**Gestión de procesos:**

El sistema operativo es el software base de nuestro sistema informático. Este es el encargado de generar una interfaz entre el usuario y la máquina, sirviendo de base para la ejecución del software de aplicación o de desarrollo.

**Funciones:**

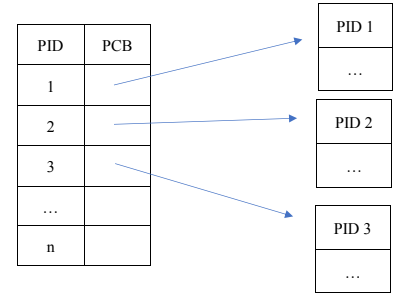
* Gestionar eficientemente los recursos hardware y software del sistema informático.
* Controlar y administrar la ejecución de programas.
* Controlar y administrar el sistema de archivos.
* Detectar e intentar solucionar errores que se puedan producir

**Procesos:**

Un proceso es un programa que está en ejecución. Cuando un usuario solicita ejecutar un fichero binario, el gestor de procesos preparará una imagen del proceso que contendrá:

* Bloque de Control del Proceso (BCP).
* Instrucciones del programa en código máquina.
* Datos estáticos del programa (constantes y variables).
* Pilas de usuario y kernel (llamada a funciones).
* Heap (datos creados de forma dinámica).

A la hora de ejecutar un proceso el SO, **si no disponemos de memoria virtual, la imagen se cargará totalmente en memoria**, siendo la carga parcial si la tenemos configurada.



**Elementos del PCB (imagen del proceso):**

*¿Qué es PCB? →* Es una estructura denominada task\_struct perteneciente al kernel de Linux

* **Puntero**. Se guarda cuando el proceso cambia de estado y debemos retener la posición actual del proceso.
* **Estado del proceso.**
* **Número de proceso.**
* **Contador de programa**. Dirección de memoria de la siguiente instrucción a ejecutar.
* **Registros**. Contiene el valor de los registros de la CPU, se utiliza para los cambios de contexto.
* **Espacio de direcciones.**
* **Lista de archivos abiertos.** Incluye la lista de archivos abiertos por el proceso.
* **Datos de contabilidad y estado**. Registra datos sobre la cantidad de CPU utilizada, restricciones de tiempo, trabajos, etc

**Tipos de procesos:**

1. ***Ejecución en Modo Kernel:***

Un proceso en modo kernel tiene acceso sin restricciones al hardware del equipo. El código comparte un único espacio de direcciones, esto es peligroso ya que si un controlador escribe una dirección incorrecta todo el S.O puede ser afectado. Para solucionar este problema acudimos a kernel panic que detiene los servicios y si es necesario reinicia o apaga el sistema

1. ***Ejecución en Modo Usuario:***

Este modo de ejecución permite un subconjunto limitado de instrucciones del procesador. Los procesos deben solicitar acceso al kernel para obtener recursos, cada uno de ellos tiene diferentes direcciones virtuales privadas. Si una aplicación se bloquea, solo le afecta a ella.

1. ***Ejecución en Primer Plano:***

Son procesos que necesitan interacción. Si el proceso se inicia en la terminal, usarán esta hasta que finalice el proceso. Durante su ejecución no se puede ejecutar otro comando. Llamados Foreground

1. ***Ejecución en Segundo Plano:***

No deberemos esperar a su finalización para poder interactuar con la terminal. Son procesos que están corriendo en el sistema sin la atención e interacción del usuario. Llamados Background. Para ejecutarlo debemos usar el carácter & al final.

**Daemons:**

Los daemons son procesos en segundo plano, pero no todos los procesos en segundo plano son daemons. Como por ejemplo: sshd, que gestiona las conexiones de red en servidores.

**El ciclo de vida de los procesos:**

**Creación:**

* Arranque del sistema.
* Ejecución de un proceso, llamada para la creación de otro proceso.
* Petición del usuario.
* Inicio de trabajo por lotes.

**Terminación:**

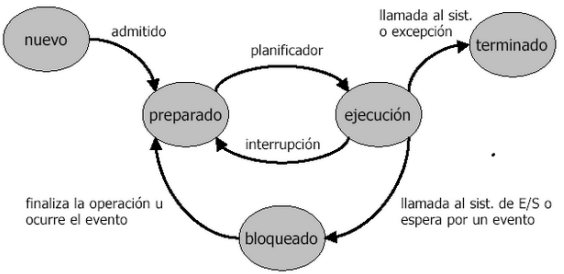
* **Salida normal**. El proceso termina de forma voluntaria.
* **Salida por error.** El proceso finaliza cuando presenta un error.
* **Terminación por error fatal**. Este tipo de finalización sucede por un error crítico en el programa.
* **Eliminación por otro proceso**. Ocasionada por petición expresa de otro proceso. Ej: cuando usamos el comando kill

**Modelo de 5 estados:**.

* **Apropiativos:** El sistema operativo puede interrumpir un proceso en ejecución para asignar la CPU a otro proceso de mayor prioridad
* **No apropiativos:** Se asigna la CPU por un periodo de tiempo, sin la posibilidad de que otro proceso le arrebate su uso

***5 estados:***

* **Nuevo.** Acaba de ser creado y está a la espera de su admisión
* **Listo o preparado.** El proceso está esperando para ser ejecutado por la CPU.
* **En ejecución**.
* **Bloqueado**. El proceso está esperando a que ocurra un determinado evento.
* **Terminado.** El proceso ha finalizado.



