

## Laboratorio 6



**Apellidos: Moreno Vera**

**Nombres: Felipe Adrian**

**Código: 20120354I**

**Asignatura: Administración de Redes (CC481)**

**2016 - I**

## **Indice**

<b>Actividad 1 .....</b>	<b>(3)</b>
<b>Actividad 2 .....</b>	<b>(8)</b>
<b>Actividad 3 .....</b>	<b>(15)</b>
<b>Actividad 4 .....</b>	<b>(17)</b>

## Preámbulo . . .

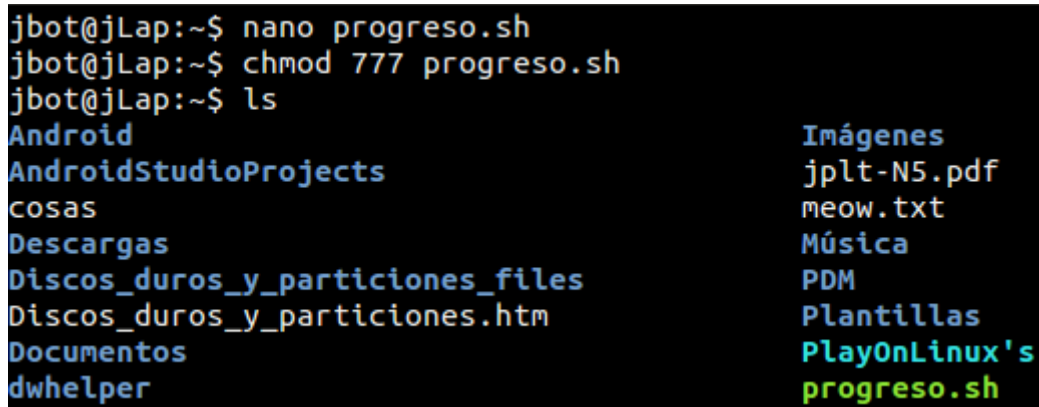
Este laboratorio lo haré desde mi Sistema Operativo Host Ubuntu 14.04.3 LTS Trusty.

## Actividad 1

**1. Para probar diferentes acciones sobre los procesos vamos a usar un comando gráfico que permita comprobar fácilmente el estado del proceso asociado.**

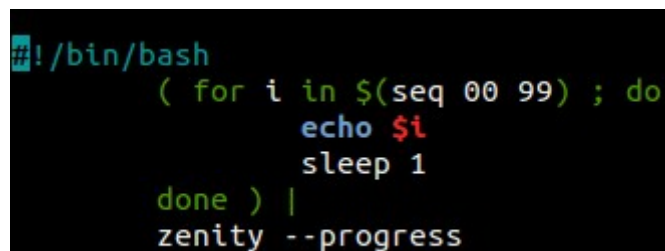
**Escribir el siguiente contenido en un archivo (progreso.sh) y darle permisos de ejecución.**

```
#!/bin/bash
( for i in $(seq 00 99) ; do
    echo $i
    sleep 1
done ) |
zenity --progress
```



A terminal window showing the execution of commands to create and list files. The commands are: `nano progreso.sh`, `chmod 777 progreso.sh`, and `ls`. The output of `ls` shows a list of files and directories, including `progreso.sh` which is highlighted in green.

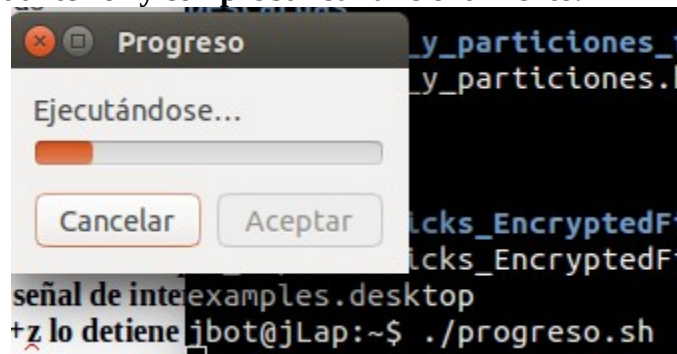
```
jbot@jLap:~$ nano progreso.sh
jbot@jLap:~$ chmod 777 progreso.sh
jbot@jLap:~$ ls
Android                               Imágenes
AndroidStudioProjects                jplt-N5.pdf
cosas                                meow.txt
Descargas                            Música
Discos_duros_y_particiones_files     PDM
Discos_duros_y_particiones.htm       Plantillas
Documentos                           PlayOnLinux's
dwhelper                             progreso.sh
```



A terminal window showing the content of the `progreso.sh` script. The script is a bash script that uses a loop to print numbers from 00 to 99, with a 1-second delay between each print, and then pipes the output to `zenity --progress`.

```
#!/bin/bash
( for i in $(seq 00 99) ; do
    echo $i
    sleep 1
done ) |
zenity --progress
```

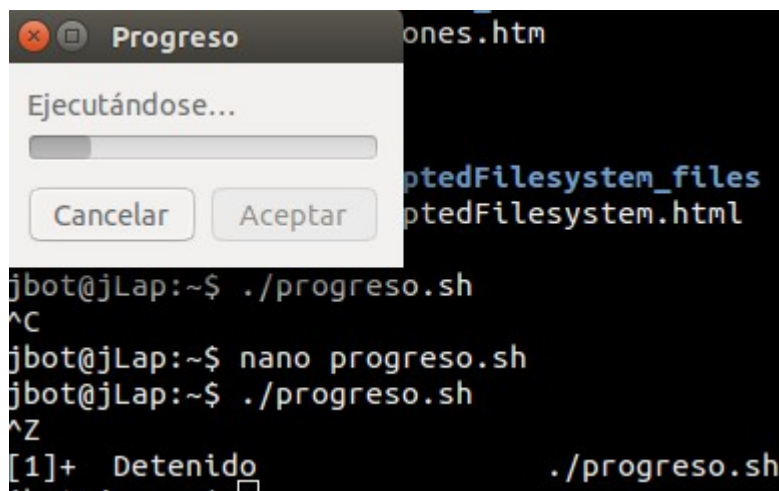
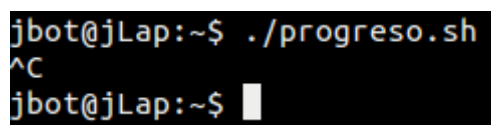
2. Ejecutar el script anterior y comprobar su funcionamiento.



