

Programación y Administración de Sistemas

5. Gestión de los recursos del sistema

Pedro Antonio Gutiérrez

Asignatura “Programación y Administración de Sistemas”
2º Curso Grado en Ingeniería Informática
Escuela Politécnica Superior
(Universidad de Córdoba)
pagutierrez@uco.es

1 de abril de 2018



Objetivos del aprendizaje I

- Conocer cuáles son los recursos básicos del sistema operativo (CPU, memoria y espacio en disco) y la necesidad e importancia de su monitorización y gestión.
- Definir los procesos en GNU/Linux y distinguir los modos de ejecución básicos (modo usuario y modo núcleo).
- Establecer los tipos de procesos que pueden ejecutarse en el sistema operativo.
- Utilizar la herramienta `ps` para ver los procesos en ejecución y sus atributos.
- Explicar el ciclo de vida de un proceso en GNU/Linux desde un punto de vista de su administración, detallando los distintos estados por los que puede pasar.
- Conocer el mecanismo de planificación utilizado en GNU/Linux para ejecutar los procesos.

Objetivos del aprendizaje II

- Utilizar el número *nice* para modificar la prioridad de los procesos.
- Enviar señales a procesos para controlar su ejecución y distinguir entre el efecto de las distintas señales.
- Monitorizar el tiempo de actividad de un sistema mediante la herramienta *uptime*.
- Monitorizar el árbol de ejecución de procesos de un sistema mediante la herramienta *pstree*.
- Monitorizar los procesos en ejecución de forma interactiva mediante la herramienta *top*.
- Obtener informes sobre la ejecución de procesos en un sistema mediante la herramienta *vmstat*.
- Conocer el contenido de la carpeta */proc* y los ficheros que en ella aparecen para cada uno de los procesos en ejecución.

Objetivos del aprendizaje III

- Postergar la ejecución de procesos mediante el uso de la herramienta `at`.
- Planificar la ejecución periódica de procesos mediante la herramienta `cron`.
- Rastrear señales y llamadas al sistema de un determinado proceso mediante la herramienta `strace`.
- Monitorizar la cantidad de memoria libre mediante la herramienta `free`.
- Monitorizar el uso de memoria mediante `vmstat`.
- Decidir el espacio de paginación necesario para un sistema operativo.
- Controlar el espacio en disco mediante las herramientas `df` y `du`.
- Monitorizar el rendimiento de los discos mediante el uso de la herramienta `iostat`.

5.1. Introducción.

5.2. Actividad de la CPU.

5.2.1. Procesos en GNU/Linux.

5.2.1.1. Modo de ejecución.

5.2.1.2. Tipos de procesos.

5.2.1.3. Herramienta `ps`.

5.2.1.4. Estados de los procesos.

5.2.2. Prioridad y señales.

5.2.2.1. Número *nice* y prioridad de procesos.

5.2.2.2. Envío de señales a procesos (`kill`, `killall`).

5.2.2.3. Señales más habituales y comportamientos por defecto.

5.2.3. Monitorizar uso CPU.

5.2.3.1. Monitorización de tiempo de actividad mediante `uptime`.

5.2.3.2. Monitorización de procesos mediante `pstree`.

5.2.3.3. Monitorización de procesos mediante `top`.

5.2.3.4. Monitorización de actividad de CPU mediante `vmstat`.

5.2.3.5. La carpeta /proc.

5.2.4. Programar ejecución de procesos.

5.2.4.1. Ejecución de tareas aplazadas mediante la herramienta `at`.

5.2.4.2. Ejecución de tareas periódicas mediante la herramienta `cron`.

5.2.5. Rastreo de señales y llamadas al sistema.

5.3. Memoria.

5.3.1. Monitorización de memoria libre mediante `vmstat` y `free`.

5.3.2. Espacio para paginación.

5.4. Dispositivos Entrada/Salida.

5.4.1. Monitorización de espacio en disco mediante `df` y `du`.

5.4.2. Monitorización de acceso a disco mediante `iostat`.

- Cuestionarios objetivos.
- Pruebas de respuesta libre.
- Tareas de administración.

Introducción

- Una **correcta administración** del sistema implica obtener información sobre sus **recursos** y rendimiento:
 - Procesos en ejecución, cantidad de memoria disponible, espacio en disco, n° de particiones, prioridad de procesos, etc.
- Si hubiese algún problema de rendimiento, deberíamos:
 - 1 Definir el problema con todo el detalle que sea posible.
 - 2 Determinar la causa o causas del mismo.
 - 3 Formular los objetivos para mejorar el rendimiento.
 - 4 Diseñar e implementar modificaciones para llevar a cabo los objetivos de mejora.
 - 5 Monitorizar el sistema para determinar si los cambios han sido efectivos.