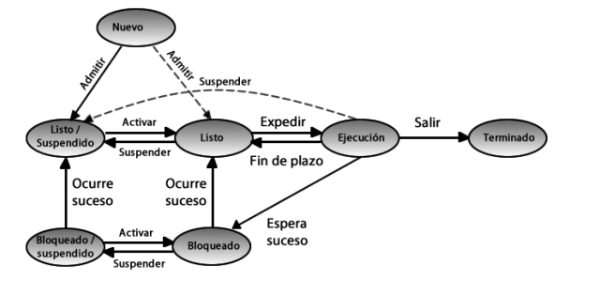
* **Planificador a largo plazo:** Determina qué procesos se admiten en el sistema.
* **Planificador a corto plazo:** Se encargará de mover procesos entre la memoria principal y el disco
* **Planificador a medio plazo:** Responsable de decidir qué proceso se ejecuta en la CPU, seleccionando de entre los trabajos cargados en la cola.

**Modelo de 7 estados:**

* **Suspendido bloqueado:** Asignado a aquellos procesos que están bloqueados. Al reanudar los procesos volverán a quedar bloqueados.
* **Suspendido listo:** Similar al anterior, pero desde el estado listo

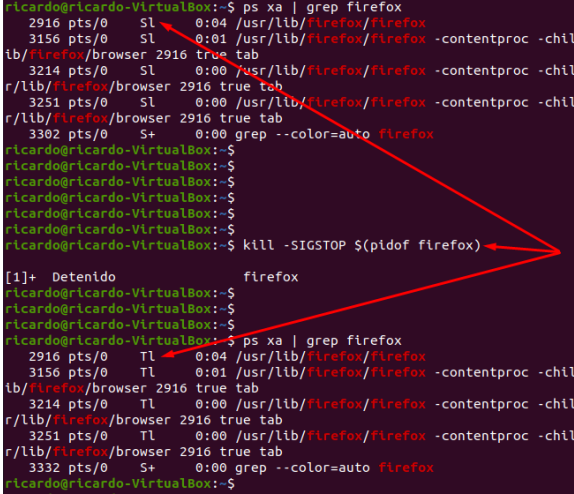
Para suspender un proceso desde sistemas Windows tendremos que utilizar la herramienta

Process Explorer. En el caso de Linux, enviando una señal SIGSTOP a través del comando “kill” nos valdría.



**Estados de los procesos en entornos Linux:**

* **S: Durmiendo;** El proceso está en ejecución, pero no está ejecutando ninguna instrucción en la CPU.
* **D: Estado bloqueado;** No puede ser interrumpido hasta que finalice la operación.
* **I**: Hilo del kernel inactivo
* **T**: Parado por la señal SIGSTOP
* **R: En ejecución;** Es un proceso que se está ejecutando de forma activa en la CPU
* **X: Muerto**; No debería aparecer el proceso
* **Z: Zombie**; Es un proceso que debería haber muerto, pero aún tiene dependencias y no puede finalizar.



Además de lo anterior, para formatos BSD (ps aux), siempre y cuando aparezca el estado de los procesos, mostraremos los caracteres adicionales:

* <. Alta prioridad.
* N. Baja prioridad.
* L. Páginas bloqueadas en memoria.
* s. Líder de sesión. Se da cuando el PID == SID.
* l. Multihilo o subprocesos múltiples.
* +. Proceso en foreground



**Estados de los procesos en entornos Windows:**

* **Preparado**. Los procesos están dispuestos para ser ejecutados.
* **Ejecución**. El proceso tiene control del procesador.
* **Bloqueado**. Son procesos que no pueden ejecutarse debido a que necesitan de algún recurso disponible.
* **Suspendido bloqueado**. Son procesos que han sido suspendidos mientras esperaban un determinado evento.
* **Suspendido programado**. Similar a la anterior, aunque en esta ocasión no parte de un estado de bloqueo.
* **Finalizado**. El proceso ha finalizado su ejecución

**Relación entre trabajo, tarea, proceso:**

***Trabajo:***

Es cualquier programa que iniciamos en la shell y que permanece vinculado a ella. Está compuesto por un conjunto de tareas y procesos que, al combinarse, completan su ejecución.

***Tarea:***

Una tarea es la unidad más pequeña de un trabajo. Puede ejecutarse en paralelo con otras tareas para completar el trabajo.

