

ÍNDICE

1. ADMINISTRACIÓN DE PROCESOS.....	2
1.1 EN MODO GRÁFICO.....	2
1.2 COMANDO PS, TOP Y HTOP.....	3
1.3 PROCESOS EN PRIMER Y SEGUNDO PLANO.....	5
1.4 COMANDO KILL Y KILLALL.....	5
1.5 COMANDOS PARA PRIORIDADES DE LOS PROCESOS.....	6
2. ADMINISTRACIÓN DE SERVICIOS.....	7
2.1 SCRITPS DE SERVICIOS.....	8
2.3 SERVICIOS EN MODO GRÁFICO.....	8
2.3 COMANDOS DE ADMINISTRACIÓN DE SERVICIOS.....	9
3. ADMINISTRAR APLICACIONES AL INICIO.....	9

1. ADMINISTRACIÓN DE PROCESOS

PROCESOS

- Un proceso es una instancia de un programa en ejecución.
- Cada proceso en el momento de su creación se le **asocia**
 - Un **número único** que lo identifica.
 - El **usuario** que lo ejecuta.
 - La **hora** en que comenzó.
 - La **línea de comandos** asociada.
 - Un **estado**: sleep, running, zombie, stopped, etc.
 - Una **prioridad** que indica la facilidad del proceso para acceder a la CPU. Oscila entre -20 y 19, donde -20 es la mayor prioridad.
 - La **terminal** donde fue invocado

1.1 EN MODO GRÁFICO

HERRAMIENTA MONITOR DEL SISTEMA

Procesos Recursos Sistemas de archivos							
Nombre del proceso	Usuario	% CPU	ID	Memoria	Lectura total	Escritura	
at-spi2-registrd	usuario	0	2379	648,0 KiB	N/D		
at-spi-bus-launcher	usuario	0	2300	784,0 KiB	4,0 KiB		
dbus-daemon	usuario	0	2088	2,1 MiB	N/D		
dbus-daemon	usuario	0	2305	464,0 KiB	N/D		
dconf-service	usuario	0	2418	652,0 KiB	4,0 KiB	72,0 KiB	
evolution-addressbook-factory	usuario	0	2424	3,5 MiB	2,3 MiB	36,0 KiB	
evolution-alarm-notify	usuario	0	2542	15,6 MiB	1,2 MiB		
evolution-calendar-factory	usuario	0	2404	4,1 MiB	5,0 MiB		
evolution-source-registry	usuario	0	2396	3,7 MiB	3,5 MiB		
gdm-x-session	usuario	0	2131	292,0 KiB	104,0 KiB		
gjs	usuario	0	2454	4,6 MiB	N/D		
gnome-calendar	usuario	0	4810	14,8 MiB	6,6 MiB		
gnome-keyring-daemon	usuario	0	2084	492,0 KiB	N/D		
gnome-session-binary	usuario	0	2169	1,5 MiB	5,7 MiB		
gnome-session-binary	usuario	0	2322	2,6 MiB	7,6 MiB	4,0 KiB	
gnome-session-ctl	usuario	0	2315	416,0 KiB	20,0 KiB		
gnome-shell	usuario	25	2337	256,0 MiB	11,7 MiB	32,0 KiB	
gnome-shell-calendar-server	usuario	0	2385	2,5 MiB	4,6 MiB		
gnome-system-monitor	usuario	9	4909	15,9 MiB	11,2 MiB	8,0 KiB	

REALIZAR OPERACIONES: CLIC DERECHO

evolution-calendar-factory	usuario	0	1850	4,1 MiB	5,0 MiB	N/D	N/D	N/D	Normal
evolution-source-registry	usuario	0	1842	3,8 MiB	3,4 MiB	N/D	N/D	N/D	Normal
gdm-x-session	usuario	0	1604	648,0 KiB	104,0 KiB	N/D	N/D	N/D	Normal
gjs	usuario	0	1891	4,6 MiB	N/D	N/D	N/D	N/D	Normal
gnome-calendar	usuario	0	2909	14,9 MiB	1,1 MiB	N/D	N/D	N/D	Normal
gnome-keyring-daemon	usuario	0	1531	916,0 KiB					N/D
gnome-session-binary	usuario	0	1624	1,5 MiB					N/D
gnome-session-binary	usuario	0	1768	2,6 MiB					N/D
gnome-session-ctl	usuario	0	1761	420,0 KiB					N/D
gnome-shell	usuario	0	1783	240,7 MiB					N/D
gnome-shell-calendar-server	usuario	0	1831	2,5 MiB					N/D
gnome-system-monitor	usuario	1	3612	16,3 MiB					N/D
goa-daemon	usuario	0	1573	5,9 MiB					N/D
goa-identity-service	usuario	0	1581	1,0 MiB					N/D
nsd-a11y-settings	usuario	0	1901	612,0 KiB	N/D	N/D	N/D	N/D	Normal