Procesos

- **Proceso**: programa en ejecución (el SO crea el proceso cuando comienza la ejecución y lo elimina al finalizarla).
- Un sistema de tiempo compartido como GNU/Linux permite **múltiples usuarios** que ejecuten **múltiples procesos**, pero la CPU solo puede ejecutar **un proceso a la vez**.
- La CPU conmuta rápidamente de un proceso al siguiente, ejecutando un **cuanto** (por ejemplo, 100ms) de cada proceso.
- El SO es el encargado de decidir qué proceso se ejecuta en qué lugar → **planificación** de la CPU.



Procesos: modo de ejecución

- Modos de ejecución (distinción para proteger mejor las direcciones de memoria a las que puede acceder un proceso)
 - Modo **usuario**: se ejecuta **código normal** del programa.
 - Modo **núcleo**: se ejecutan las funciones del **núcleo** (en realidad, es el *kernel* ejecutándose en nombre del proceso):
 - 1 **Llamadas al sistema**: Los procesos de usuario solicitan servicios explícitamente a través de la interfaz de llamadas al sistema (p.ej. crear un hilo).
 - 2 **Excepciones**: Situaciones excepcionales (división por cero, errores de direccionamiento...) causan excepciones *hardware* que requieren intervención del *kernel*.
 - 3 **Interrupciones**: Los dispositivos periféricos interrumpen para notificar al *kernel* de diversos sucesos (terminación de E/S, cambio de estado...).



Procesos: tipos de procesos I

Procesos de usuario

- Procesos creados por un usuario real.
- Se ejecutan en modo usuario, excepto en los casos anteriores.

Procesos demonio

- No asociados a un usuario, o asociados a uno ficticio.
- Se ejecutan en modo usuario, excepto en los casos anteriores.
- Realizan tareas periódicas relacionadas con la administración del sistema (gestión de la red, crontab...).



Procesos: tipos de procesos II

Procesos núcleo

- No asociados a un usuario.
- Corresponden al código del *kernel*.
- Se ejecutan siempre en modo núcleo.
- Tareas de administración más delicadas (planificación, intercambio de procesos, intercambio de páginas...).



Procesos: monitorizar con ps

- **ps**: información sobre los procesos en ejecución
 - **USER** ⇒ usuario que lanzó el programa.
 - **PID** ⇒ identificador del proceso.
 - **PPID** ⇒ identificador del proceso padre (los nuevos procesos se crean clonándose con `fork`).
 - **%CPU** ⇒ porcentaje de la CPU consumido por este proceso (en ese momento).
 - **%MEM** ⇒ fracción de memoria consumida (es una estimación).
 - **VSZ** ⇒ tamaño virtual (código+datos+pila) en KB.
 - **RSS** ⇒ memoria real usada en KB (VSZ incluye a RSS).
 - **TTY** ⇒ terminal asociado con el proceso.



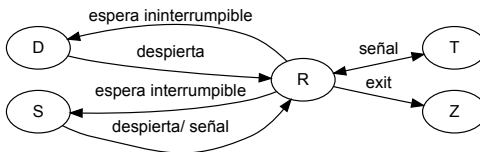
Procesos: monitorizar con ps

- **ps**: información sobre los procesos en ejecución
 - STAT \Rightarrow estado del proceso.

| | | |
|---------------------------------------|--------------------------------|---|
| R: en ejecución | N: prioridad baja (> 0) | L: tiene páginas bloqueadas en memoria |
| S: durmiendo | <:prioridad alta (< 0) | s: líder de sesión |
| T: parado (señal o trace) | | 1: tiene <i>multithread</i> |
| Z: proceso <i>zombie</i> | | +: proceso <i>foreground</i> |
| D: durmiendo ininterrumpible (E/S) | | |



Procesos: estados de los procesos



- R: un proceso en ejecución está listo para ejecutarse en cuanto la CPU esté libre. Tiene todos los recursos que necesita y está esperando su *cuanto* para ejecutarse.
- S: durmiendo, esperando a que ocurra un evento específico (petición I/O, lectura de un *socket*...). `bash` y los demonios del sistema pasan casi todo su tiempo durmiendo, esperando la entrada por terminal o que un cliente haga una petición por la red. Estos procesos no recibirán tiempo de CPU hasta que el evento ocurra o que se reciba una señal específica.



Procesos: estados de los procesos

- s: líder de sesión. Los procesos se pueden agrupar. Si se manda una señal al grupo, se le manda a todos los procesos. El líder es el que interactúa con la terminal.
- 1: hilos creados con `CLONE_THREAD` (p.ej. hilos *Native Posix Thread Library*, NPTL).
- L: el proceso ha pedido al *kernel* bloquear determinadas páginas de memoria, para evitar que no se modifiquen mientras se hacen determinadas operaciones.
- +: *foreground*, proceso de primer plano, iniciado sin `&`.



```

1 pagutierrez@TOSHIBA:~$ ps aux | less #a-> Todos usuarios, x-> Procesos sin
   terminal, u -> Añadir nombre de usuario
2 USER  PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
3 root    1  0.0  0.1  24124  1160 ?        Ss   09:22   0:01 /sbin/init
4 root    2  0.0  0.0      0      0 ?        S    09:22   0:00 [kthreadd]
5 root    3  0.0  0.0      0      0 ?        S    09:22   0:00 [ksoftirqd/0]
6 root  1016  0.0  0.1  76188  1248 ?        Ss   09:22   0:00 /usr/sbin/cupsd -F
7 root  1032  0.0  0.0   6200   332 tty4      Ss+  09:22   0:00 /sbin/getty -8 34 tty4
8 1000  1558  0.5  1.4 407856 14804 ?        Sl   09:22   0:22 gnome-panel
9 1000  4893  0.0  0.1  21948  1380 pts/3    R+   10:30   0:00 ps aux
10 ...

