n esta práctica veremos algunos mandatos para la gestión de procesos. Aprenderemos a seguir el estado de ejecución de los procesos, a detener un proceso o a matarlo. También veremos cómo ejecutar un proceso en segundo plano y cómo intercambiarlo entre primero y segundo plano. Por último, aprenderemos a programar tareas con el mandato “cron”.

Durante el desarrollo de esta práctica elabora un informe en el que indiques los pasos dados en su realización, y contesta a las preguntas que se te van haciendo en el guión. Añade los pantallazos pertinentes para seguir la práctica. Entrega este informe a través del aula virtual de la asignatura. !

Guión de la práctica:

Una de las tareas más importantes del SO es la gestión de los procesos que se están ejecutando en una máquina. El hecho de que todos los procesos deban compartir los recursos hardware disponibles (memoria RAM, CPU) hace que el SO juegue un papel primordial en gestionar esos recursos para que los procesos se ejecuten de forma concurrente. Un proceso es la entidad que crea el SO para la ejecución de un programa. Cada proceso, durante su ejecución, guarda información sobre su “contexto” que incluye, entre otras cosas, información sobre su proceso “padre” (quién lo creó o invocó), los recursos del sistema que se están consumiendo

(segmentos de memoria asignados), permisos de seguridad, etc.

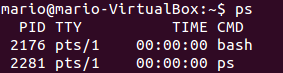
1. Visita el enlace http://structio.sourceforge.net/guias/AA\_Linux\_colegio/procesos-y-tareas.html y lee el punto 3.1 del mismo.

2. Inicia la máquina virtual Ubuntu.

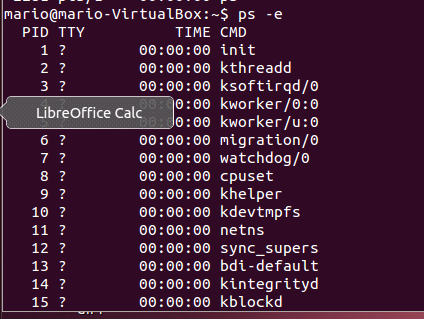
3. Lee la página del manual sobre el mandato “ps”.



4. Visualiza los procesos que se están ejecutando en este momento. El primer valor que aparece es el identificador de proceso (PID). El segundo es la terminal que está asociada a ese proceso. Después también podemos observar el tiempo acumulado de uso de CPU, y finalmente el nombre del programa que ha dado lugar a este proceso. Apunta los procesos activos y sus valores.



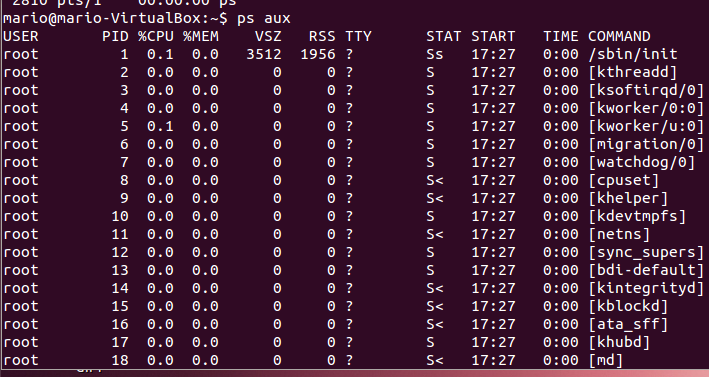
5. Por defecto, ps sólo ha mostrado los procesos asociados con la misma terminal e iniciados por el mismo usuario. Vamos a recuperar las opciones que nos permiten mostrar todos los procesos. Ejecuta el mandato “ps -e”. Comprueba la lista de procesos que están corriendo en tu máquina. ¿Cuál lleva el PID igual a 1?



6. Recuerda lo que sucedía al pulsar “Ctrl + Alt + F1” hasta “Ctrl + Alt + F6”; comprueba el nombre de cada una de esas terminales. Vuelve a la terminal en la que se encuentra el entorno gráfico (Ctrl + Alt + F7). ¿Qué procesos se están ejecutando en las terminales tty1, tty2...tty6?¿Qué proceso está corriendo sobre tty7?