3. Para detener la ejecución terminar un proceso se envía una señal. La combinación Ctrl+c envía la señal de interrupción (SIGINT) que interrumpe el proceso y Ctrl+z lo detiene (SIGSTOP). Comprobar el efecto de ambos sobre el comando.

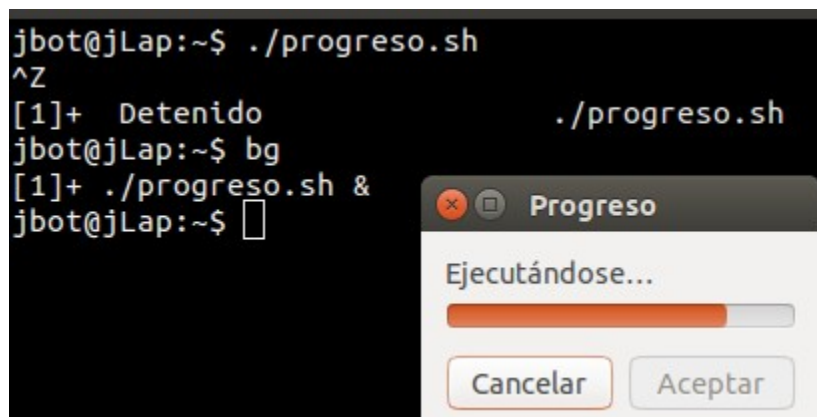
Ctrl+c : esta combinación de teclas hace que el proceso ya no se ejecute o es interrumpido.

Ctrl+z : esta combinación de teclas hace que el proceso se detenga o entre en estado de pausa.



4. Ejecuciones en primer y segundo plano. Cuando ejecutamos un comando decimos que está en primer plano (foreground) y puede recibir las señales generadas por teclado (ej. Ctrl+c). Equivalentemente, un proceso en segundo plano mantiene su ejecución pero desvincula su ID de grupo de procesos del de la shell. Los comandos fg y bg permiten poner en primer (foreground) y segundo plano (background) un proceso, respectivamente:

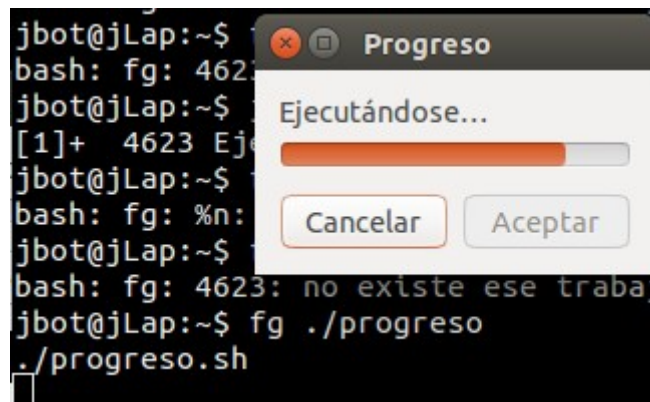
1. Enviar un proceso a segundo plano: (1) arrancar progreso.sh; (2) Ctrl+z; (3) bg.



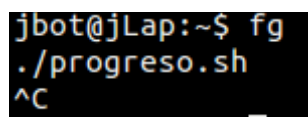
```
jbot@jLap:~$ ./progreso.sh
^Z
[1]+  Detenido                  ./progreso.sh
jbot@jLap:~$ bg
[1]+  ./progreso.sh &
jbot@jLap:~$
```

2. Recuperar la ejecución en primer plano y terminar su ejecución Ctrl+c

Con bg, volvimos a ejecutar el proceso, y al llegar al final, damos fg para mandarlo a primer plano y después Ctrl + c