```

```

1 pagutierrez@TOSHIBA:~$ ps al #a-> Todos usuarios, l -> Formato "long"
2 F  UID  PID PPID PRI NI    VSZ   RSS WCHAN  STAT TTY      TIME COMMAND
3  4      0 1001  959  20  0 140016 15628 poll_s Ss+  tty7  25:20 /usr/bin/X :0 -nr -
   verbose -auth /var/run/gdm/auth-for-gdm-puS6qe/database -nolisten tcp vt7
4  0      0 1032    1  20  0   6200   332 n_tty_ Ss+  tty4   0:00 /sbin/getty -8 tty4
5  0      0 1036    1  20  0   6200   328 n_tty_ Ss+  tty5   0:00 /sbin/getty -8 tty5
6  0      0 1049    1  20  0   6200   332 n_tty_ Ss+  tty2   0:00 /sbin/getty -8 tty2
7  0      0 1053    1  20  0   6200   332 n_tty_ Ss+  tty3   0:00 /sbin/getty -8 tty3
8  0      0 1061    1  20  0   6200   332 n_tty_ Ss+  tty6   0:00 /sbin/getty -8 tty6
9  0      0 1414    1  20  0   6200   332 n_tty_ Ss+  tty1   0:00 /sbin/getty -8 tty1
10 0 1000  4685 4666  20  0  27556   892 n_tty_ Ss+  pts/2   0:01 /bin/bash
11 0 1000  4777 2489  20  0  27556  1800 wait   Ss   pts/3   0:00 bash
12 0 1000  5118 4777  20  0  13516  1136 -      R+   pts/3   0:00 ps al

```



Control/gestión de la actividad de la CPU

- Número *nice* (“buena gente”) y prioridad de procesos:
 - Planificación de procesos por prioridades dinámicas.
 - Al lanzar el proceso, se le asigna un **número *nice*** o prioridad estática (se hereda por defecto del proceso padre).
 - La prioridad por defecto se obtiene mediante el número ***nice***.
 - Valores bajos (negativos): más prioridad.
 - Valores altos (positivos): menos prioridad.
 - Rango de prioridad estática $\Rightarrow [-20, 19]$
 - Asignación de prioridades mayores o menores que la actual.
 - `nice -5 nautilus`: lanzar nautilus con nº *nice* incrementado en 5.
 - `nice --10 nautilus`: lanzar nautilus con nº *nice* decrementado en 10 (solo **root**).
 - `renice 14 890`: prioridad 14 al proceso 890.
 - `renice 5 -u pedroa`: prioridad 5 para todos los procesos del usuario pedroa.



Control/gestión de la actividad de la CPU

- Envío de señales a los procesos (pararlos, hacer que continúen, eliminarlos...):
 - `kill -señal pid` (donde señal es un número).
 - `kill pid`: mandar señal por defecto al proceso `pid` (señal SIGTERM, número 15, [se puede capturar](#)).
 - SIGKILL (9) fuerza la salida del proceso. [No se puede capturar](#).
 - Parar un proceso SIGSTOP (19), Reiniciarlo SIGCONT (18).
 - [killall comando](#): permite mandar una señal a todos los procesos con un determinado nombre de comando.
 - [pkill](#) ó [skill](#) ⇒ enviar una señal usando el nombre u otros atributos o criterios (`uid`, `gid`, `terminal...`).
 - Los procesos en estado D o Z no se detienen pese a recibir la señal KILL.



Control/gestión de la actividad de la CPU

| # | Nombre | Descripción | Por defecto | ¿Se puede capturar? | ¿Se puede bloquear? | ¿core dump? |
|----|--------|-----------------------|-------------|---------------------|---------------------|-------------|
| 1 | HUP | Hang up (terminal) | Terminar | Si | Si | No |
| 2 | INT | Interrumpir (Ctrl+C) | Terminar | Si | Si | No |
| 3 | QUIT | Similar a TERM | Terminar | Si | Si | Si |
| 9 | KILL | Matar proceso | Terminar | No | No | No |
| * | BUS | Error manejo bus | Terminar | Si | Si | Si |
| 11 | SEGV | Violación de segmento | Terminar | Si | Si | Si |
| 15 | TERM | Parar <i>software</i> | Terminar | Si | Si | No |
| * | STOP | Parada | Parar | No | No | No |
| * | TSTP | Parada (Ctrl+Z) | Parar | Si | Si | No |
| * | CONT | Continuar (tras STOP) | Ignorar | Si | No | No |
| * | WINCH | Cambio tamaño | Ignorar | Si | Si | No |
| * | USR1 | A definir | Terminar | Si | Si | No |
| * | USR2 | A definir | Terminar | Si | Si | No |

*: depende del Sistema Operativo.