***Trabajo vs Tarea:***

Un trabajo es el proceso completo, mientras que una tarea es una unidad específica dentro de ese trabajo que ayuda a completarlo

***Proceso vs Programa:***

La principal diferencia entre ellos es que el proceso contiene una imagen del proceso y funciona mientras está en ejecución

**Concepto de hilo de ejecución:**

Unidad de procesamiento mínima que puede ser planificada por un sistema operativo. Los hilos permiten la ejecución concurrente dentro de un proceso, compartiendo su memoria y recursos.

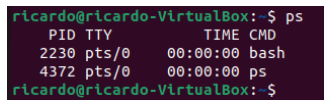
Un proceso está compuesto por uno o más hilos que comparten los mismos recursos. Esto permite que cualquier hilo pueda modificar los datos.

**Herramientas para la gestión de procesos**

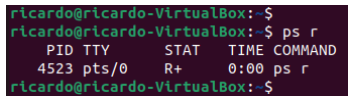
***Herramienta ps:***

Muestra información sobre una selección de los procesos activos en el sistema.

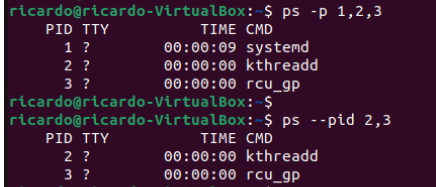
Por defecto, ps selecciona todos los procesos asociados al usuario actual y a la misma terminal. Muestra el PID, TTY, tiempo de CPU acumulado y CMD.

****

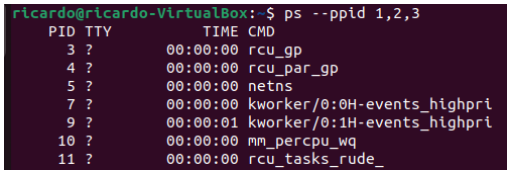
* **ps -r: Restringe la selección solo a procesos en ejecución.**

****

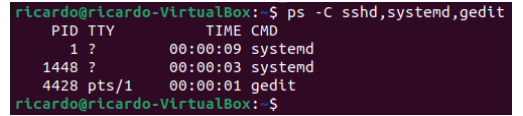
* **ps -p: Devuelve el proceso con el PID introducido.**

****

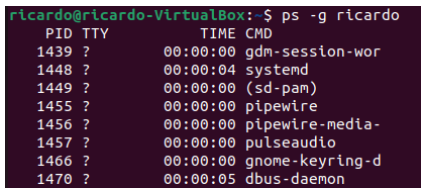
* –**ppid: Selecciona los procesos que son hijos del indicado**



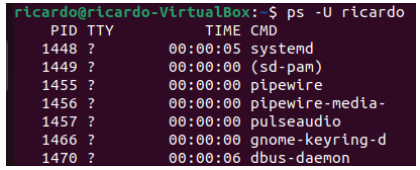
* **-C. Selecciona por nombre de comando.**

****

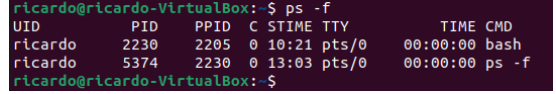
* **-G. Filtra por ID o nombre de grupo real.**

****

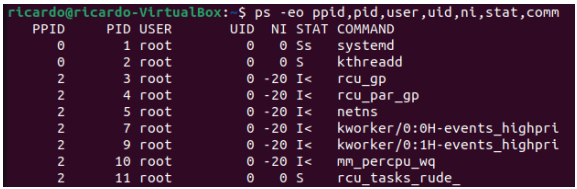
* **-U. Similar a los grupos, en este caso tendremos los usuarios.**

****

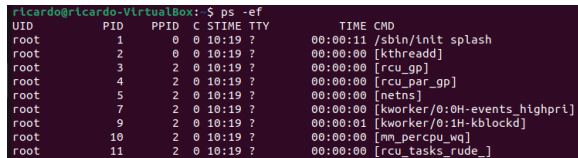
* **-f. Lista de procesos en formato completo.**

****

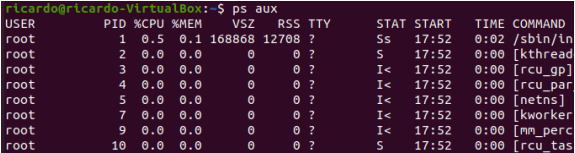
* **-o. Formato definido por el usuario.**

****

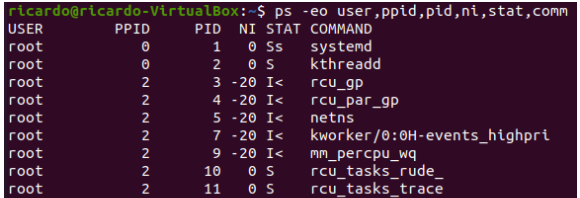
* **Mostrar todos los procesos en formato largo.**