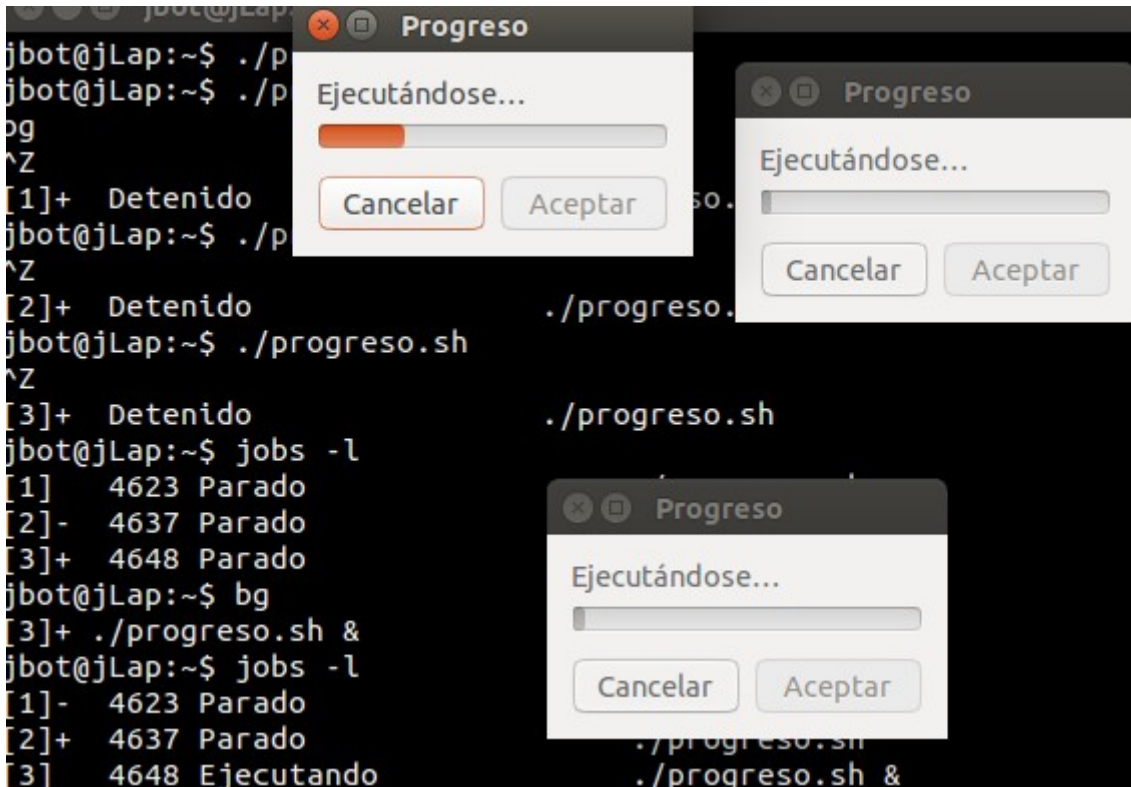


```
jbot@jLap:~$ fg
bash: fg: 4623: no existe ese trabajo
jbot@jLap:~$ fg
[1]+  4623 Ejecutando ./progreso.sh
jbot@jLap:~$ fg
bash: fg: %n: no existe ese trabajo
jbot@jLap:~$ fg ./progreso
./progreso.sh
```



```
jbot@jLap:~$ fg
./progreso.sh
^C
```

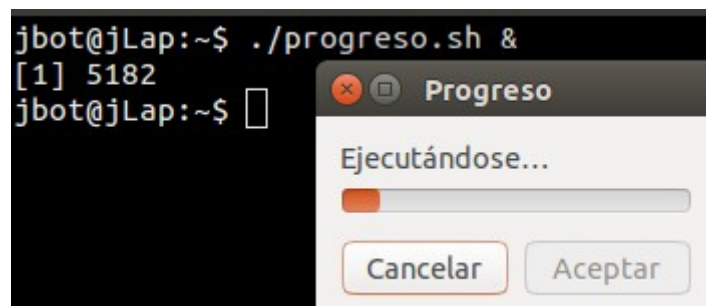
3. **bg** y **fg** admiten especificar el trabajo que modifican de varias formas (%n). Usar esta funcionalidad cuando hay varios procesos en segundo plano o suspendidos. Ver la lista con **jobs** (opción -l).



4. Consultar la página de manual **bash** en relación a **jobs**, **fg** y **bg**.

Estos comandos son propios del **bash**, es decir, no son binarios sino que están dentro de la shell. El número que usan **fg** y **bg** para elegir el proceso es el de la primera columna que sale del comando `'jobs -l'` o por el nombre de proceso.

5. Es posible arrancar un proceso en segundo plano directamente usando **&** al final. Arrancar el comando de prueba en segundo plano y comprobarlo con **jobs**.



**6. Terminal de control. Los procesos tienen asociados un terminal de control al que se envían las salidas estándar y de error y que recoge la entrada estándar si es necesaria. Ejecutar el siguiente comando (sleep 1; date ; cat - > /tmp/entrada; sleep 1; date) en segundo plano y analizar dónde se muestra la salida estándar y qué sucede cuando lee de la entrada estándar.**

La salida estándar se observa en el archivo /tmp/entrada, y la entrada estándar es en la línea de comandos, cuando se lee de la entrada estándar la shell lo lee y si no es un comando válido se detiene el proceso.

Mostramos el proceso ...

```
jbot@jLap:~$ (sleep 1; date ; cat - > /tmp/entrada; sleep 1; date)
mar feb  9 15:42:29 PET 2016

^Z[1]  Hecho                ./progreso.sh

[2]+  Detenido              ( sleep 1; date; cat - > /tmp/entrada; sleep 1; da
te )
jbot@jLap:~$ bg
[2]+ ( sleep 1; date; cat - > /tmp/entrada; sleep 1; date ) &
jbot@jLap:~$
```

Ahora en esta imagen, se observan la entrada, al estar en primer plano, en entrada no había nada, al mandarlo a segundo plano, hizo un [ENTER] en la entrada.

```
jbot@jLap:/tmp$ cat entrada
jbot@jLap:/tmp$ cat entrada

jbot@jLap:/tmp$
```

**7. Arrancar el comando de prueba en segundo plano, cerrar la ventana del terminal donde se arranco y comprobar qué sucede. Ejecutar el mismo comando con la orden nohup (man nohup) también en segundo plano y repetir el ejercicio.**

```
jbot@jLap:~$ (sleep 1; date ; cat - > /tmp/entrada; sleep 1; date) &
[3] 5512

[2]+  Detenido              ( sleep 1; date; cat - > /tmp/entrada; sleep 1; da
te )
jbot@jLap:~$ mar feb  9 15:48:47 PET 2016

[3]+  Detenido              ( sleep 1; date; cat - > /tmp/entrada; sleep 1; da
te )
jbot@jLap:~$
```

El proceso se elimina al cerrar la shell. y con nohup se crea un archivo de nombre nohup.out en el directorio /home del usuario. lo que hace el comando nohup es redirigir la entrada estándar si es el terminal a /dev/null y la salida estándar si es el terminal a nohup.out.

```
jbot@jLap:~$ nohup sleep 1; date ; cat - > /tmp/entrada; sleep 1; date &
nohup: se descarta la entrada y se añade la salida a «nohup.out»
mar feb  9 16:29:36 PET 2016

bg
^Z
[4]+  Detenido                  cat - > /tmp/entrada
[5] 6075
jbot@jLap:~$ mar feb  9 16:30:26 PET 2016

[5]  Hecho                      date
jbot@jLap:~$
```

## Actividad 2

1. La herramienta principal para ver los procesos en ejecución del sistema es ps. Ejecutar la orden sin argumentos y comprobar su salida.

```
jbot@jLap:~$ ps
  PID TTY          TIME CMD
 6053 pts/3        00:00:00 bash
 6167 pts/3        00:00:00 ps
```

2. Estudiar la página de manual de ps, especialmente las opciones más comunes (ej. aux) y el significado de los datos mostrados por cada proceso.