1.2 COMANDO PS, TOP Y HTOP

PS

Ver los procesos y sus características

```
usuario@server2:~$ ps
  PID TTY          TIME CMD
 2679 pts/0        00:00:00 bash
 2699 pts/0        00:00:00 ps
```

Cada proceso se muestra su:

- ID (identificación o numero)
- la terminal desde donde se invocó
- el tiempo de CPU que se le ha asignado hasta el momento
- el comando que lo desencadenó.

Algunas **opciones**:

- x** : todos los procesos del usuario actual.
- a** : todos los procesos de todos los usuarios.
- l** : formato más largo (muestra más información).
- u** : formato orientado a usuario.
- f** : relaciones jerárquicas entre los procesos.
- e** : entorno de cada proceso.

Ejemplos:

ps aux

```
usuario@server2:~$ ps aux
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root             1   0.0   0.6 167576 11628 ?        Ss   09:25   0:01 /sbin/init ma
root             2   0.0   0.0      0      0 ?        S    09:25   0:00 [kthreadd]
root             3   0.0   0.0      0      0 ?        I<   09:25   0:00 [rcu_gp]
root             4   0.0   0.0      0      0 ?        I<   09:25   0:00 [rcu_par_gp]
root             6   0.0   0.0      0      0 ?        I<   09:25   0:00 [kworker/0:0H
root             9   0.0   0.0      0      0 ?        I<   09:25   0:00 [mm_percpu_wq
```

ps e

```
usuario@server2:~$ ps e
  PID TTY          STAT TIME  COMMAND
 2131 tty2      Ssl+   0:00  /usr/lib/gdm3/gdm-x-session --run-script env GNOME_
 2138 tty2      Sl+    0:02  /usr/lib/xorg/Xorg vt2 -displayfd 3 -auth /run/user
 2169 tty2      Sl+    0:00  /usr/libexec/gnome-session-binary --systemd --syste
 2679 pts/0      Ss     0:00  bash GJS_DEBUG_TOPICS=JS ERROR;JS LOG SSH_AUTH_SOCK
 3715 pts/0      R+     0:00  ps e SHELL=/bin/bash SESSION_MANAGER=local/server2:
```

ps xf

```
usuario@server2:~$ ps xf
  PID TTY          STAT TIME  COMMAND
 2131 tty2      Ssl+   0:00  /usr/lib/gdm3/gdm-x-session --run-script env GNOME_
 2138 tty2      Sl+    0:03  \_ /usr/lib/xorg/Xorg vt2 -displayfd 3 -auth /run/
 2169 tty2      Sl+    0:00  \_ /usr/libexec/gnome-session-binary --systemd --s
 2282 ?        Ss     0:00  \_ /usr/bin/ssh-agent /usr/bin/im-launch env G
 2276 ?        S      0:00  /usr/bin/VBoxClient --vmsvga
 2279 ?        Sl     0:00  \_ /usr/bin/VBoxClient --vmsvga
 2269 ?        S      0:00  /usr/bin/VBoxClient --draganddrop
```


TOP

Ver los procesos dinámicamente de forma interactiva. También muestra algunas estadísticas generales del sistema:

```
top - 13:07:56 up 7 min, 1 user, load average: 1,41, 1,83, 1,00
Tareas: 205 total, 2 ejecutar, 203 hibernar, 0 detener, 0 zombie
%Cpu(s): 0,7 us, 3,5 sy, 0,0 ni, 0,0 id, 94,8 wa, 0,0 hi, 1,0 si, 0,0 st
MiB Mem : 2165,0 total, 72,4 libre, 1028,6 usado, 1064,0 búfer/caché
MiB Intercambio: 2140,0 total, 1920,1 libre, 219,9 usado. 936,9 dispon Mem
```