Control/gestión de la actividad de la CPU

- KILL (1): No se puede bloquear ni capturar.
- INT (2): La que se envía al pulsar Crt1+C.
 - Se puede bloquear.
 - Si se manda a un intérprete de órdenes, podría cancelar la orden que está ejecutando, pero no el programa completo.
- TERM (15): La que se manda al cerrar el proceso padre o al reiniciar. Se puede bloquear y capturar.
- Diferencia entre STOP y TSP: STOP no se puede ni bloquear ni capturar.



Control/gestión de la actividad de la CPU

- HUP (1):
 - Si se trata de demonios, debería provocar que se reinicien, volviendo a leer su configuración.
 - Si se trata de procesos iniciados en una terminal, se manda al cerrar la terminal (algunos intérpretes hacen inmunes los procesos *background* a esta señal, en *bash*, hay que hacerlo con el comando *nohup*).
- QUIT (3): Similar a TERM pero hace un *core dump*.
- TSTP: La que se envía al pulsar Crt1+Z.
- Los procesos detenidos con TSTP o con STOP, se puede reanudar con:
 - La señal CONT.
 - usando el comando *fg* (vuelve al *foreground*) o *bg*.



Control/gestión de la actividad de la CPU

- **uptime**: hora actual, cuánto tiempo lleva en marcha el sistema, número de usuarios conectados, y carga media del sistema (el número medio de procesos del sistema que durante los últimos 1, 5 y 15 minutos han estado en los estados R o D).
 - Valores altos implican que el sistema se está usando mucho, pero ¿cuándo se considera que un valor es alto? → depende del número de núcleos.
 - Valores bajos no significan que el tiempo de respuesta vaya a ser bajo.

```
1 pagutierrez@TOSHIBA:~$ uptime
2 13:31:05 up 1:32, 3 users, load average: 0.18, 0.19, 0.19
```



Control/gestión de la actividad de la CPU

- **ps-tree** ⇒ visualiza un árbol de los procesos en ejecución

```
1  init---NetworkManager---2*[{NetworkManager}]
2      |--acpid
3      |--atd
4      |--avahi-daemon---avahi-daemon
5      |--bamfd daemon
6      |--bluetoothd
7      |--bonobo-activati---2*[{bonobo-activat}]
8      |--console-kit-dae---64*[{console-kit-da}]
9      |--cron
10     |--cupsd
11     |--2*[{dbus-daemon}]
12     |--dbus-launch
13     |--dconf-service---{dconf-service}
14     |--evince---3*[{evince}]
15     |--evince---{evince}
16     |--firefox---plugin-containe---7*[{plugin-contain}]
17     |         |--21*[{firefox}]
18     |--gconfd-2
```