PID : número identificador del proceso.

% CPU : porcentaje de memoria usado.

% MEM: tamaño de la memoria virtual del proceso en KiB.

VSZ: la memoria física que no ha sido enviada al swap que ha sido usado en kb.

TTY: el terminal que lo controla

STAT: el estado del proceso.

START: la hora de inicio del proceso.

TIME: tiempo acumulado de uso del cpu.

COMMAND: el nombre del proceso en listado y el comando mediante el cual ha sido lanzado.



```
jbot@jLap:~$ ps aux
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.0	33908	4444	?	Ss	14:40	0:02	/sbin/init
root	2	0.0	0.0	0	0	?	S	14:40	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	S	14:40	0:00	[ksoftirqd/0]
root	4	0.0	0.0	0	0	?	S	14:40	0:00	[kworker/0:0]
root	5	0.0	0.0	0	0	?	S<	14:40	0:00	[kworker/0:0H]
root	7	0.0	0.0	0	0	?	R	14:40	0:02	[rcu_sched]
root	8	0.0	0.0	0	0	?	S	14:40	0:00	[rcuos/0]
root	9	0.0	0.0	0	0	?	S	14:40	0:00	[rcuos/1]

3. El consumo de la memoria virtual se puede obtener con free(1). Consultar el consumo de memoria del sistema.

```
jbot@jLap:~$ free
```

	total	usado	libre	compart.	búffers	almac.
Mem:	12223344	3601144	8622200	410848	147088	1477196
-/+ buffers/cache:	1976860	10246484				
Intercambio:	12500988	0	12500988			

4. La herramienta top es muy útil porque muestra un resumen del sistema que incluye: carga (ver comando uptime), estado de la memoria (free) y procesos (ps). Esta herramienta permite filtrar por usuario, enviar señales a procesos, ordenar la lista de procesos según diferentes criterios y configurar la información mostrada. Ejecutar top y estudiar su uso (man top; pulsar h una vez arrancada).

```
top - 16:45:23 up 2:04, 2 users, load average: 0,40, 0,44, 0,35
Tareas: 218 total, 2 ejecutar, 216 hibernar, 0 detener, 0 zombie
%Cpu(s): 4,1 usuario, 1,9 sist, 0,0 adecuado, 93,4 inact, 0,6 en espera, 0,0 ha
KiB Mem: 12223344 total, 3634364 used, 8588980 free, 150568 buffers
KiB Swap: 12500988 total, 0 used, 12500988 free. 1444384 cached Mem
```

PID	USUARIO	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	HORA+	ORDEN
3580	jbot	20	0	97,959g	1,000g	101008	S	15,0	8,6	26:06.80	firefox
1231	root	20	0	446588	100392	81332	S	4,7	0,8	2:21.41	Xorg
1966	jbot	20	0	1628548	214152	53736	S	4,3	1,8	2:26.99	compiz
6440	jbot	20	0	632888	26524	21548	S	1,7	0,2	0:00.19	gnome-screensho
1941	jbot	9	-11	514060	12604	9732	S	1,3	0,1	0:44.41	pulseaudio
3699	jbot	20	0	488604	62612	45576	S	0,7	0,5	0:45.40	plugin-containe
4593	jbot	20	0	656852	32876	24880	S	0,3	0,3	0:05.05	gnome-terminal
1	root	20	0	33908	4444	2684	S	0,0	0,0	0:02.90	init
2	root	20	0	0	0	0	S	0,0	0,0	0:00.01	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:00.03	ksoftirqd/0
4	root	20	0	0	0	0	S	0,0	0,0	0:00.34	kworker/0:0
5	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/0:0H
7	root	20	0	0	0	0	R	0,0	0,0	0:03.14	rcu_sched
8	root	20	0	0	0	0	S	0,0	0,0	0:00.66	rcuos/0
9	root	20	0	0	0	0	S	0,0	0,0	0:00.58	rcuos/1
10	root	20	0	0	0	0	S	0,0	0,0	0:02.07	rcuos/2
11	root	20	0	0	0	0	S	0,0	0,0	0:00.55	rcuos/3

```

Help for Interactive Commands - procps-ng version 3.3.9
Window 1:Def: Cumulative mode Apagado. System: Delay 3,0 secs; Secure mode Apagado.

Z,B,E,e Global: 'Z' colors; 'B' bold; 'E'/'e' summary/task memory scale
l,t,m Toggle Summary: 'l' load avg; 't' task/cpu stats; 'm' memory info
0,1,2,3,I Toggle: '0' zeros; '1/2/3' cpus or numa node views; 'I' Irix mode
f,F,X Fields: 'f'/'F' add/remove/order/sort; 'X' increase fixed-width