PID	USUARIO	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	HORA+	ORDEN
5996	root	25	5	72416	56924	3084	D	2,3	2,6	0:12.46	dpkg
2585	usuario	20	0	4045424	224524	49132	S	1,0	10,1	0:13.29	gnome-shell
3900	usuario	20	0	574232	41604	28996	S	0,7	1,9	0:01.11	gnome-terminal-
9908	usuario	20	0	21976	4256	3420	R	0,7	0,2	0:00.07	top
14	root	20	0	0	0	0	R	0,3	0,0	0:01.25	rcu_sched
83	root	0	-20	0	0	0	I	0,3	0,0	0:01.86	kworker/0:1H-kblockd
628	message+	20	0	11196	5980	3580	S	0,3	0,3	0:01.44	dbus-daemon
2026	usuario	20	0	162180	1876	1622	S	0,2	0,1	0:01.05	VBoxClient

HTOP

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
2337	usuario	20	0	3598M	338M	125M	S	2.1	19.6	0:27.03	/usr/bin/gnome-
4392	usuario	20	0	8100	4080	3380	S	1.4	0.2	0:01.10	htop
4429	usuario	20	0	8068	4128	3436	R	0.7	0.2	0:00.16	htop
2270	usuario	20	0	149M	2696	2328	S	0.7	0.2	0:30.04	/usr/bin/VBoxCl
2671	usuario	20	0	793M	52292	39548	S	0.0	2.9	0:02.58	/usr/libexec/gn
2277	usuario	20	0	149M	2696	2328	S	0.0	0.2	0:30.04	/usr/bin/VBoxCl
2138	usuario	20	0	249M	71680	46148	S	0.0	4.0	0:05.45	/usr/lib/xorg/X
2167	usuario	20	0	249M	71680	46148	S	0.0	4.0	0:00.98	/usr/lib/xorg/X
1	root	20	0	163M	11628	8392	S	0.0	0.7	0:01.90	/sbin/init mayb
1110	root	20	0	350M	2708	2332	S	0.0	0.2	0:02.20	/usr/sbin/VBoxS
1114	root	20	0	350M	2708	2332	S	0.0	0.2	0:01.54	/usr/sbin/VBoxS
345	root	19	-1	70124	18680	16984	S	0.0	1.1	0:00.91	/lib/systemd/sy
367	root	20	0	2488	572	508	S	0.0	0.0	0:00.00	bpfilter_umh
396	root	20	0	24004	7496	3972	S	0.0	0.4	0:01.30	/lib/systemd/sy
613	root	RT	0	273M	17996	8204	S	0.0	1.0	0:00.15	/sbin/multipath
614	root	RT	0	273M	17996	8204	S	0.0	1.0	0:00.00	/sbin/multipath
615	root	RT	0	273M	17996	8204	S	0.0	1.0	0:00.02	/sbin/multipath

F1 Help F2 Setup F3 Search F4 Filter F5 Tree F6 SortBy F7 Nice - F8 Nice + F9 Kill F10 Quit

1.3 PROCESOS EN PRIMER Y SEGUNDO PLANO

Podemos ejecutar los procesos en:

- **foreground** (primer plano) o
- **background** (segundo plano).

EJECUTAR UN PROCESO EN BACKGROUND

AÑADIR AL FINAL DEL PROCESO &

Ejemplo:

updatedb &

MODIFICAR EL ESTADO DE UN PROCESO EN FOREGROUND

• **Ctrl-c** : trata de **interrumpir** el proceso en **foreground**. Si es efectivo, el proceso finaliza su ejecución (se le mata).

• **Ctrl-z** : trata de **detener** el proceso en **foreground**. Si es efectivo el proceso continúa activo aunque deja de acceder al procesador (está detenido y pasa a **background**).

COMANDO JOBS

Visualiza los procesos detenidos o en background con los comandos asociados y un identificador.

Ejemplo:

jobs

El resultado que se obtiene es:

```
[1] Running sleep 10000 &
[2]- Stopped cp /var/log/messages /tmp
[3]+ Stopped updatedb
```

COMANDOS BG Y FG

Los procesos detenidos se pueden llevar al **background**.

Los procesos en background pueden trasladarse al **foreground**.

Se indica el **identificador especial** del proceso.

Si no se especifica se asumirá el último detenido o llevado al background.