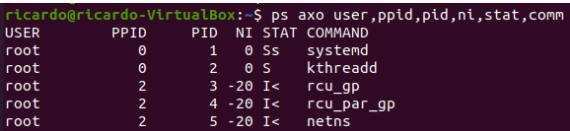
****

* **Mostrar todos los procesos en formato orientado al usuario.**

****

* **Mostrar todos los procesos con un formato definido por el usuario.**

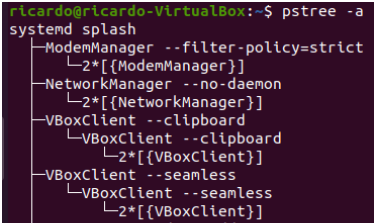
****

****

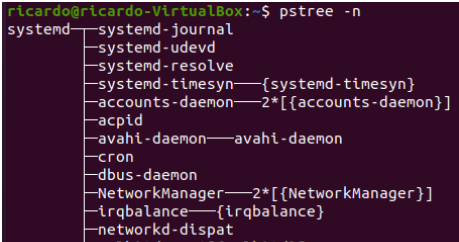
***Herramienta pstree:***

Esta utilidad muestra los procesos en ejecución como un árbol.

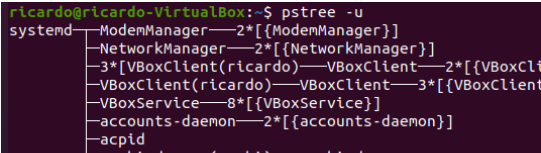
* -a. Muestra argumentos de la línea de comandos.



* **-n. Ordena procesos con el mismo padre por PID en lugar de por nombre.**



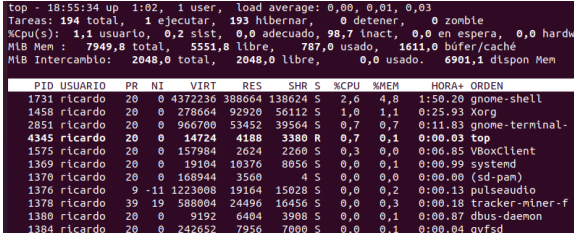
* **-u. Muestra los usuarios asociados.**



***Herramienta top:***

La herramienta top muestra en tiempo real el estado del sistema y los procesos en ejecución.





***Herramienta htop:***

Tendremos la posibilidad de interactuar con el teclado y el puntero del ratón. Con ella podremos buscar, filtrar, mostrar en forma de árbol, cambiar la prioridad o matar un proceso.

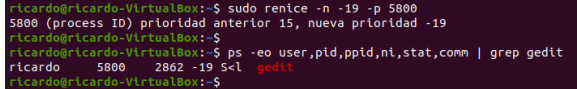
***Herramientas nice y renice***

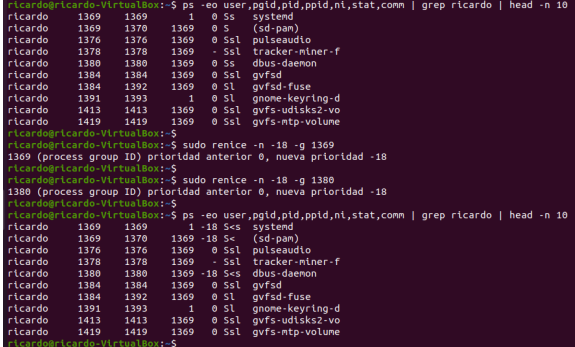
Permiten establecer y cambiar la prioridad de un proceso. Por defecto, los procesos inician con una prioridad de 0. El rango de prioridad varía de -20 a 19, donde un valor menor indica mayor prioridad



En el caso de renice utilizaremos la opción -n para cambiar la prioridad y -g, -p o -u según

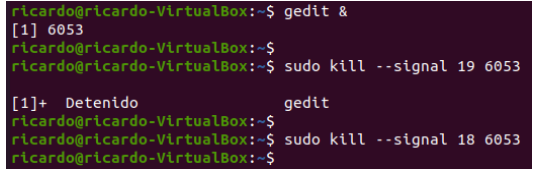
queramos cambiar la prioridad de un único proceso, de un grupo de ellos o usuario.