L,&,<,> . Locate: 'L'/'&' find/again; Move sort column: '<'/'>' left/right
R,H,V,J . Toggle: 'R' Sort; 'H' Threads; 'V' Forest view; 'J' Num justify
c,i,S,j . Toggle: 'c' Cmd name/line; 'i' Idle; 'S' Time; 'j' Str justify
x,y . Toggle highlights: 'x' sort field; 'y' running tasks
z,b . Toggle: 'z' color/mono; 'b' bold/reverse (only if 'x' or 'y')
u,U,o,O . Filter by: 'u'/'U' effective/any user; 'o'/'O' other criteria
n,#,^0 . Set: 'n'/'#' max tasks displayed; Show: Ctrl+'0' other filter(s)
C,... . Toggle scroll coordinates msg for: up,down,left,right,home,end

k,r Gestiona tareas: «k» detener; «r» reiniciar
d o s Establece intervalo de actualización
W,Y Write configuration file 'W'; Inspect other output 'Y'
q Quit
( commands shown with '.' require a visible task display window )
Press 'h' or '?' for help with Windows,
Type 'q' or <Esc> to continue

```

Con el comando h, nos muestra la opción de Help.

Con las diversos comandos que ofrece se puede, por ejemplo, filtrar los procesos por usuario(u), matar los procesos(k), cambiar el tiempo de actualización(d). También se puede modificar el modo de mostrar la información, como ordenar por campos mostrados(F).

**5. De la misma forma el comando vmstat permite recoger información sobre el rendimiento dinámico del sistema. Estudiar su uso (man vmstat) y el significado de la información que muestra.**

```

jbot@jLap:~$ vmstat
procs -----memory----- ---swap-- -----io----- -system-- -----cpu-----
 r  b   swpd  libre búfer caché   si   so    bi    bo   in   cs us sy id wa st
 2   0     0 8568424 157208 1447672    0    0   31   44 158 1179 6 1 92 1 0

```

Segun el manual. vmstat muestra informacion sobre los procesos, memoria, paginacion, bloques recibidos de IO y actividad del CPU.

Procs:

r: numero de procesos esperando para correr.

b: numero de procesos en bloqueo ininterrumpible

Memory:

swpd: cantidad de memoria virtual usada

free: cantidad de memoria virtual libre

buff: cantidad de memoria usada como buffer

cache: cantidad de memoria usada como cache

inact: cantidad de memoria inactiva

active: cantidad de memoria activa

Swap

si: cantidad de memoria intercambiada desde el disco  
so: cantidad de memoria intercambiada hacia el disco

IO

bi: bloques recibidos desde un dispositivo  
bo: bloques enviados a un dispositivo

System

in: el numero de interrupciones por segundo  
cs: numero de cambios de contexto por segundo

CPU

us: tiempo(%) del cpu usado en modo non-kernel  
sy: tiempo(%) del cpu usado en modo kernel  
id: tiempo(%) del cpu en modo ocioso  
wa: tiempo(%) del cpu esperando por IO  
st: tiempo(%) del cpu robado por una maquina virtual

**6. Otra forma sencilla de localizar procesos específicos (por ejemplo para enviar una señal) es mediante los comandos pgrep y pidof. Estudiar el uso de pgrep y pidof, buscar por ejemplo los procesos que encajen con gnome.**

Con pidof, encuentras el PID de un proceso específico.

```
jbot@jLap:~$ pidof gnome-keyring-daemon
1706
```

Con pgrep, tal cual su nombre indica, devuelve el PID de los procesos que tengan en su nombre el argumento escrito.

```
jbot@jLap:~$ pgrep -l gnome
1706 gnome-keyring-d
1870 gnome-session
1989 polkit-gnome-au
4593 gnome-terminal
4600 gnome-pty-helpe
```

## 7. El comando kill sirve para enviar señales a un proceso:

### 1. Consultar kill -l para ver las señales disponibles.

Nos devuelve toda la lista compatible al comando kill.

```
jbot@jLap:~$ kill -l
1) SIGHUP      2) SIGINT      3) SIGQUIT     4) SIGILL      5) SIGTRAP
6) SIGABRT     7) SIGBUS     8) SIGFPE     9) SIGKILL    10) SIGUSR1
11) SIGSEGV    12) SIGUSR2    13) SIGPIPE    14) SIGALRM    15) SIGTERM
16) SIGSTKFLT  17) SIGCHLD   18) SIGCONT    19) SIGSTOP    20) SIGTSTP
21) SIGTTIN    22) SIGTTOU   23) SIGURG     24) SIGXCPU    25) SIGXFSZ
26) SIGVTALRM  27) SIGPROF   28) SIGWINCH   29) SIGIO      30) SIGPWR
31) SIGSYS     34) SIGRTMIN   35) SIGRTMIN+1 36) SIGRTMIN+2 37) SIGRTMIN+3
38) SIGRTMIN+4 39) SIGRTMIN+5 40) SIGRTMIN+6 41) SIGRTMIN+7 42) SIGRTMIN+8
43) SIGRTMIN+9 44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5 60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
```

### 2. kill puede usar un PID o un job (%n). Repetir los ejercicios fg/bg usando kill en lugar de Ctrl+c y Ctrl+z.

Comando para detener (Ctrl + z).

```
jbot@jLap:~$ kill -s 19 6969
```

```
jbot@jLap:~$ ./progreso.sh
```

```
[1]+  Detenido          ./progreso.sh
```

el Ctrl + C es simplemente killall [nombre process]

```
jbot@jLap:~$ ./progreso.sh
```

```
Terminado
```

```
jbot@jLap:~$
```

```
jbot@jLap: ~
```

```
jbot@jLap:~$ killall progreso.sh
```

```
jbot@jLap:~$
```

3. Se puede enviar un señal a un proceso por nombre (a todos lo procesos que encajen) con pkill.

```
jbot@jLap:~$ ./progreso.sh
./progreso.sh: línea 6: 4599 Tubería rota          ( for i in $(seq 00 99);
do
    echo $i; sleep 1;
done )
4600 Terminado | zenity --progress
jbot@jLap:~$ █

jbot@jLap:~$ pkill -f zenity
jbot@jLap:~$ █
```

4. Probar el envío de señales en top con el comando k (kill).

