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: IILI43

1. **TÍTULO DE LA EXPERIENCIA:** Comandos para administrar y gestionar fácilmente los procesos en

Linux.

1. **TEMAS:**

* Administración y gestión de procesos en Linux.

1. **OBJETIVO:**
   * Conocer y utilizar los comandos más importantes para la gestión de procesos en Linux.
2. **RECURSOS:**

* Un computador personal.
* SO Linux Fedora.
* Guía del laboratorio.

1. **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 |

1. **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.

**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.

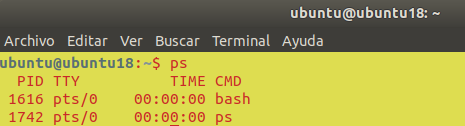
Los **procesos**juegan un papel muy importante en las distribuciones Linux, ya que son los que consumirán estos recursos hardware tan preciados en entornos de producción, **administrarlos y gestionarlos correctamente**es de vital importancia ya que estos procesos y la gestión que hace el sistema sobre 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.

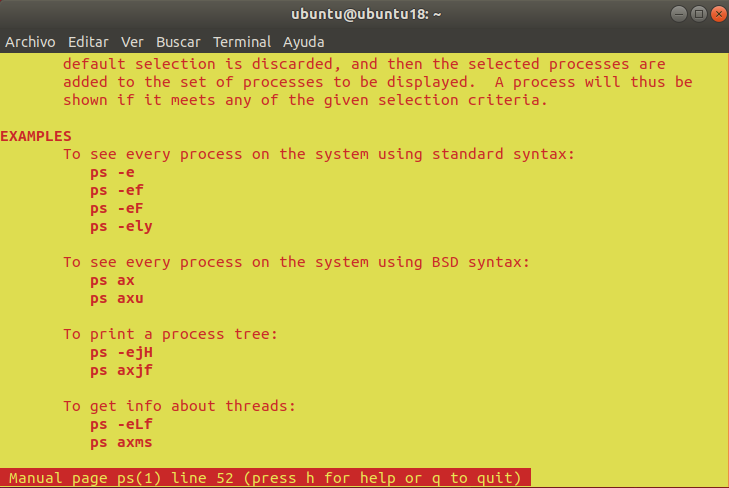
**Explicación y ejecución de los comandos**

ps [opciones]



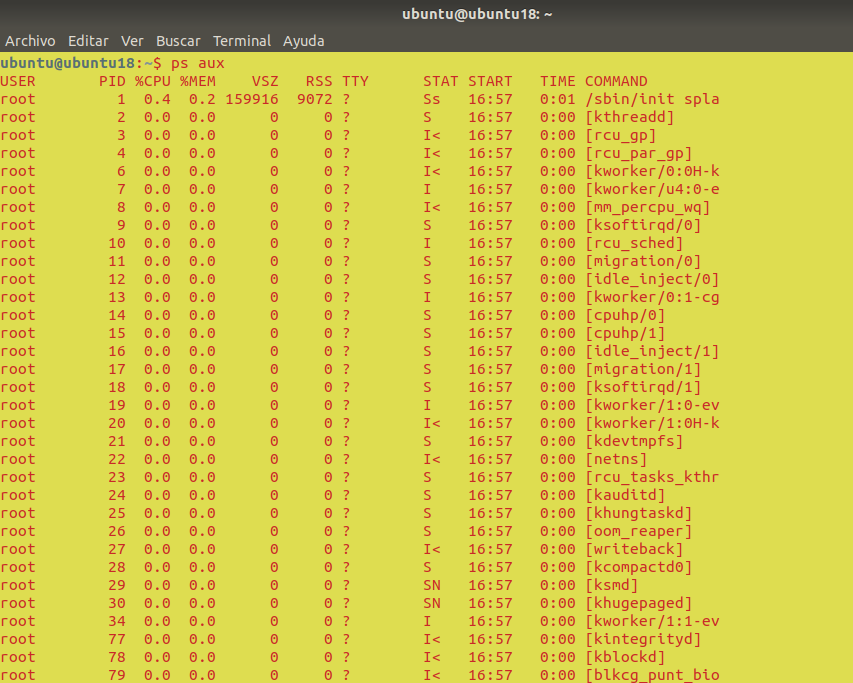
Como de costumbre, podemos **revisar el manual de ps**dentro del sistema para conocer todas las opciones posibles:

man ps



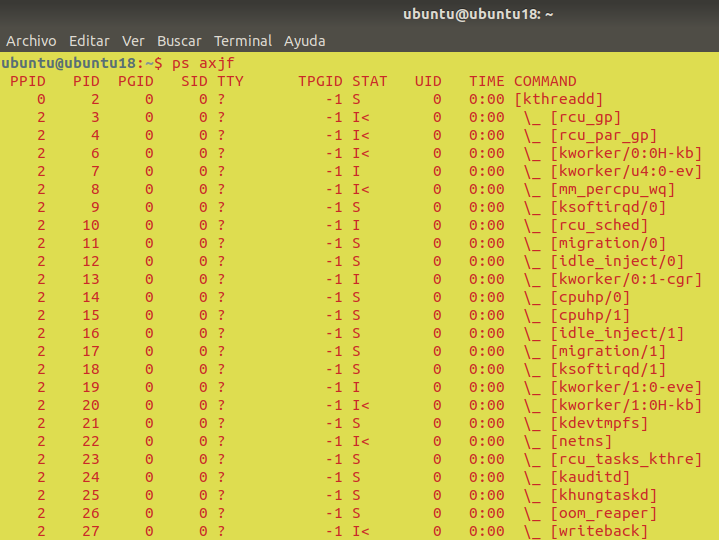
**Siendo las más habituales**:

ps aux (muestra todos los procesos del sistema)



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.