Control/gestión de la actividad de la CPU

- **top**: proporciona una visión continua de la actividad del procesador, en tiempo real, mostrando las tareas que hacen más uso de la CPU. Además, permite manipular procesos de forma **interactiva**.
 - Las cinco líneas primeras muestran información general:
 - Estadísticas **uptime**.
 - Resumen de procesos en el sistema: n° procesos, n° procesos en ejecución, durmiendo, parados o zombies.
 - Porcentaje de tiempo de CPU gastado en: modo usuario (**us**), modo sistema o núcleo (**sy**), procesos valor **nice** positivo (**ni**), procesos ociosos (**id**), procesos esperando eventos E/S (**wa**), tratando interrupciones (*hardware* o *software*, **hi** o **si**), espera involuntaria en virtualización (**st**).
 - Estado actual de la memoria física: total disponible, usada, libre, usada en *buffers*.
 - Espacio swap: total disponible, usada, libre, usada en *buffers*, usada en caché de página.



```

1 top - 11:33:17 up 2:11, 4 users, load average: 0.12, 0.19, 0.35
2 Tasks: 183 total, 1 running, 181 sleeping, 0 stopped, 1 zombie
3 Cpu(s): 6.9%us, 2.6%sy, 0.0%ni, 89.8%id, 0.8%wa, 0.0%hi, 0.0%si, 0.0%st
4 Mem: 1012004k total, 970028k used, 41976k free, 8444k buffers
5 Swap: 2080760k total, 507884k used, 1572876k free, 278284k cached
6
7 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
8 1001 root 20 0 170m 17m 4496 S 12 1.7 30:23.62 Xorg
9 2489 pagutier 20 0 335m 11m 6200 S 9 1.1 0:50.35 gnome-terminal
10 1545 pagutier 20 0 304m 7240 4112 S 6 0.7 4:09.81 compiz
11 2208 pagutier 20 0 392m 10m 5784 S 4 1.0 4:48.32 plugin-containe
12 2148 pagutier 20 0 881m 208m 15m S 3 21.1 7:50.88 firefox
13 5549 pagutier 20 0 763m 22m 12m S 2 2.3 0:21.16 knotify4
14 5677 root 20 0 0 0 0 S 0 0.0 0:00.06 kworker/2:1
15 5693 pagutier 20 0 19460 1500 1060 R 0 0.1 0:00.12 top
16 1565 pagutier 20 0 320m 6080 4636 S 0 0.6 0:05.66 gnome-power-man
17 1611 pagutier 20 0 534m 10m 4932 S 0 1.0 0:38.68 dropbox
18 1917 pagutier 20 0 360m 6856 3688 S 0 0.7 0:28.67 ubuntuone-syncd
19 4765 pagutier 20 0 585m 40m 8980 S 0 4.1 0:52.57 evince
20 197 root 20 0 0 0 0 S 0 0.0 0:02.40 usb-storage
21 457 messageb 20 0 24892 1684 668 S 0 0.2 0:09.26 dbus-daemon
22 507 avahi 20 0 32404 1152 800 S 0 0.1 0:09.01 avahi-daemon
23 513 root 20 0 162m 2900 2292 S 0 0.3 0:04.79 NetworkManager
24 1086 root 20 0 15784 448 364 S 0 0.0 0:03.76 irqbalance
25 1478 pagutier 20 0 238m 3908 3144 S 0 0.4 0:01.42 gnome-session
26 1515 pagutier 20 0 26708 2032 560 S 0 0.2 0:12.53 dbus-daemon
27 1531 pagutier 20 0 464m 5392 4264 S 0 0.5 0:20.41 gnome-settings-
28 1556 pagutier 20 0 671m 12m 6984 S 0 1.3 4:44.00 nautilus
29 1558 pagutier 20 0 398m 9304 5460 S 0 0.9 0:30.69 gnome-panel
30 1559 pagutier 20 0 398m 8876 5552 S 0 0.9 0:09.11 nm-applet

```



Control/gestión de la actividad de la CPU

- **top**:
 - Los datos de la parte inferior son similares a los de **ps**, excepto:
 - **SHR**: memoria compartida disponible para ser utilizada.
 - Procesos ordenados decrecientemente por uso de CPU.
 - Lista actualizada interactivamente, normalmente cada 5s.
 - Tareas sobre los procesos:
 - Cambiar la prioridad de alguno utilizando la opción "r".
 - Matar o enviar una señal con la opción "k".
 - Ordenarlos según diferentes criterios (por PID con "N", uso de CPU con "P", tiempo con "T", por memoria con "M", etc.).
 - Con "n" se cambia el número de procesos que se muestran.
 - Para salir se utiliza la letra "q".
 - "u" mostrar un usuario.
 - "R" cambiar ordenación.
 - "1" información independiente por cada procesador.
 - **htop**: similar pero con colores (también top + "z").



```

1 top - 11:33:17 up 2:11, 4 users, load average: 0.12, 0.19, 0.35
2 Tasks: 183 total, 1 running, 181 sleeping, 0 stopped, 1 zombie
3 Cpu(s): 6.9%us, 2.6%sy, 0.0%ni, 89.8%id, 0.8%wa, 0.0%hi, 0.0%si, 0.0%st
4 Mem: 1012004k total, 970028k used, 41976k free, 8444k buffers
5 Swap: 2080760k total, 507884k used, 1572876k free, 278284k cached
6
7 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
8 1001 root 20 0 170m 17m 4496 S 12 1.7 30:23.62 Xorg
9 2489 pagutier 20 0 335m 11m 6200 S 9 1.1 0:50.35 gnome-terminal
10 1545 pagutier 20 0 304m 7240 4112 S 6 0.7 4:09.81 compiz
11 2208 pagutier 20 0 392m 10m 5784 S 4 1.0 4:48.32 plugin-containe
12 2148 pagutier 20 0 881m 208m 15m S 3 21.1 7:50.88 firefox
13 5549 pagutier 20 0 763m 22m 12m S 2 2.3 0:21.16 knotify4
14 5677 root 20 0 0 0 0 S 0 0.0 0:00.06 kworker/2:1
15 5693 pagutier 20 0 19460 1500 1060 R 0 0.1 0:00.12 top
16 1565 pagutier 20 0 320m 6080 4636 S 0 0.6 0:05.66 gnome-power-man
17 1611 pagutier 20 0 534m 10m 4932 S 0 1.0 0:38.68 dropbox
18 1917 pagutier 20 0 360m 6856 3688 S 0 0.7 0:28.67 ubuntuone-syncd
19 4765 pagutier 20 0 585m 40m 8980 S 0 4.1 0:52.57 evince
20 197 root 20 0 0 0 0 S 0 0.0 0:02.40 usb-storage
21 457 messageb 20 0 24892 1684 668 S 0 0.2 0:09.26 dbus-daemon
22 507 avahi 20 0 32404 1152 800 S 0 0.1 0:09.01 avahi-daemon
23 513 root 20 0 162m 2900 2292 S 0 0.3 0:04.79 NetworkManager
24 1086 root 20 0 15784 448 364 S 0 0.0 0:03.76 irqbalance
25 1478 pagutier 20 0 238m 3908 3144 S 0 0.4 0:01.42 gnome-session
26 1515 pagutier 20 0 26708 2032 560 S 0 0.2 0:12.53 dbus-daemon
27 1531 pagutier 20 0 464m 5392 4264 S 0 0.5 0:20.41 gnome-settings-
28 1556 pagutier 20 0 671m 12m 6984 S 0 1.3 4:44.00 nautilus
29 1558 pagutier 20 0 398m 9304 5460 S 0 0.9 0:30.69 gnome-panel
30 1559 pagutier 20 0 398m 8876 5552 S 0 0.9 0:09.11 nm-applet