```
KiB Swap: 12500988 total, 0 used, 12500988 free. 1632036 cached Mem
Enviar la señal pid 4719 [15/sigterm] █
```

PID	USUARIO	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	HORA+	ORDEN
4719	jbot	20	0	16624	2708	2548	S	0,0	0,0	0:00.00	progreso.sh
4720	jbot	20	0	16632	2332	2092	S	0,0	0,0	0:00.00	progreso.sh
4721	jbot	20	0	499880	33700	27748	S	0,0	0,3	0:00.08	zenity
4735	jbot	20	0	11396	812	732	S	0,0	0,0	0:00.00	sleep

```
jbot@jLap:~$ ./progreso.sh
Terminado
jbot@jLap:~$ █
```

8. Finalmente el comando lsof permite ver los descriptores que tiene abierto un proceso, consultar la página de manual y probar las siguientes opciones:

1. Procesos tienen abierto para escritura /var/log/messages.

```
[root@localhost fapCentOSuser1]# lsof /var/log/messages
COMMAND  PID USER  FD  TYPE DEVICE SIZE/OFF  NODE NAME
rsyslogd 1255 root   1w   REG  8,3  176847 259943 /var/log/messages
abrt-dump 1731 root   4r   REG  8,3  176847 259943 /var/log/messages
[root@localhost fapCentOSuser1]#
```

Este comando lo hice en mi máquina virtual, pues en ubuntu no existe el fichero messages.

## 2. Procesos que tienen abierto algún fichero o directorio de /etc (+D).

```
jbot@jLap:~$ sudo lsof +D /etc/
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1000/gvfs
Output information may be incomplete.
COMMAND      PID  USER  FD   TYPE DEVICE SIZE/OFF      NODE NAME
avahi-dae    805  avahi  cwd   DIR   8,6    4096  5505039 /etc/avahi
avahi-dae    805  avahi  rtd   DIR   8,6    4096  5505039 /etc/avahi
cups-brow   1068   root   3r    REG   8,6    2038  5511919 /etc/passwd
Xorg        1247   root   cwd   DIR   8,6    4096  5505029 /etc/X11
ibus-dcon   1961  jbot   mem   REG   8,6    3931  5506123 /etc/dconf/db/ibus
cpsd        3765   root   7r    REG   8,6    2038  5511919 /etc/passwd
dbus        3768    lp     4r    REG   8,6    2038  5511919 /etc/passwd
```

## 3. Ficheros que tiene abierta la shell que está usando (-p).

```
jbot@jLap:~$ ps
  PID TTY          TIME CMD
 4330 pts/3        00:00:00 bash
 5282 pts/3        00:00:00 ps
jbot@jLap:~$ lsof -p 4330
COMMAND      PID  USER  FD   TYPE DEVICE SIZE/OFF      NODE NAME
bash         4330  jbot   cwd   DIR   8,6    4096  7733250 /home/jbot
bash         4330  jbot   rtd   DIR   8,6    4096           /
bash         4330  jbot   txt   REG   8,6  1021112  7602182 /bin/bash
bash         4330  jbot   mem   REG   8,6   47712  6291475 /lib/x86_64-linux-gnu/libn
ss_files-2.19.so
bash         4330  jbot   mem   REG   8,6   47760  6295432 /lib/x86_64-linux-gnu/libn
ss_nis-2.19.so
bash         4330  jbot   mem   REG   8,6   97296  6291550 /lib/x86_64-linux-gnu/libn
```

Muestra información de un proceso.

## 4. NOTA: Las conexiones de red (sockets) se pueden ver con -i.

```
jbot@jLap:~$ lsof -i | head
COMMAND      PID  USER  FD   TYPE DEVICE SIZE/OFF      NODE NAME
firefox      2363  jbot   40u  IPv4  63870      0t0  TCP 172.16.9.10:34919->d3-11-0-0-19-0.a03.nycmny01.u
s.ra.verio.net:https (ESTABLISHED)
firefox      2363  jbot   43u  IPv4  61476      0t0  TCP 172.16.9.10:60322->xx-fbcdn-shv-01-gru2.fbcdn.ne
t:https (ESTABLISHED)
firefox      2363  jbot   60u  IPv4  61152      0t0  TCP 172.16.9.10:41124->a184-29-227-206.deploy.static
.akamai technologies.com:https (ESTABLISHED)
firefox      2363  jbot   61u  IPv4  61156      0t0  TCP 172.16.9.10:42706->client-200.60.136.49.speedy.n
et.pe:http (ESTABLISHED)
firefox      2363  jbot   65u  IPv4  27904      0t0  TCP 172.16.9.10:43651->edge-star-mini-shv-01-gru2.fa
cebook.com:https (ESTABLISHED)
firefox      2363  jbot   90u  IPv4  21346      0t0  TCP 172.16.9.10:57944->xx-fbcdn-shv-01-mia1.fbcdn.ne
t:https (ESTABLISHED)
firefox      2363  jbot   95u  IPv4  27114      0t0  TCP 172.16.9.10:58282->edge-star-shv-01-gru2.facebo
ok.com:https (ESTABLISHED)
unity-sco    3128  jbot    4u  IPv4  63403      0t0  TCP 172.16.9.10:41405->productsearch.ubuntu.com:http
s (ESTABLISHED)
gvfsd-htt    3239  jbot    9u  IPv4  29322      0t0  TCP 172.16.9.10:40514->productsearch.ubuntu.com:http
s (CLOSE_WAIT)
```



## Actividad 3

1. La programación de trabajos generales se especifica en `/etc/crontab`.

Cada línea especifica una variable o un trabajo, mediante 5 parámetros temporales.

Un `*` en uno de los parámetros significa para todos los posibles valores.

Determinar la configuración para ejecutar un comando:

1. 15 minutos después de la medianoche todos los sábados.

2. El primer día de cada mes a las 3:30 AM.