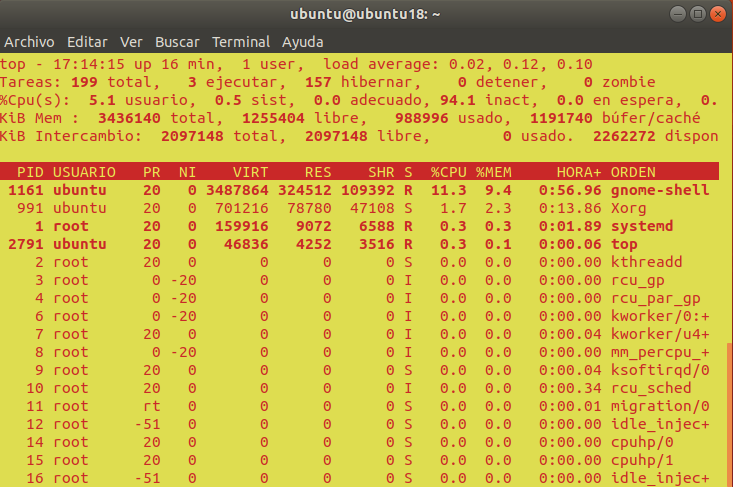
ps aux | grep bash



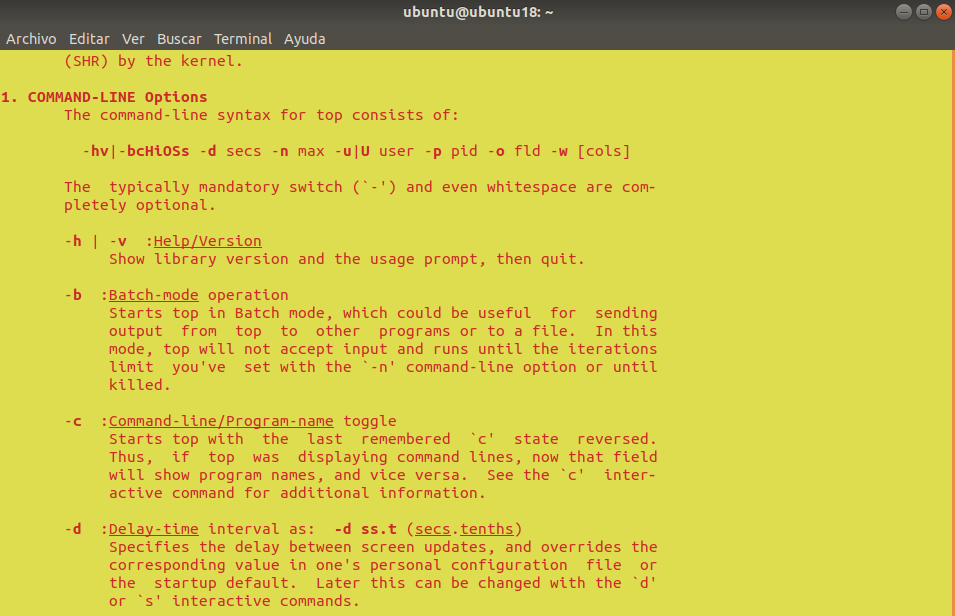
El **PID**es el **número identificador de proceso**que le asigna el sistema a cada proceso que se inicia, mientras que el **command**es el programa al cual pertenece dicho proceso.

**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



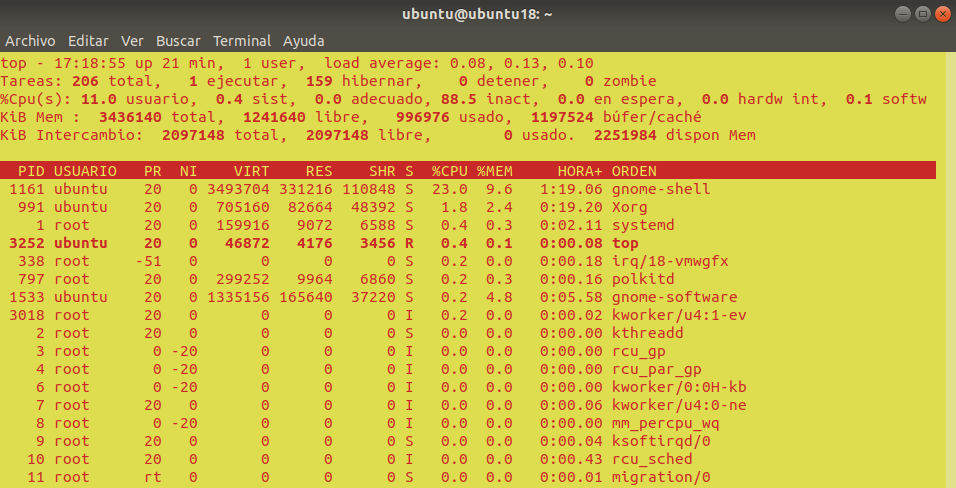
man top

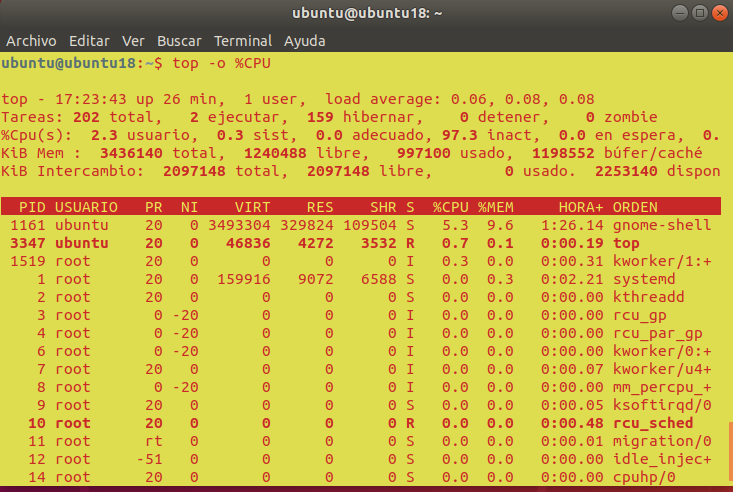


Aquí, como vemos en su manual, podemos controlar más aspectos, como los de los siguientes ejemplos entre otros:

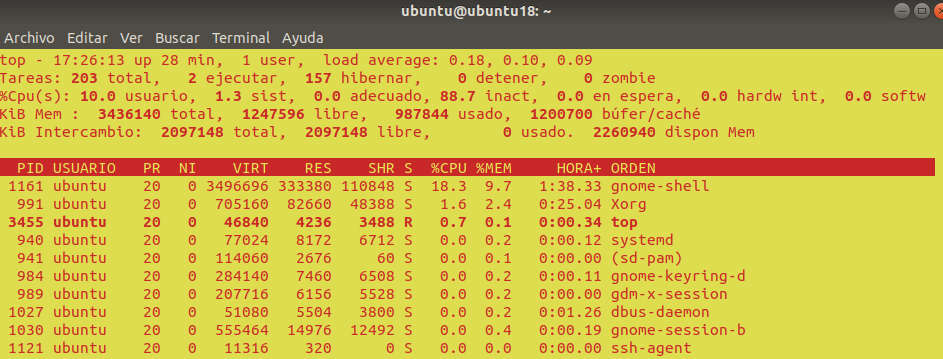
top –d 5  (Donde 5 es el número de segundos a transcurrir entre cada muestreo)

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



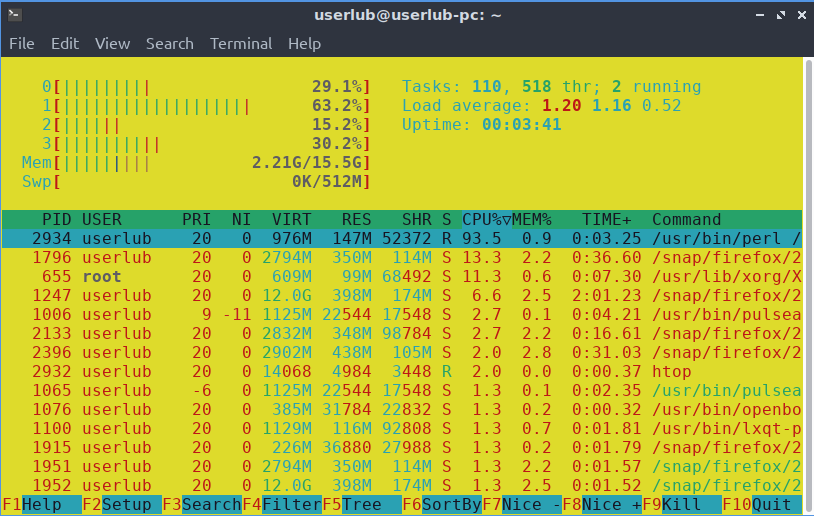


top –u ubuntu  (Donde **ubuntu**es el usuario del cual queremos mostrar los procesos).



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



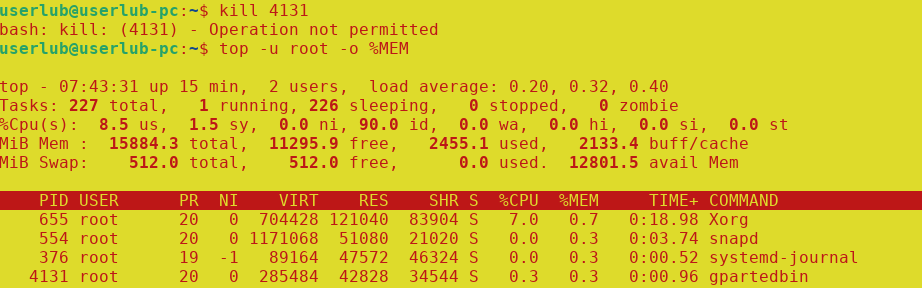
man htop



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.