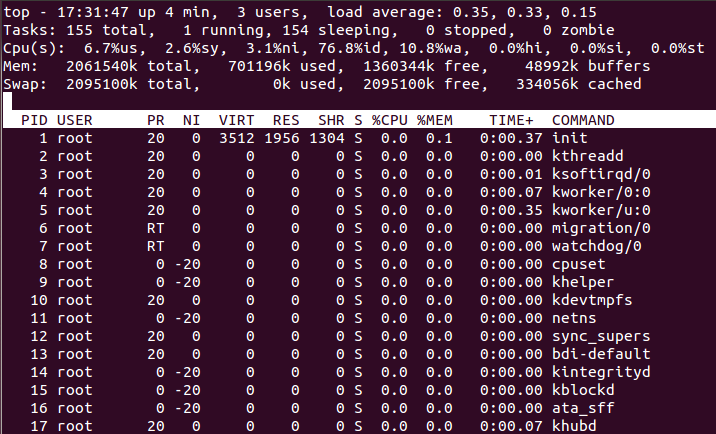
Todos los terminales anteriores

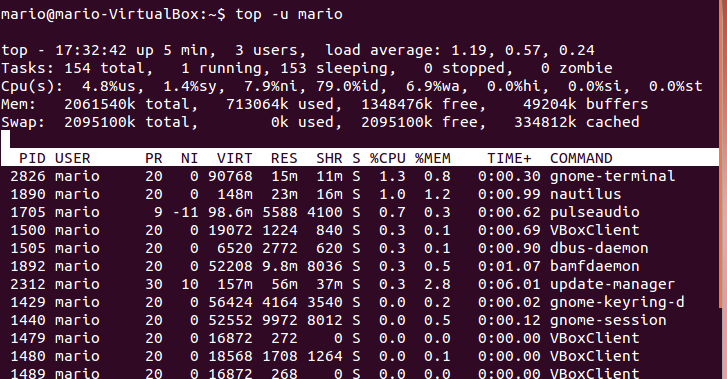
7. El mandato “ps” todavía nos puede ofrecer más información sobre los procesos en ejecución. Por ejemplo, ¿quién ha iniciado cada uno de los procesos en nuestra máquina? Vamos a usar las siguientes opciones del mandato ps: “a” nos permite conocer todos los procesos que tienen una terminal asociada; “x” aquellos que no tienen terminal; la opción “u” nos muestra la salida en un formato más legible. (ps aux)



9. Todos los mandatos y opciones que hemos visto hasta ahora ofrecían información estática sobre los procesos. Esta información se extrae del directorio “/proc” del sistema. Hay algunas aplicaciones que también nos permiten conocer en tiempo real las características de cada proceso. Por ejemplo, el mandato “top”.

10. Lee la página del manual sobre el mandato “top”. ¿Qué hace el mandato “top”? Ejecútalo. (top) Como puedes observar, la información sobre el sistema se refresca cada 3 segundos (se puede modificar ese parámetro). Por lo demás, la interfaz de usuario de “top” no es especialmente agradable, aunque sí resulta sencillo modificar ciertas opciones y ajustarla a nuestros requisitos. Pulsa “q” para salir de top.

11. Ejecuta ahora el mandato “top -u alumno”. ¿Cómo ha cambiado la salida del mandato? ¿A quién pertenecen los procesos que observas ahora?



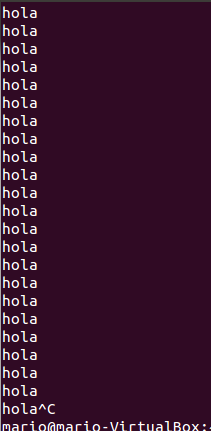
16. Vuelve a tu directorio personal (cd $HOME, cd ~, cd /home/alumno, ...) Veremos ahora algunos atajos de teclado que nos permiten gestionar procesos.



17. Lee la página del manual sobre el comando “yes”. Aunque el mismo pueda no parecer de gran utilidad, a nosotros nos va a servir para comprobar cómo podemos detener y “matar” procesos.



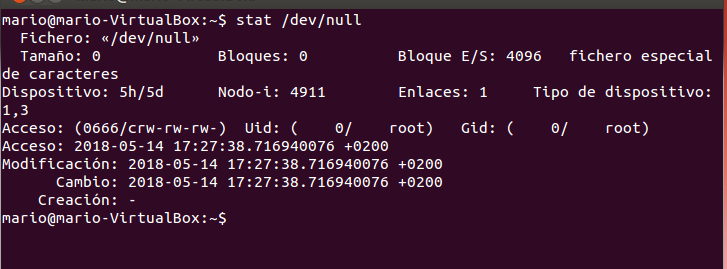
18. Ejecuta el mandato “yes hola”. Observa que el mensaje aparece indefinidamente. Vamos a “matar” esta tarea. Intenta salir de la tarea con “q”. La tecla “q” (quit, salir) nos permite salir de ciertas aplicaciones en ejecución, pero no acabar con una tarea. Teclea el atajo “Ctrl + C”. El mismo debería terminar con el proceso activo. ¿Qué ha sucedido? El atajo de teclado “Ctrl + C” se encarga de terminar (o matar) una tarea. El atajo “Ctrl + Z” se encarga únicamente de detenerla (aunque el proceso siga “vivo” y se pueda retomar en el estado en que se detuvo).