```
# m = minute(0-59)
# h = hours(0-23)
# dom = day of month(1-31)
# mon = month(1-12) OR jan,feb,mar,apr...
# dow = day of week(0-6) (Sunday=0 or 7)
# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
# Anadimos los comandos cat meow.txt
15 0 * * 6 cat meow.txt
30 3 1 * * ls
```

2. Consultar los archivos del directorio `/etc/cron.d` e interpretar su contenido.

```
jbot@jLap:/etc/cron.d$ ls
anacron
jbot@jLap:/etc/cron.d$ cat anacron
# /etc/cron.d/anacron: crontab entries for the anacron package

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

30 7 * * * root    start -q anacron || :
jbot@jLap:/etc/cron.d$
```

Anacron es un programa libre que ejecuta asíncronamente tareas programadas en sistemas UNIX de manera periódica. 1 Se lanza como daemon (demonio ) durante el inicio del sistema y permanece en segundo plano.

A diferencia de cron, anacron no es un demonio, es decir, no está corriendo todo el tiempo. De hecho solo corre a través de scripts de inicio del sistema o a través de tareas programadas de cron. Con anacron no se pueden programar tareas en intervalos menores a días, mientras que con cron se pueden planificar tareas a ser ejecutadas en horas o minutos. Por otro lado, anacron no ejecuta tareas en tiempo específicos como cron hace.

Es una herramienta complementaria, no sustituye al cron.

### 3. Cron está a disposición de los usuarios:

1. `crontab -e`, permite editar una nueva entrada. Planificar la ejecución de un comando (`date >> /tmp/salida`) cada 5 minutos (`*/5`).

```
GNU nano 2.2.6 Archivo: /tmp/crontab.rqG1Pt/crontab
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
*/5 * * * * date >> /tmp/salida
```

2. comprobar la planificación de trabajos con `crontab -l`.

```
#
# m h dom mon dow   command
*/5 * * * * date >> /tmp/salida
jbot@jLap:/etc/cron.d$ crontab -l
```

### 4. El comando `at` sirve para planificar trabajos específicos que se ejecutan una sola vez, planificar la ejecución de un comando (ej. `Date > /tmp/at.ejemplo`):

1. `at <hora>`, la hora se puede especificar de muchas formas ver (`man at`), por ejemplo `at now + 3 minutes`
2. Escribir los comandos que se ejecutarán, terminar con `Ctrl+d`

```
jbot@jLap:~$ at now +3 minutes
El programa «at» no está instalado. Puede instalarlo escribiendo:
sudo apt-get install at
```

No teníamos el paquete, así que instalamos ...

```
jbot@jLap:~$ at now +3 minutes
warning: commands will be executed using /bin/sh
at> date > /tmp/at.ejemplo
at> <EOT>
at> <EOT>
job 1 at Wed Feb 10 18:02:00 2016
```



3. Consultar los trabajos planificados con atq (atrm elimina comandos)

```
jbot@jLap:~$ atq
1          Wed Feb 10 18:02:00 2016 a jbot
jbot@jLap:~$ atrm 1
jbot@jLap:~$ atq
```

## Actividad 4

1. Comprobar que el paquete rsyslog está instalado y que está en ejecución.

```
jbot@jLap:~$ ps aux | grep rsyslog
syslog      821  0.0  0.0 255848  9164 ?        Ssl  09:36   0:00 rsyslogd
jbot        5071  0.0  0.0  15968  2236 pts/0    S+   12:05   0:00 grep --color=au
to rsyslog
```

2. El archivo de configuración es /etc/rsyslog.conf. Estudiar la sección rules, cada regla hace referencia <servicio>.<severidad>. Los servicios son authpriv, cron, kern, mail, news, user, y uucp; y los niveles de severidad, de menor a mayor son: debug, info, notice, warn, err, crit, alert, emerg.

```
#### RULES ####

# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                  /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none    /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                  /var/log/secure

# Log all the mail messages in one place.
mail.*                                       -/var/log/maillog

# Log cron stuff
cron.*                                       /var/log/cron

# Everybody gets emergency messages
*.emerg                                     *

# Save news errors of level crit and higher in a special file.
uucp,news.crit                             /var/log/spooler

# Save boot messages also to boot.log
local7.*                                    /var/log/boot.log
```

Explican sobre el servicio, y el lugar en donde se guardan todos los datos que retorna dicho servicio, por ejemplo, local7 se guarda en boot.log

### 3. Estudiar los archivos de log en /var/log, ejemplo boot.log, syslog. Y auth.log.

Boot.log: contiene el registro de todos los procesos que se arrancan al bootear el SO.

```
jbot@jLap:~$ head /var/log/boot.log
* Stopping Read required files in advance [
OK ]
* Starting Mount filesystems on boot [
OK ]
* Starting Populate and link to /run filesystem [
OK ]
* Stopping Populate and link to /run filesystem [
OK ]
* Stopping Track if upstart is running in a container [
OK ]
* Starting Initialize or finalize resolvconf [
OK ]
* Starting set console keymap [
OK ]
* Starting Signal sysvinit that virtual filesystems are mounted [
OK ]
* Starting Signal sysvinit that virtual filesystems are mounted [
OK ]
* Starting Bridge udev events into upstart [
OK ]
jbot@jLap:~$
```

Syslog Contiene registro con fecha hora minuto y segundo de todos los procesos relativos a la red hechos por tu máquina desde que prendió hasta apagar.