```



Control/gestión de la actividad de la CPU

- **vmstat**: información sobre memoria virtual (también procesos), desde el último reinicio:
 - **r** ⇒ número de procesos esperando su tiempo de ejecución
 - **b** ⇒ número de procesos en espera ininterrumpible
 - **us** ⇒ tiempo de usuario como porcentaje de tiempo total (modo usuario)
 - **sy** ⇒ tiempo de sistema como porcentaje de tiempo total (modo núcleo)
 - **id** ⇒ tiempo de inactividad como porcentaje de tiempo total
 - **wa** ⇒ tiempo usado en espera de E/S

```

1 pagutierrez@TOSHIBA:~$ vmstat 2 5 # 5 informes cada 2 segundos
2procs-----memory-----swap--  ---io---  -system--  ---cpu----
3 r  b      swpd    free    buff    cache    si   so    bi   bo    in    cs  us  sy  id  wa
4 0  0    465400    60300    15376   196640    11   25   140   58   213   319  10   3  83   4
5 0  0    465400    60236    15384   196628     0    0     0   10   865   658   3   2  95   1
6 0  0    465400    60096    15384   196716     0    0     0   50   858   704   4   1  95   0
7 0  0    465400    60128    15384   196672     0    0     0   0   787   657   3   2  95   0
8 0  0    465400    59384    15392   196744     0    0     0   38   787   907  13   2  85   0
    
```



Carpeta /proc

- `ps` y `top` leen la información que necesitan de `/proc`.
- Cada proceso tiene una carpeta (cuyo nombre es el `pid`) y en esa carpeta hay información sobre el mismo:
 - `cmdline`: línea de comandos con que fue iniciado.
 - `cwd`: enlace simbólico al directorio actual del proceso.
 - `environ`: Las variables de entorno en el momento de invocación.
 - `exe`: enlace simbólico al fichero ejecutado.
 - `fd`: carpeta con cualquier descriptor de fichero abierto.
 - `maps`: información de mapeo de memoria.
 - `root`: enlace simbólico a la raíz del sistema (`/`).
 - `stat`: estado del proceso.
 - `statm`: uso de memoria.



Control/gestión de la actividad de la CPU

- **at**: ejecutar tareas a una determinada hora.
 - Puede recibir un fichero de texto con las órdenes a ejecutar.
 - Dispone de un prompt para ir introduciendo las órdenes (Ctrl+D para finalizar).
 - **atd**: demonio que ejecuta las órdenes.
 - **atq**: consulta la lista de órdenes.
 - **atrm**: eliminar órdenes.



Control/gestión de la actividad de la CPU

- **at**: ejecutar tareas a una determinada hora.

```
1 pagutierrez@pagutierrez--TOSHIBA:~$ at 14:38
2 warning: commands will be executed using /bin/sh
3 at> echo "HOLA" > /tmp/p2
4 at> <EOT>
5 job 10 at Sat Mar  8 14:38:00 2018
6 pagutierrez@pagutierrez--TOSHIBA:~$ date
7 sáb mar  8 14:37:47 CET 2018
8 pagutierrez@pagutierrez--TOSHIBA:~$ cat /tmp/p2
9 cat: /tmp/p2: No existe el fichero o el directorio
10 pagutierrez@pagutierrez--TOSHIBA:~$ date
11 sáb mar  8 14:38:01 CET 2018
12 pagutierrez@pagutierrez--TOSHIBA:~$ cat /tmp/p2
13 HOLA
```



Control/gestión de la actividad de la CPU

- **cron**: ejecutar tareas periódicamente.
 - **crond**: demonio encargado de ejecutar las órdenes.
 - **crontab**: establecer las tareas a ejecutar (**-e**: añadir/modificar tareas, **-l**: listar tareas, **-r**: eliminar tareas).
 - **/etc/crontab**: fichero de configuración del administrador.
 - **/etc/cron.d**: directorio en el que el administrador puede copiar ficheros con formato del crontab que ejecutará cron.