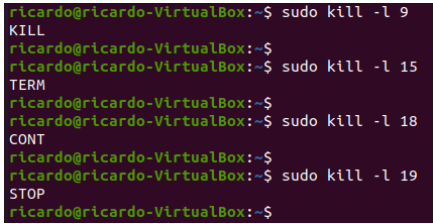




***Herramienta kill***

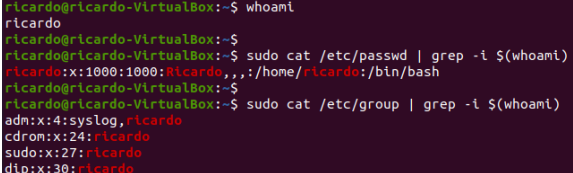
Envía señales para gestionar o terminar los procesos que están en ejecución. Las señales más comunes son SIGTERM (15) que solicita al proceso que se termine de forma ordenada, SIGKILL (9) que fuerza la terminación inmediata del proceso y SIGHUP (1) que reinicia el proceso sin terminarlo





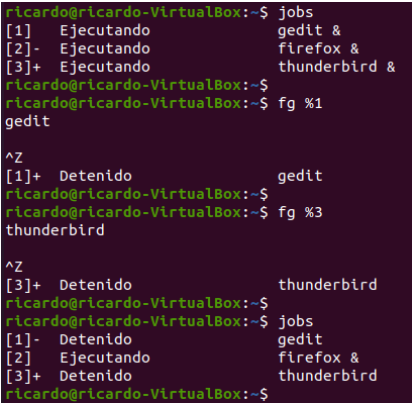
***Herramienta w, who y whoami***

* **w:** muestra información sobre los usuarios que están haciendo uso actualmente de la máquina.
* **who:** la usaremos para ver quién se encuentra logueado en el sistema
* **whoami:** arroja el user id efectivo



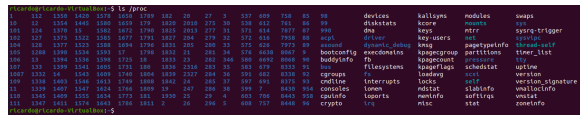
***Herramientas fg, bg, jobs***

* **JOBS**: Visualiza los trabajos que están corriendo en el sistema.
* **FG**: Trae un trabajo de segundo plano al primer plano
* **BG**: Pasa un trabajo a segundo plano

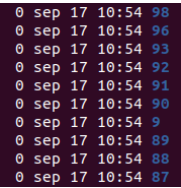


***Directorio /proc***

Se crea cada vez que arrancamos el sistema. Este directorio especial contiene un conjunto de ficheros virtuales, almacenando todos los detalles del sistema.



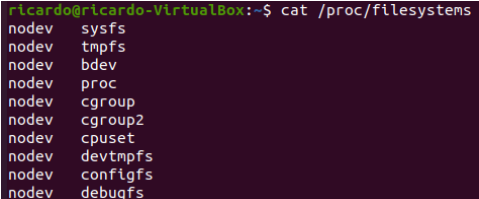
Los directorios numerados corresponden a procesos que se encuentran en ejecución, los enlaces simbólicos apuntan al proceso actual y los archivos virtuales proveen información relacionada con el hardware, con archivos o los parámetros del kernel.



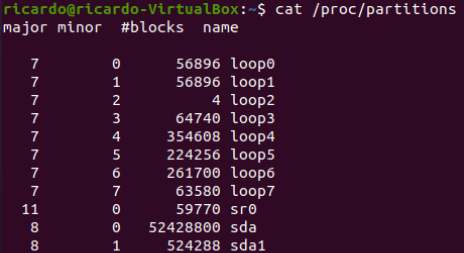
* **/proc/cmdline. Parámetros que fueron utilizados con el kernel en el momento del arranque.**



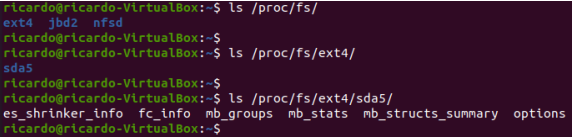
* **/proc/cpuinfo. Datos sobre nuestro procesador.**
* **/proc/stat. Estadísticas del sistema.**
* **/proc/devices. Dispositivos actuales configurados y cargados de carácter y bloque. /proc/ide y /proc/scsi sirven datos de los distintos dispositivos.**
* **/proc/filesystems**. Tipos de sistemas de archivos soportados por el kernel. El primer campo indica si el sistema de ficheros está montado en un dispositivo de bloques y el segundo el propio sistema.



* **/proc/partitions y /proc/swap. Muestran las particiones y el espacio de intercambio respectivamente.**



* **/proc/fs. Contiene información asociada a cada sistema de ficheros.**

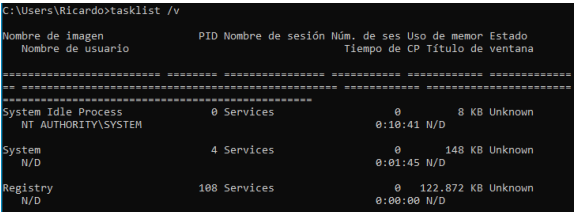


**Herramientas para la gestión de procesos en entornos Windows**

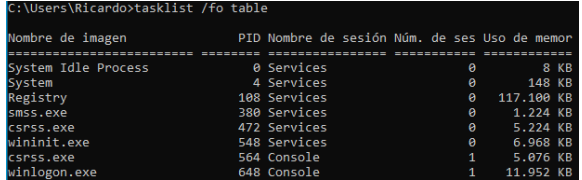
***Herramienta tasklist***

Obtiene todas las aplicaciones que se están ejecutando en el equipo

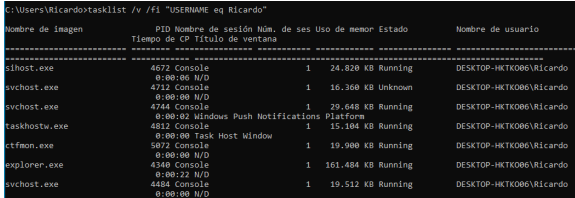
* **tasklist /v. Muestra información detallada**