19. Podemos ahora redirigir la salida del mandato a un fichero (observa que esto podría darnos serios problemas de memoria en nuestra máquina). Para poder redirigir la salida de mandatos a un fichero sin peligro de que eso colapse nuestra memoria, Linux dispone de un fichero cuya localización es “/dev/null”. Busca en la Wikipedia “/dev/null” para comprender mejor qué es y ejecuta el mandato: (yes adios > /dev/null)



20. Como puedes observar, la tarea en ejecución no permite seguir utilizando la terminal. Teclea “Ctrl + C” para detenerlo. ¿Qué tamaño ocupa ahora en disco el fichero /dev/null? ¿Dónde ha ido a parar toda la información que hemos enviado? (stat /dev/null)

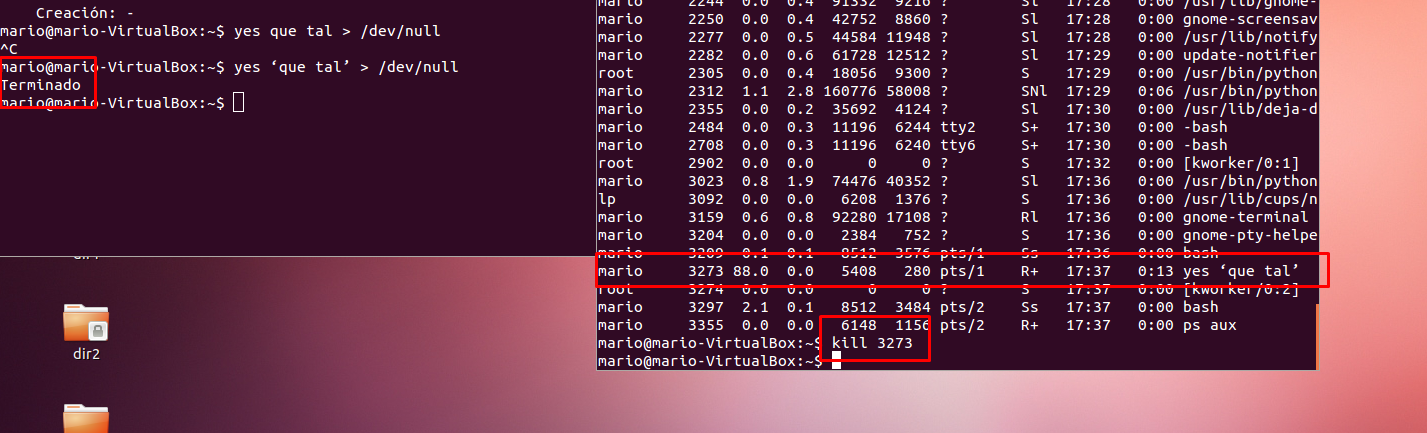


21. Ejecutamos de nuevo el mandato “yes”. (yes ‘que tal’ > /dev/null. Abre una nueva terminal y localiza el PID del proceso “yes” iniciado.



22. Lee la página del manual sobre el mandato “kill”. Como puedes observar, kill nos permite mandar señales a un proceso. El tipo de señales que permite mandar lo puedes encontrar, por ejemplo, en http://tldp.org/LDP/Bash-Beginners-Guide/html/sect\_12\_01.html Lee las Secciones 12.1.1.3 y 12.1.2 del anterior enlace. Apunta algunas de las señales más comunes que se pueden enviar a procesos en ejecución y la forma de hacerlo (por teclado, o por medio de kill). Anota la diferencia entre SIGKILL y SIGTERM.

23. Interrumpe el proceso activo “yes” por medio del mandato “kill”



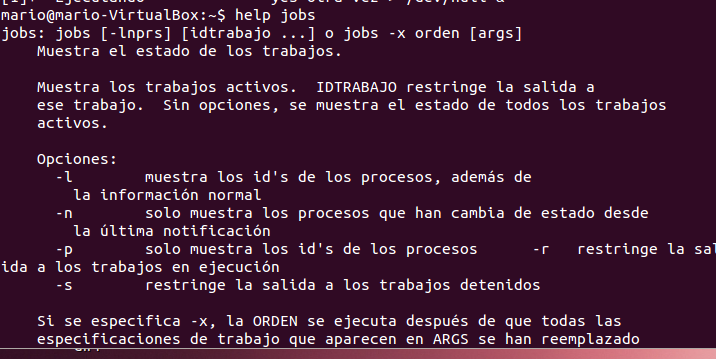
24. En nuestra terminal original vuelve a ejecutar el proceso “yes que tal > /dev/null” y, desde la segunda terminal que hemos abierto, envíale ahora una señal de SIGKILL. Comprueba que el resultado externo ha sido el mismo que antes.