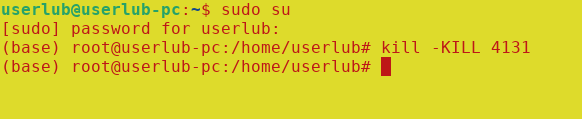
kill [PID del proceso]



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]

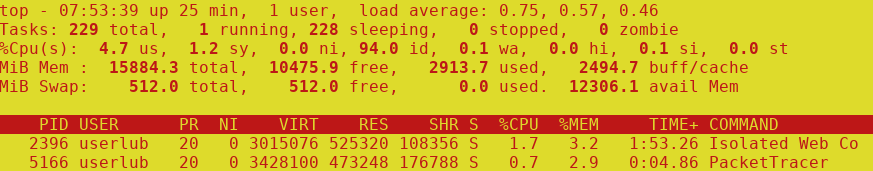


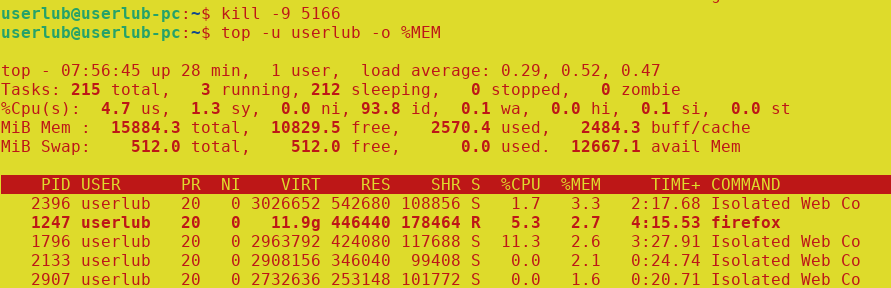
Con este último comando, no estamos mandado al proceso ninguna señal, directamente estamos diciéndole al kernel del sistema que descarte y cierre dicho proceso.

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]





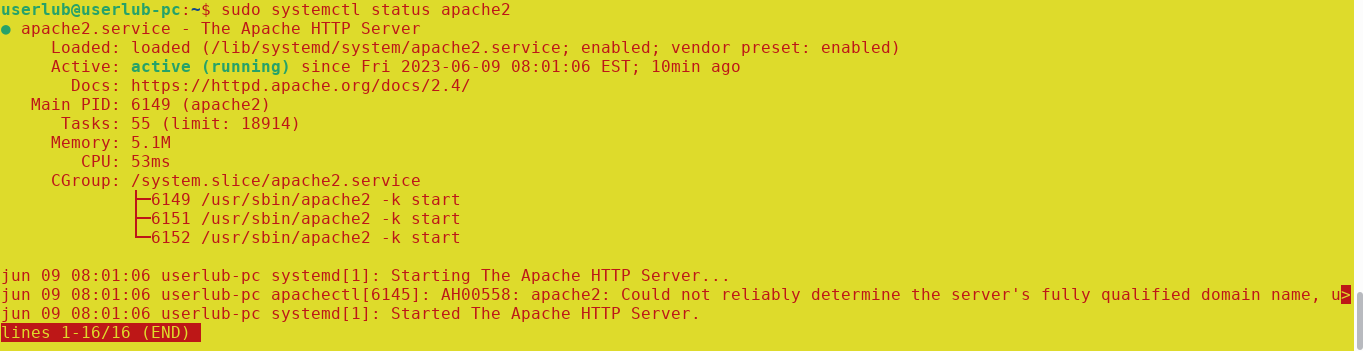
El comando kill además de para finalizar procesos, también podemos usarlo para reiniciar ciertos servicios. Uno de los que más necesita reiniciarse suele ser Apache, sobre todo si aún estamos con la configuración base, para ir viendo que todo funciona correctamente.