* **tasklist /fo {table | list | csv} Indica el formato de salida**

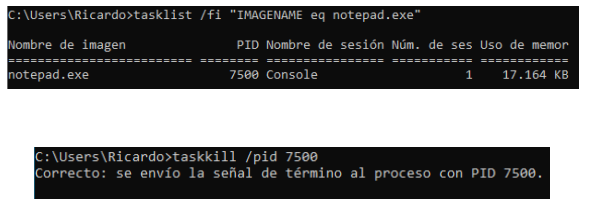
****

* **tasklist -fi. Lo usaremos para filtrar los procesos**



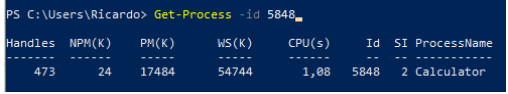
***Herramienta taskkill***

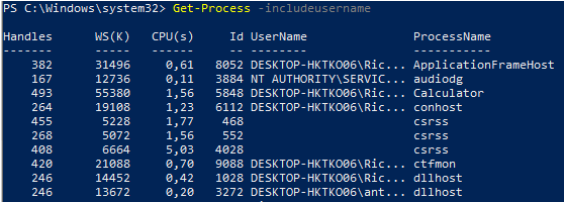
Utilizado para finalizar procesos.

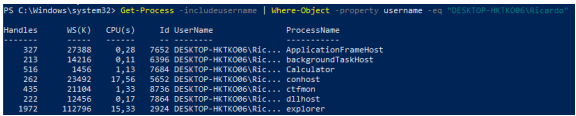


**Herramientas cmdlets Process en Windows PowerShell**

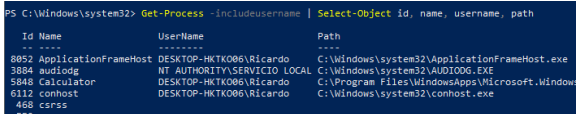
Para obtener los procesos que se están ejecutando en el equipo local, utilizaremos Get-Process.





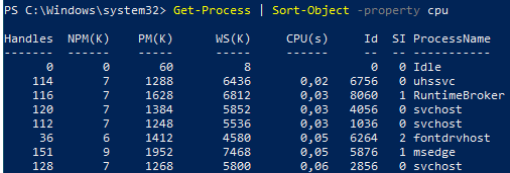


A la salida generada por Get-Process, podemos aplicarle un Select-Object para obtener ciertas propiedades específicas de un objeto o conjunto de ellos.

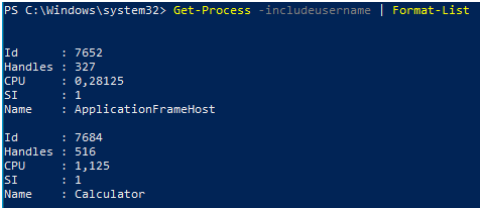


Con el comando Sort-Object tendremos la opción de ordenar el listado según el campo

que deseemos.



La elección de formato para la salida, esto lo haremos como se puede observar en las siguientes imágenes.



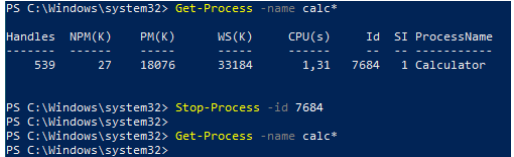
Es importante resaltar que Format-Wide solamente dejará mostrar una propiedad de cada

objeto.



A partir del listado de procesos podemos proceder a pararlos. Para ello, utilizaremos la

utilidad Stop-Process.



**Interrupciones y excepciones asociadas al uso de la CPU:**

**Interrupciones:**

Una interrupción es una señal al procesador que le indica detener su ejecución actual y ejecutar código específico para manejar una determinada situación.

* **Interrupción software:** Son aquellas generadas por un programa en ejecución
* **Interrupción hardware**: Generadas por periféricos para notificar al procesador que necesitan atención

**Excepciones:**

Las excepciones son interrupciones por errores como divisiones por cero o accesos inválidos. Protegen la integridad del sistema y, si no pueden resolverse, el sistema aborta el proceso afectado.

**¿Que es un cambio de contexto?**

Es cuando el sistema operativo pausa la ejecución de un proceso, guarda su estado, para reanudarlo más tarde y ejecuta un nuevo proceso.