Control/gestión de la actividad de la CPU

- Formato de crontab:
minuto hora día_mes mes día_semana [user] comando
- Se interpreta como una conjunción de condiciones, salvo para día_semana y día_mes (que sería disyunción).
- Los domingos son el día 0 y 7 de la semana.

```
1 # Hacer una copia de seguridad del home cada semana
2 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
3 # Otras tareas
4 5 9 * * * $HOME/tareadiaria # 9:05
5 15 14 1 * * $HOME/tareames # 14:15 el día 1 cada mes
6 0 22 * * 1-5 $HOME/tareasemanal # 22:0 de lunes a viernes
7 21 0-23/2 * * * $HOME/tareacada2horas # 0h, 2h, 4h, 6h, y 21m
8 5 4 * * sun $HOME/tareadomingos # Domingos a las 4:05
9 0 9 1 * 5 $HOME/otratarea # A las 9:00h el día 1
10 # de cada mes 0 los viernes
```



Control/gestión de la actividad de la CPU

- Si la máquina no está encendida cuando se ha requerido lanzar el proceso \Rightarrow **cron** no lo lanza.
- Se podría hacer `0 10 */3 * *` para conseguir algo parecido (lo intentaría cada tres días).
- **anacron**: no asume que la máquina está siempre encendida.
 - Combina el uso de *scripts* al inicio con el uso de **cron**.
 - Permite especificar tareas diarias, semanales o mensuales, de forma muy simple.
 - Introducir aplicaciones o enlaces a las mismas en:
 - `/etc/cron.daily/`
 - `/etc/cron.hourly/`
 - `/etc/cron.monthly/`
 - `/etc/cron.weekly/`



Rastreo de señales y llamadas al sistema

- El comando `strace` nos permite observar qué es lo que está haciendo un proceso.
- Muestra cada llamada al sistema que hace y cada señal que recibe.
 - `strace -p pid`: rastrear un proceso ya iniciado.
 - `strace comando`: iniciar un proceso y rastrearlo.
 - `strace -o salida.txt comando`: utilizar un fichero para guardar la salida.
- Procesos acaparadores:
 - Como administradores, debemos sospechar cuando un proceso acapara mucha CPU.
 - Antes de matarlos, deberíamos saber qué están haciendo.
 - Si el proceso parece legítimo, deberíamos suspenderlo con `STOP`, aplicarle `renice` y reanudarlo con `CONT` tras hablar con el dueño del proceso.



Rastreo de señales y llamadas al sistema

```
1 while 1
2     mkdir adir
3     cd adir
4     touch afile
5 end
```

- No consume mucho espacio, pero bloquea el uso del disco
¿por qué?.
- El árbol que se genera es tan grande, que ni si quiera `rm -R` es capaz de manejarlo.



Control/gestión de la actividad de la memoria

- Intercambio y paginación \Rightarrow memoria virtual para alojar procesos.
- Debemos gestionar la RAM y la zona de intercambio.
- **vmstat** (todo en KBs):
 - **swpd** \Rightarrow Cantidad de memoria virtual (intercambio) ocupada.
 - **free** \Rightarrow Cantidad de memoria virtual sin usar.
 - **buff** \Rightarrow Cantidad de memoria empleada como buffers para E/S (memoria temporal empleada por algunos dispositivos, p.ej. una tarjeta de red).
 - **cache** \Rightarrow La cantidad de memoria empleada como caché de disco.

```
1 pagutierrez@TOSHIBA:~$ vmstat 2 5
2 procs-----memory----- --swap-- --io--- -system-- ----cpu----
3 r  b    swpd    free    buff    cache    si    so    bi    bo    in    cs    us    sy    id    wa
4 0  0  465400  60300  15376  196640   11   25   140   58   213  319  10   3  83   4
5 0  0  465400  60236  15384  196628    0    0    0   10  865  658   3   2  95   1
```



Control/gestión de la actividad de la memoria

- **vmstat:**

- **si** ⇒ Cantidad de memoria traída del espacio de intercambio desde disco en KB/s.
- **so** ⇒ Cantidad de memoria intercambiada al disco en KB/s.
- **bi** ⇒ Bloques recibidos desde un dispositivo de bloques (en bloques/s).
- **bo** ⇒ Bloques enviados a un dispositivo de bloques (en bloques/s).
- **in** ⇒ N° de interrupciones por segundo (contando el reloj).
- **cs** ⇒ N° de cambios de contexto por segundo.

```
1 pagutierrez@TOSHIBA:~$ vmstat 2 2
2 procs-----memory-----swap-- ---io--- -system-- ----cpu----
3 r  b   swpd   free   buff   cache   si   so   bi   bo   in   cs  us  sy  id  wa
4 0  0  465400  60300  15376  196640   11   25  140  58  213  319  10   3  83   4
5 0  0  465400  60236  15384  196628    0    0    0  10  865  658   3   2  95   1
```



Control/gestión de la actividad de la memoria

- **Espacio para paginación:**
 - ¿Qué **tamaño** es el adecuado para la paginación?. Depende:
 - Memoria requerida por los procesos, número de procesos simultáneos, etc...
 - Demanda del sistema.
 - En portátiles, para posibilitar la hibernación, **al menos tanto espacio como memoria RAM**.
 - Se puede tener una **partición de intercambio** o un **fichero de intercambio**, ¿qué opción es la mejor?
 - Se puede controlar con números de prioridad en `/etc/fstab`.