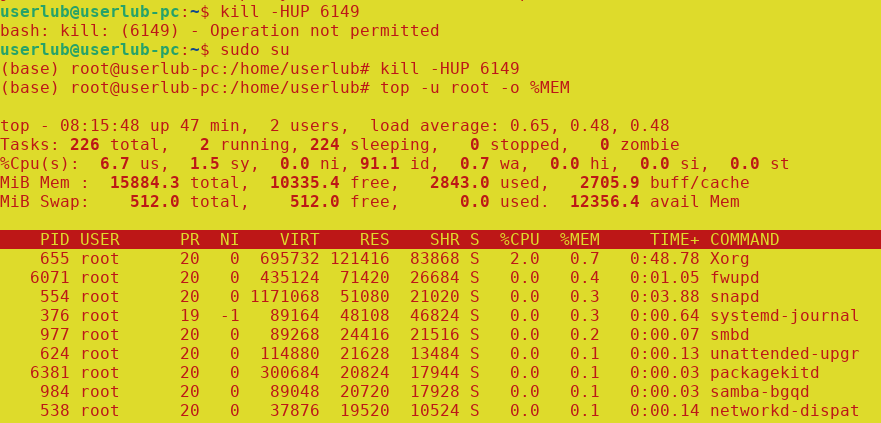
Al igual que Apache, multitud de servicios necesitan ser reiniciados, y la mayoría de ellos responde al argumento **‘HUP’ (Hang up)**de kill. Mediante el siguiente comando, el servicio perteneciente a Apache, se reiniciará y volverá a cargar el fichero de configuración, permitiéndonos ver si los cambios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

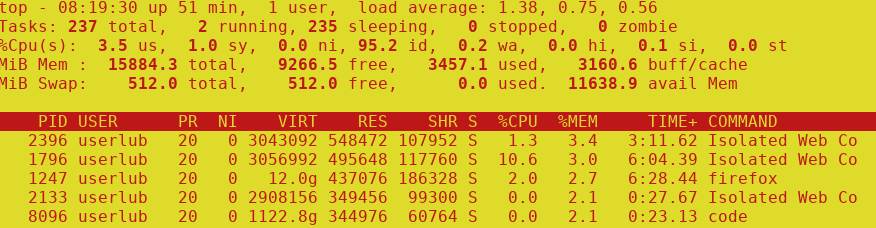


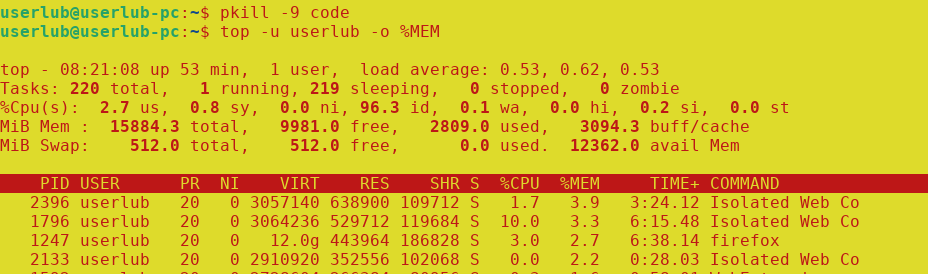


Un dato importante es que además de por su PID, si conocemos el nombre exacto del proceso también podemos usarlo en el lugar en el que usaríamos el PID. Para esto, usaremos **pkill** en lugar de **kill**, que funciona exactamente igual, pero preparado para trabajar con nombres de proceso en lugar de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

kill -9 8096

pkill -9 code

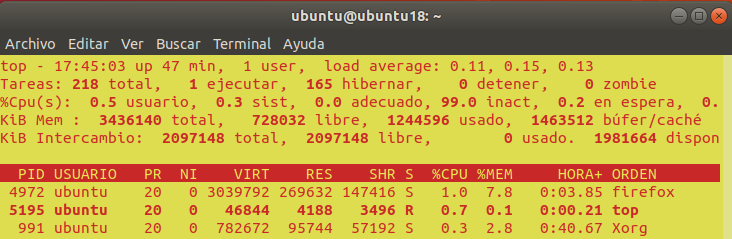




**killall** es una variante del comando **kill** con el que enviaremos la misma señal a todos los procesos pertenecientes a un programa. Por ejemplo:

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.





1. **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.*