**No me encuentra la orden**

25. todos los procesos que ejecutamos en una terminal deben ejecutarse en primer plano (bloqueando así la terminal). También podemos hacer lo que se conoce como ejecución en segundo plano. Puedes leer en UD8 información acerca de las principales diferencias entre ejecutar un proceso en primero o segundo plano (esencialmente tiene que ver con la prioridad del mismo). La forma de hacer que un programa se ejecute en segundo plano es escribiendo el programa en el intérprete de mandatos seguido de un símbolo “&”. Comprueba, en nuestra terminal original, el siguiente mandato:(yes otra vez > /dev/null &)



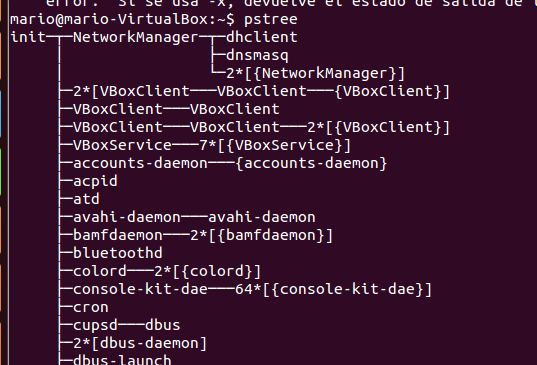
26. Vamos a hacer ahora uso del mandato “jobs”. Comprueba en primer lugar su función por medio de “help jobs”, y el significado de la opción “-l”. La diferencia entre una “tarea” (job) y un proceso (process) es que los “jobs” son obligatoriamente iniciados desde una terminal y están asociados a ella (son procesos “hijos” de la terminal).



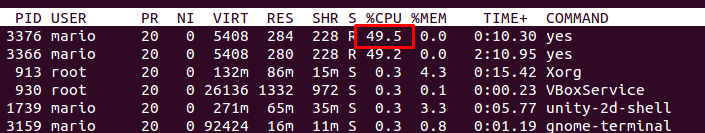
27. Ejecuta el mandato “jobs” en la misma terminal en la que has ejecutado “yes”. ¿En qué estado se encuentra el proceso? Compruébalo también con los mandatos “ps” y “top”. Anota el porcentaje de CPU que consume.



28. Entre los procesos de Linux siempre existe una jerarquía definida, ya que cada proceso debe tener un proceso padre (excepto el proceso de inicio, llamado init). Esta jerarquía adquiere relevancia ya que “matar” a un proceso padre por lo general conlleva acabar también con los procesos hijos. En algunos casos, un proceso padre y sus hijos pueden incluso compartir memoria. Comprueba la jerarquía de procesos en tu máquina por medio del mandato pstree (puedes ver alguna de sus opciones en man pstree). Comprueba sus ancestros. Apunta en tu informe de qué procesos desciende “yes”.



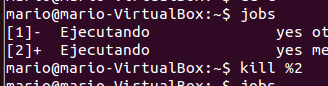
29. Vuelve a ejecutar “yes” en la misma terminal y también en segundo plano. (yes mensaje > / dev/null &) Vuelve a comprobar el árbol de procesos por medio de “pstree -h”. Usando el mandato “top”, anota en tu informe el porcentaje de CPU (aproximado) que suman estos dos procesos yes.



30. Vuelve a comprobar el estado de los procesos iniciados en esta shell por medio de “jobs”. Comprueba que aparecen las dos tareas iniciadas y que en la segunda aparece el símbolo + indicando que es la última que se ha ejecutado. Apunta el estado de ambas en tu informe.

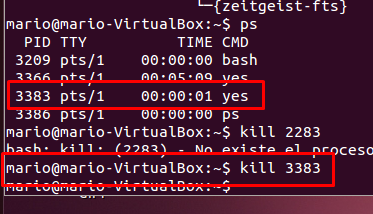


32. A través de “jobs”, cada tarea que se está ejecutando desde nuestra terminal recibe un nuevo número (1, 2, ...). Este número aparece entre corchetes. Mata la segunda tarea iniciada (la de mayor PID). Por ejemplo, puedes ejecutar top y capturar su PID y enviarle una señal de kill (en lugar del PID también puedes usar el símbolo “%” seguido del número de tarea). Comprueba por medio de jobs que sólo queda una tarea activa.