Control/gestión de la actividad de la memoria

- **Espacio para paginación:**

- `swapon -s`: nos da un listado de particiones o ficheros activos.
- `swapon /dev/sdd1`: activar una determinada partición.
- `swapoff /dev/sdd1`: desactivar una determinada partición.
- ¿Cómo se crea un fichero de paginación?

```
1 sudo dd if=/dev/zero of=./fichero_swap bs=1048576 count=1024
2 sudo mkswap ./fichero_swap
3 sudo sync
4 sudo swapon ./fichero_swap
```

- **free**: obtener información sobre el uso de memoria (mismos campos que `top`).

```
1 pedroa@pedroa-laptop ~ $ free
2              total        used        free      shared    buffers     cached
3 Mem:           6014848      5572868      441980           0       651400      2002768
4 -/+ buffers/cache:      2918700      3096148
5 Swap:          4112636           0      4112636
```



Control/gestión de la actividad de la memoria

- **Espacio en disco:**

- **df:** muestra la capacidad, el espacio libre y el punto de montaje de cada sistema de ficheros del equipo.

```
1 pagutierrez@TOSHIBA:~$ df -h
2 S.ficheros          Tam.  Usado  Disp.  % Uso  Montado en
3 /dev/sda4           114G   28G    81G   26%    /
4 none               487M   720K   486M    1%    /dev
5 none               495M   384K   494M    1%    /dev/shm
6 none               495M   100K   495M    1%    /var/run
7 none               495M     0   495M    0%    /var/lock
8 /dev/sdd6           367G    74G   275G   22%    /media/cade...
9 /dev/sdd5           91G    12G    75G   14%    /media/4396...
```

- Si el sistema de ficheros raíz se quedase sin espacio el sistema tendría problemas. P.ej., no podría arrancar, (**¿por qué?**).
- “-i” nos permite mostrar información sobre los nodos-i.

```
1 pagutierrez@TOSHIBA:~$ df -i /dev/sda4
2 S.ficheros          Inodos   IUsado   ILibre  IUsado% Montado en
3 /dev/sda4           7569408  295620  7273788    4%    /
```



Control/gestión de la actividad de la memoria

- **Espacio en disco:**

- **du:** muestra el espacio usado por cada subdirectorio del directorio actual.

```
1 pagutierrez@TOSHIBA:~/PAS$ du -h --max-depth=1
2 4,8M ./Guiones
3 268K ./Programa1112
4 204K ./GuiaDocente
5 39M ./MaterialAdicional
6 46M .
```

- Si no ponemos `--max-depth=1` nos muestra **todas** las carpetas.
- La última línea es el acumulado.
- **¡OJO!** **du** cuenta bloques del sistema de ficheros, estén o no completamente ocupados (para un fichero de 1B cuenta 4 KB).

- **iotop.**



Control/gestión de la actividad de la memoria

- **Control de dispositivos de entrada/salida:**
 - `iostat intervalo numero`: presenta estadísticas sobre la CPU y los dispositivos y particiones de E/S.
 - `tps` \Rightarrow n° de transferencias por segundo.
 - `Blk_read/s` \Rightarrow n° de bloques leídos por segundo.
 - `Blk_wrtn/s` \Rightarrow n° de bloques escritos por segundo.
 - `Blk_read` \Rightarrow n° total de bloques leídos.
 - `Blk_wrtn` \Rightarrow n° total de bloques escritos.

```
1  pedroa@pedroa-laptop ~ $ iostat
2  Linux 3.11-2-amd64 (pedroa-laptop)  14/03/15  _x86_64_ (8 CPU)
3
4  avg-cpu:  %user   %nice %system %iowait  %steal   %idle
5             2,48    0,00    0,45    0,08    0,00   96,99
6
7  Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
8  sda                23,40        307,28        319,16    2279946    2368084
9  sdb                 0,71         4,01         0,62     29747     4588
```



Referencias



Evi Nemeth, Garth Snyder, Trent R. Hein y Ben Whaley
Unix and Linux system administration handbook.
Capítulo 5. *Controlling processes*, Capítulo 9. *Periodic processes*.
Prentice Hall. Cuarta edición. 2010.



Aeleen Frisch.
Essential system administration.
Capítulo 15. *Managing system resources*.
O'Reilly and Associates. Tercera edición. 2002.



Programación y Administración de Sistemas

5. Gestión de los recursos del sistema

Pedro Antonio Gutiérrez

Asignatura “Programación y Administración de Sistemas”
2º Curso Grado en Ingeniería Informática
Escuela Politécnica Superior
(Universidad de Córdoba)
pagutierrez@uco.es

1 de abril de 2018