```
jbot@jLap:~$ head /var/log/syslog
Feb 13 10:45:46 jLap rsyslogd: [origin software="rsyslogd" swVersion="7.4.4" x-pid="821" x-info="http://www.rsyslog.com"] rsyslogd was HUPed
Feb 13 10:46:25 jLap anacron[3728]: Job `cron.daily' terminated
Feb 13 10:46:25 jLap anacron[3728]: Normal exit (1 job run)
Feb 13 10:50:01 jLap CRON[4337]: (jbot) CMD (date >> /tmp/salida)
Feb 13 10:55:01 jLap CRON[4416]: (jbot) CMD (date >> /tmp/salida)
Feb 13 10:56:55 jLap dhclient: DHCPREQUEST of 192.168.0.17 on wlan0 to 192.168.0.1 port 67 (xid=0x36bc103)
Feb 13 10:56:55 jLap dhclient: DHCPACK of 192.168.0.17 from 192.168.0.1
Feb 13 10:56:55 jLap dhclient: bound to 192.168.0.17 -- renewal in 1658 seconds.
Feb 13 10:56:55 jLap NetworkManager[1139]: <info> (wlan0): DHCPv4 state changed renew -> renew
Feb 13 10:56:55 jLap NetworkManager[1139]: <info> address 192.168.0.17
jbot@jLap:~$
```

auth.log: Es el log de autorizaciones (programas como su, passwd, login, shutdown, sshd) para Debian y Ubuntu (Para entornos Red Hat y Fedora el archivo es /var/log/secure). Este log se debería resguardar bien, por más que ejecutando sudo su o sudo sh, en este log no siga dejando huellas

```
jbot@jLap:~$ head /var/log/auth.log
Feb  8 17:46:48 jLap pkexec: pam_unix(polkit-1:session): session opened for user root by (uid=1000)
Feb  8 17:46:48 jLap pkexec[4269]: jbot: Executing command [USER=root] [TTY=unknown] [CWD=/home/jbot] [COMMAND=/usr/lib/update-notifier/package-system-locked]
Feb  8 17:47:10 jLap gnome-keyring-daemon[1798]: keyring alias directory: /home/jbot/.local/share/keyrings
Feb  8 17:54:57 jLap dbus[747]: [system] Rejected send message, 7 matched rules; type="method_return", sender=":1.11" (uid=0 pid=1207 comm="/usr/sbin/dnsmasq --no-resolv --keep-in-foreground") interface="(unset)" member="(unset)" error name="(unset)" requested_reply="0" destination=":1.5" (uid=0 pid=869 comm="NetworkManager ")
Feb  8 18:05:06 jLap sudo:      jbot : TTY=pts/0 ; PWD=/home/jbot ; USER=root ; COMMAND=/sbin/poweroff
Feb  8 18:05:06 jLap sudo: pam_unix(sudo:session): session opened for user root by jbot(uid=0)
Feb  8 18:05:06 jLap sudo: pam_unix(sudo:session): session closed for user root
Feb  9 11:44:51 jLap systemd-logind[821]: New seat seat0.
Feb  9 11:44:51 jLap systemd-logind[821]: Watching system buttons on /dev/input/event2 (Power Button)
Feb  9 11:44:51 jLap systemd-logind[821]: Watching system buttons on /dev/input/event6 (Video Bus)
jbot@jLap:~$
```

4. Los archivos de log pueden crecer demasiado lo que dificulta su manejo. Por defecto la utilidad logrotate rota los logs cada semana. Estudiar los contenidos de /etc/logrotate.conf y los archivos de configuración específicos de cada servicio en /etc/logrotate.d.

```
# see "man logrotate" for details
# rotate log files weekly
weekly

# use the syslog group by default, since this is the owning group
# of /var/log/syslog.
su root syslog

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own wtmp, or btmp -- we'll rotate them here
/var/log/wtmp {
    missingok
    monthly
    create 0664 root utmp
    rotate 1
}

/var/log/btmp {
    missingok
    monthly
    create 0660 root utmp
    rotate 1
}

# system-specific logs may be configured here
jbot@jLap:~$
```

/etc/logrotate.conf : los archivos log en /var/log los monitorea y les realiza acciones dependiendo de la configuración escrita en el archivo logrotate.conf, por ejemplo la configuración por defecto dice que los archivos log se rotan(guardan y cambian) cada semana, se pueden rotar hasta 4 veces antes de ser eliminados, cuando se rota un log se crea uno vacío, cada log rotado tiene en su nombre agregado la fecha que fue rotado. Además se pueden agregar reglas para logs específicos como se aprecia en las dos últimas secciones que parecen funciones, empiezan con la dirección física del log y entre llaves se agrega las reglas para ese log.

/etc/logrotate.d : como se mencionó antes, se puede agregar para cada archivo log un conjunto de reglas específicas para este log, esta carpeta tiene las reglas para los logs de:

```
jbot@jLap:/etc/logrotate.d$ ls
apport      dpkg        ppp         speech-dispatcher  upstart
apt         iptraf      rsyslog     ufw               winbind
cups-daemon pm-utils    samba       unattended-upgrades
```

## Referencias:

<http://rm-rf.es/nohup-mantiene-ejecucion-comando-pese-salir-terminal/>  
<https://en.wikipedia.org/wiki/Nohup>  
[http://linuxcommand.org/man\\_pages/vmstat8.html](http://linuxcommand.org/man_pages/vmstat8.html)  
<http://storm.malditainternet.com/wp/2011/05/usando-y-entendiendo-vmstat/>  
<http://ss64.com/bash/fg.html>  
<http://www.tldp.org/LDP/abs/html/x9644.html>  
<http://www.thegeekstuff.com/2010/05/unix-background-job/>  
<http://francisconi.org/linux/archivos-directorios-importantes/var-log-authlog>  
<http://www.computerhope.com/unix/uat.htm>  
<https://es.wikipedia.org/wiki/Anacron>  
<https://es.wikipedia.org/wiki/Syslog>  
<http://www.computerhope.com/unix/signals.htm>  
[https://en.wikipedia.org/wiki/Unix\\_signal](https://en.wikipedia.org/wiki/Unix_signal)  
[https://www.centos.org/docs/5/html/Deployment\\_Guide-en-US/ch-autotasks.html](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-autotasks.html)  
<http://www.alcanceibre.org/staticpages/index.php/configuracion-uso-cron>