) root@userlub-pc:/home/userlub# top -u root -o %MEM

top - 08:15:48 up 47 min,  2 users,  load average: 0.65, 0.48, 0.48
Tasks: 226 total,  2 running, 224 sleeping,  0 stopped,  0 zombie
%Cpu(s):  6.7 us,  1.5 sy,  0.0 ni, 91.1 id,  0.7 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem : 15884.3 total, 10335.4 free,  2843.0 used,  2705.9 buff/cache
MiB Swap:  512.0 total,  512.0 free,  0.0 used. 12356.4 avail Mem

  PID USER      PR  NI   VIRT   RES    SHR S  %CPU  %MEM    TIME+  COMMAND
  655 root        20   0 695732 121416 83868 S   2.0   0.7   0:48.78 Xorg
 6071 root        20   0 435124  71420 26684 S   0.0   0.4   0:01.05 fwupd
  554 root        20   0 1171068 51080 21020 S   0.0   0.3   0:03.88 snapd
  376 root        19  -1  89164  48108 46824 S   0.0   0.3   0:00.64 systemd-journal
  977 root        20   0  89268  24416 21516 S   0.0   0.2   0:00.07 smbd
  624 root        20   0 114880  21628 13484 S   0.0   0.1   0:00.13 unattended-upgr
 6381 root        20   0 300684  20824 17944 S   0.0   0.1   0:00.03 packagekitd
  984 root        20   0  89048  20720 17928 S   0.0   0.1   0:00.03 samba-bgqd
  538 root        20   0  37876  19520 10524 S   0.0   0.1   0:00.14 networkd-dispat
```

Un dato importante es que además de por su PID, si conocemos el nombre exacto del proceso también podemos usarlo en el lugar en el que usaríamos el PID. Para esto, usaremos **kill** en lugar de **kill**, que funciona exactamente igual, pero preparado para trabajar con nombres de proceso en lugar de con PID. Es decir, estos dos comandos harán exactamente lo mismo:

kill -9 8096

pkill -9 code

```
top - 08:19:30 up 51 min,  1 user,  load average: 1.38, 0.75, 0.56
Tasks: 237 total,  2 running, 235 sleeping,  0 stopped,  0 zombie
%Cpu(s):  3.5 us,  1.0 sy,  0.0 ni, 95.2 id,  0.2 wa,  0.0 hi,  0.1 si,  0.0 st
MiB Mem : 15884.3 total,  9266.5 free,  3457.1 used,  3160.6 buff/cache
MiB Swap:  512.0 total,  512.0 free,  0.0 used. 11638.9 avail Mem

  PID USER      PR  NI   VIRT   RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 2396 userlub   20   0 3043092 548472 107952 S   1.3   3.4   3:11.62 Isolated Web Co
 1796 userlub   20   0 3056992 495648 117760 S  10.6   3.0   6:04.39 Isolated Web Co
 1247 userlub   20   0 12.0g 437076 186328 S   2.0   2.7   6:28.44 firefox
 2133 userlub   20   0 2908156 349456 99300 S   0.0   2.1   0:27.67 Isolated Web Co
 8096 userlub   20   0 1122.8g 344976 60764 S   0.0   2.1   0:23.13 code
```


GUÍA DE LABORATORIO N°.4

```
userlub@userlub-pc:~$ pkill -9 code
userlub@userlub-pc:~$ top -u userlub -o %MEM
```

```
top - 08:21:08 up 53 min, 1 user, load average: 0.53, 0.62, 0.53
Tasks: 220 total, 1 running, 219 sleeping, 0 stopped, 0 zombie
%Cpu(s): 2.7 us, 0.8 sy, 0.0 ni, 96.3 id, 0.1 wa, 0.0 hi, 0.2 si, 0.0 st
MiB Mem : 15884.3 total, 9981.0 free, 2809.0 used, 3094.3 buff/cache
MiB Swap: 512.0 total, 512.0 free, 0.0 used, 12362.0 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2396	userlub	20	0	3057140	638900	109712	S	1.7	3.9	3:24.12	Isolated Web Co
1796	userlub	20	0	3064236	529712	119684	S	10.0	3.3	6:15.48	Isolated Web Co
1247	userlub	20	0	12.0g	443964	186828	S	3.0	2.7	6:38.14	firefox
2133	userlub	20	0	2910920	352556	102068	S	0.0	2.2	0:28.03	Isolated Web Co

killall es una variante del comando **kill** con el que enviaremos la misma señal a todos los procesos pertenecientes a un programa. Por ejemplo:

killall Firefox

Con estos comandos y herramientas ya podremos gestionar de forma correcta y eficiente los procesos de nuestro sistema, monitorizándolos para ver si hay algo que no debiese estar, o que se encuentre consumiendo recursos por encima de lo normal; optimizando así nuestra distribución y el aprovechamiento que hacemos de nuestro hardware.

```
ubuntu@ubuntu18: ~
```

Archivo Editar Ver Buscar Terminal Ayuda

```
top - 17:45:03 up 47 min, 1 user, load average: 0.11, 0.15, 0.13
Tareas: 218 total, 1 ejecutar, 165 hibernar, 0 detener, 0 zombie
%Cpu(s): 0.5 usuario, 0.3 sist, 0.0 adecuado, 99.0 inact, 0.2 en espera, 0.
KiB Mem : 3436140 total, 728032 libre, 1244596 usado, 1463512 búfer/caché
KiB Intercambio: 2097148 total, 2097148 libre, 0 usado, 1981664 dispon
```

PID	USUARIO	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	HORA+	ORDEN
4972	ubuntu	20	0	3039792	269632	147416	S	1.0	7.8	0:03.85	firefox
5195	ubuntu	20	0	46844	4188	3496	R	0.7	0.1	0:00.21	top
991	ubuntu	20	0	782672	95744	57192	S	0.3	2.8	0:40.67	Xorg

```
ubuntu@ubuntu18:~$ killall firefox
```

G. CONSIDERACIONES FINALES: Opinión sobre el logro del objetivo y el desarrollo de la experiencia.

La experiencia del laboratorio fue enriquecedora permitiendo tener una mejor comprensión del manejo de los procesos en la terminal. El desarrollo de la experiencia pudo llevarse a cabo exitosamente gracias a los recursos provistos en el laboratorio, tales como: guía del mismo y los equipos asociados al laboratorio.



UNIVERSIDAD TECNOLÓGICA DE PANAMÁ
FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES
DEPARTAMENTO DE ARQUITECTURA Y REDES DE COMPUTADORAS



GUÍA DE LABORATORIO N°.4