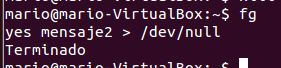




33. Ejecuta el mandato “yes mensaje2 > /dev/null”. Por medio del teclado (Ctrl + Z), o por medio de kill (con la señal SIGSTOP ó 19 y con el PID correspondiente) envía al proceso una señal de “detenido”. Comprueba que el proceso está detenido por medio de jobs.

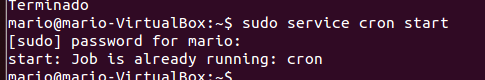


34. Los números de tarea pueden ser usados por el mandato “fg” (foreground) para traer dichas tareas a primer plano (fg 1, fg 2, ...), o por el mandato “bg” para mandarlas a segundo plano (bg 1, bg 2, ...). Para recuperar una tarea detenida sólo tienes que ejecutar fg (foreground) o bg (background), dependiendo de que quieras que la tarea se ejecute en primer o segundo plano. Comprueba con fg que la tarea vuelve a primer plano.



38.Linux también dispone de utilidades para la programación de tareas; las tareas programadas son procesos que se ejecutarán (siempre y cuando la máquina esté encendida) de forma planificada. El programa que nos permite programar tareas desde línea de mandatos se llama “cron”. Puedes leer en la UD8 información sobre el cron y crontab.

39.Comprueba que “cron” está activo en tu ordenador (puedes observar si aparece en pstree). El proceso “cron” se debe encontrar siempre en ejecución, para que a la hora y día que tenga programada alguna tarea pueda ejecutar la misma. También te puedes asegurar de que el mismo está en marcha por medio del mandato: (sudo service cron start) (También con sudo start cron)

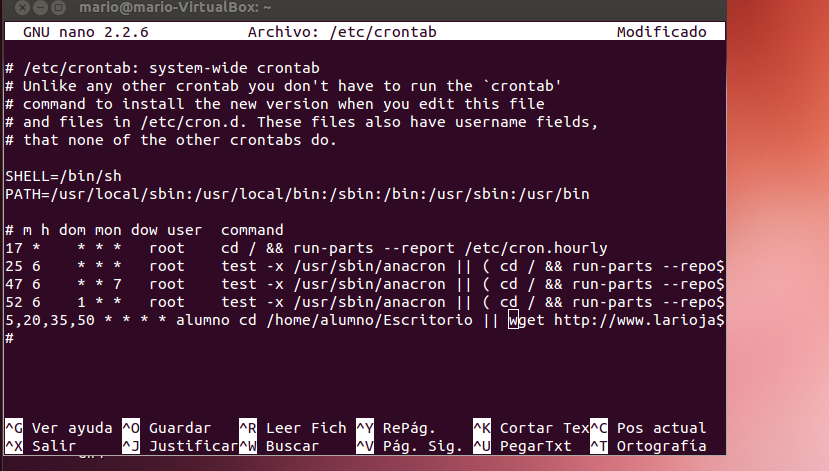


40.Existen diversas formas de programar nuevas tareas. El fichero en el que se encuentran las tareas programadas se llama “/etc/crontab”. Lee la página del manual sobre el fichero “/etc/ crontab”. (man 5 crontab)



41.El fichero “crontrab” es un fichero de texto, así que puedes editarlo y modificarlo, por ejemplo, con nano (pero deberás disponer de permisos de superusuario). Ejecuta: (sudo nano /etc/crontab)Por defecto deberían aparecer varias tareas programadas del sistema. La estructura de cada una de las líneas es la siguiente:

minuto(s) - hora - día del mes – mes - día de la semana – usuario - mandato De esta forma, la siguiente línea: 5,20,35,50 \* \* \* \* alumno cd /home/alumno/Escritorio;wget http://www.larioja.com significará que a los minutos 5, 20, 35, 50 de todas las horas (\*), de todos los días del mes (\*), de todos los meses y de todos los días de la semana (\*), el usuario “alumno” ejecutará los mandatos “cd /home/alumno/Escritorio; wget http://www.larioja.com”. Inserta la línea anterior en la última línea de tu fichero crontab, pero de forma que la acción se ejecute cada cinco minutos. (Escribe \*/5)



43. ¿Cuál es la diferencia entre el comando AT y utilizar cron/crontab?

La diferencia entre AT y Cron es que el primero no es persistente, por lo que si reiniciamos la PC se perderá la tarea que le encomendamos. ¿Cómo funciona AT? Pues muy sencillo, la forma básica sería escribir en el terminal