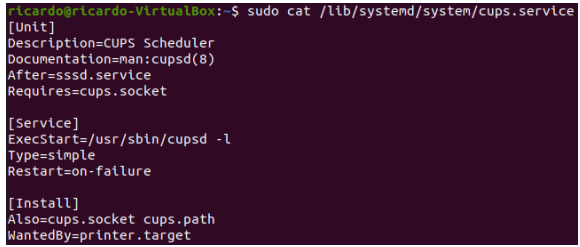
**¿Que es una violacion de segmento?**

Es una señal que nos avisa cuando hay un error inesperado durante la ejecución

**Secuencia de arranque**

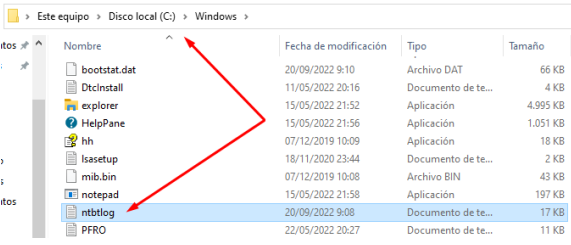
Un daemon o servicio es un programa que se ejecuta en segundo plano, sin interacción directa con el usuario, y ofrece funciones al sistema.

Los daemons se inician con el sistema y gestionan peticiones locales, remotas o del hardware, ofreciendo servicios a otros programas.



**Proceso de arranque en sistemas Windows**

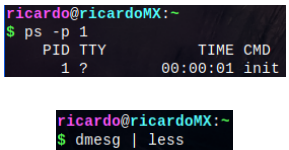
1. Se cargan los ajustes del firmware y se verifica el estado del hardware.
2. Se localiza el registro de volumen de arranque y se ejecuta el Windows Boot Manager (desde el BCD).
3. BOOTMGR llama a Winload.exe, que detecta el hardware y carga NTOSKRNL.exe y HAL.dll. Aquí se inicia el núcleo y se cargan los servicios del sistema operativo



1. Se ejecuta Winlogon y se muestra la pantalla de login.

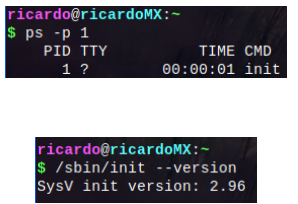
**Proceso de arranque en sistemas GNU/Linux**

1. La CPU ejecuta el firmware (BIOS/UEFI) al encender el ordenador.
2. El firmware verifica el hardware y busca el cargador de arranque.
3. El cargador de arranque carga el kernel y el contenido de /dev/initrd.
4. El kernel descomprime y mueve el contenido de /dev/initrd a /dev/ram0.
5. Si existe, el programa /linuxrc se ejecuta para cargar controladores.
6. El kernel monta la partición raíz y ejecuta el programa /sbin/init como primer proceso.



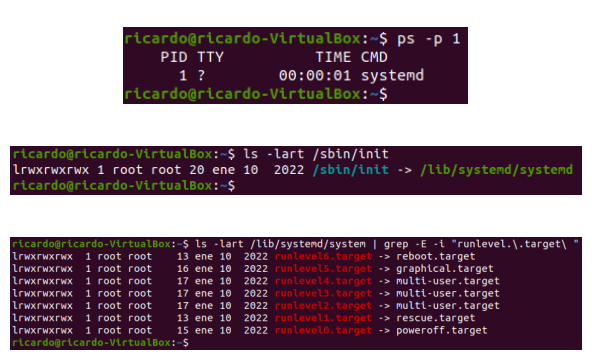
**Gestor de servicios SysVinit**

El programa init inicia servicios en segundo plano durante la inicialización, adaptándose a las necesidades del usuario. Los niveles de ejecución (0 a 6) determinan qué servicios se inician, con un nivel predeterminado establecido al arrancar.



**Gestor de servicios Systemd**

Systemd reemplaza múltiples scripts de shell por un programa monolítico que utiliza archivos de configuración individuales para cada servicio, utilizando units y targets en lugar de niveles de ejecución.



**Preguntas:**

**1. En relación al hardware, ¿por qué necesitamos al sistema operativo?**

Es el encargado de generar una interfaz entre el usuario y la máquina, sirviendo de base para la ejecución del software de aplicación o de desarrollo.

**2. Comenta los tipos de procesos según su modo de ejecución y el plano donde se ejecuten.**

Según su modo de ejecución tenemos:

* **Ejecución en modo kernel:** los procesos tienen acceso total al hardware. Esto puede ser peligroso si hay errores, por eso existe el kernel panic.
* **Ejecución en modo usuario:** los procesos tienen acceso limitado y están aislados, lo que protege al sistema si uno falla

Según el plano donde se ejecutan:

* **Ejecución en primer plano:** Son procesos que necesitan interacción
* **Ejecución en segundo plano:** No deberemos esperar a su finalización para poder interactuar con la terminal

**3. ¿Qué ocurre cuando suspendemos un proceso?**

Suspender un proceso significa pausarlo y, si hace falta, liberar su memoria en moviéndola a la memoria de intercambio para dejar espacio a otros procesos.

**4. ¿Qué implica que un proceso, en la especificación de su estado, tenga un carácter +?**

Significa que es un proceso en segundo plano (Foreground)

**5. Define el concepto de trabajo. ¿Qué comando utilizaremos en GNU/Linux para visualizarlos?**

Es cualquier programa que iniciamos de forma interactiva y que no se desvincule de la misma. Podemos visualizar los trabajos con el comando jobs

**6. ¿Qué es un hilo de ejecución? ¿Por qué decimos que los thread son elementos esenciales en el proceso de concurrencia?**

Unidad de procesamiento mínima que puede ser planificada por un sistema operativo. Los hilos permiten la ejecución concurrente dentro de un proceso, compartiendo su memoria y recursos, lo que hace mejorar la eficiencia del sistema

**7. ¿Qué implica que un proceso esté corriendo en primer plano? ¿Cómo lo paso a segundo plano?**

Implica que el proceso necesite la interacción del usuario. Para moverlo a segundo plano usamos el comando bg

**8. ¿Qué diferencia existe entre una interrupción software y una interrupción hardware?**

Una interrupción software la genera un programa cuando necesita que el sistema operativo realice una tarea, mientras que una interrupción hardware la provoca un dispositivo para avisar al procesador que requiere atención

**9. ¿Que es un cambio de contexto?**

Es cuando el sistema operativo pausa la ejecución de un proceso, guarda su estado, para reanudarlo más tarde y ejecuta un nuevo proceso.

**10. ¿Que es una violacion de segmento?**

Es una señal que nos avisa cuando hay un error inesperado durante la ejecución

**11**. **Explica qué es un programa y un proceso, realizando una comparación de ambos.**

| **PROGRAMA** | **PROCESO** |
| --- | --- |
| Es estático, sólo es el código escrito | Está en ejecución y usa recursos del sistema |
| Se guarda en dispositivos de almacenamiento como SSD o Discos duros | Se carga en la memoria RAM |
| Sigue existiendo hasta que se elimine del disco | Solo existe mientras se esté ejecutando |

**12. ¿Para qué usa el sistema el CP?**

Indica la dirección de memoria de la próxima instrucción que el procesador debe ejecutar

**13. ¿Para qué sirven las llamadas al sistema?**

Permiten a los procesos en modo usuario solicitar servicios del sistema operativo

**14**. **¿Qué relación existe entre una tarea y un trabajo?**

Un trabajo puede estar compuesto por una o varias tareas

**15. ¿Por qué es más liviano el cambio de contexto entre dos hilos de un mismo proceso?**

Porque los hilos comparten la misma memoria y recursos, así el sistema no necesita cambiar la imagen completa del proceso

**16. ¿De qué dos maneras, vía terminal, podemos revisar los trabajos?**

Podemos usar el comando jobs y el comando ps

**17. ¿Qué muestra el comando top -u usuario?**

Muestra los procesos del usuario en tiempo real

**Crea una tabla con los siguientes campos:**

| **Proceso** | **S.O** | **Función** | **Esencial** | **Justificación** |
| --- | --- | --- | --- | --- |
| **Init/systemd** | Linux | Es el primer proceso que se inicia durante el arranque del sistema y es responsable de iniciar y gestionar todos los demás servicios y procesos | Si | Sin él, el sistema no podría funcionar |
| **Bash** | Linux | Permite realizar tareas repetitivas, tantas veces como desees | NO | Aunque muy útil, no es esencial para el funcionamiento del sistema. |
| **Cron** | Linux | Permite ejecutar tareas programadas | NO | El sistema puede funcionar sin él. |
| **Syslog** | Linux | Daemon que gestiona los registros del sistema, permitiendo el almacenamiento y visualización de logs de eventos y errores | NO | Aunque es muy útil para la administración y solución de problemas, no es crítico para el funcionamiento del sistema |
| **Explorer.exe** | Win | Sirve como shell para el escritorio, la barra de tareas y el menú Inicio, y también administra la organización y visualización de archivos y carpetas | SI | Le ayuda al sistema a obtener los archivos que necesita de forma rápida y sencilla |
| **Svchost.exe** | Linux | Este proceso es esencial ya que permite que muchos servicios de Windows se ejecuten en modo compartido | SI | Optimiza el uso de recursos del sistema |
| **Services.exe** | Win | Este proceso es el gestor de servicios en Windows, encargado de iniciar, detener y administrar los servicios del sistema | SI | Sin Services.exe, los servicios no funcionarán correctamente, lo que afectaría negativamente al sistema operativo. |
| **Winlogon.exe** | Win | Maneja las sesiones de usuario, incluyendo el inicio y cierre de sesión. | SI | Es fundamental para la autenticación y seguridad del sistema. |