Ejemplos:

1. Envía a segundo plano, el proceso detenido 2: **bg 2**
2. Trae un proceso de segundo plano a primer plano: **fg**
3. Si se comenzara a ejecutar un proceso y este se demora mucho y no interesan sus resultados se puede:
 - detener y **enviarlo al background** haciendo **Ctrl-z**.
 - **volver a traerlo a 1º plano** con **fg**.

1.4 COMANDO KILL Y KILLALL

COMANDO KILL

Envía señales con diversos significados. Existen muchos tipos de señales. Para verlas todas: **kill -l**

Por **defecto** kill envía la señal **15 (TERM)** al proceso que debe terminar.

La señal **9 (KILL)** lo finaliza forzosamente.

Ejemplos:

kill 1000 # envía la señal de terminar al proceso 1000

kill -9 10101 # envía la señal 9 (finalizar forzosamente) al proceso 10101

COMANDO KILLALL

Envía señales a los procesos **a través de sus nombres**.

El nombre de un proceso no es único, de ahí la utilidad de este comando.

Ejemplo:

killall -9 ssh

1.5 COMANDOS PARA PRIORIDADES DE LOS PROCESOS

COMANDO NICE

Sin argumentos visualiza la **prioridad** asignada por defecto a los procesos del usuario actual.

Se puede indicar la nueva prioridad precedida del signo “-“.

Si no se indica la prioridad se incrementa en 10 la por defecto.

Sólo el usuario **root** puede asignar a sus procesos prioridades con valores inferiores a cero.

Ejemplos:

nice tar cvf /tmp/etc.tgz /etc # **incrementa en 10 la prioridad por defecto del comando**

nice - 10 updatedb # **ejecuta un comando con prioridad 10**

nice - -10 updatedb # **ejecuta un comando con prioridad -10**

COMANDO RENICE

Reajusta la prioridad de un proceso en ejecución.

El valor de la prioridad no va precedido por el signo “-“.

Las prioridades de los procesos **sólo se pueden disminuir**, nunca aumentar, con excepción de root que puede hacerlo indistintamente.

Ejemplos:

renice -19 1001 # **ajusta la prioridad a -19 del proceso 1001**

renice 1 602 # **ajusta la prioridad de un proceso a 1**

renice 10 -u pepe # **ajusta a 10 la prioridad de todos los procesos del usuario pepe**

renice 5 -g ppp uucp # **ajusta a 5 la prioridad de todos los procesos de usuarios miembros de grupos ppp y uucp.**

COMANDO NOHUP

Si la sesión termina en un terminal (shell), los procesos activos recibirán la señal HUP.

Si se desea que los procesos continúen, aunque termine la sesión del terminal se utiliza NOHUP

Por defecto NOHUP reduce la prioridad en 5.

Ejemplo:

nohup gran_calculo &
logout

Se ejecuta un proceso gran_calculo en segundo plano.

Cuando hacemos logout mataremos nuestro shell y a todos sus hijos, con lo que mataríamos gran_calculo.

Pero al haber lanzado gran_calculo con **nohup**, seguirá ejecutándose aunque nos salgamos del sistema.

2. ADMINISTRACIÓN DE SERVICIOS

Los **servicios (demonios)**, son **programas que se ejecutan en segundo plano** para ofrecer una función concreta.

Algunos de los servicios más frecuentes en *Linux*:

Servicio	Descripción	Servicio	Descripción
atd	Ejecuta las tareas programadas con el comando at	cups	Servidor de impresión
crond	Ejecuta las tareas programadas del sistema.	dhcpcd	Servidor DHCP
netfs	Monta sistemas de archivos en red	httpd	Servidor de páginas web Apache
network	Activa las interfaces de red del sistema	innd	Servidor de noticias local
Network Manager	Herramienta de administración de conexiones de red	iptables	Cortafuegos del sistema
nfs	Servidor de ficheros en red	mdmonitor	Comprueba los sistemas RAID
ntp	Sincroniza la hora en la red (Network Time Protocol)	mysqld	Servidor de base de datos MySQL
rpc	Permite la ejecución remota de procesos (Remote Process Call)	named	Servidor DNS
rsyslog	Almacena los sucesos del sistema	sendmail	Servidor de correo electrónico
sshd	Habilita los servicios seguros de red (Secure Shell)	smb	Comparte archivos e impresoras con Windows
xinetd	Activa servicios de red	squid	Servidor proxy

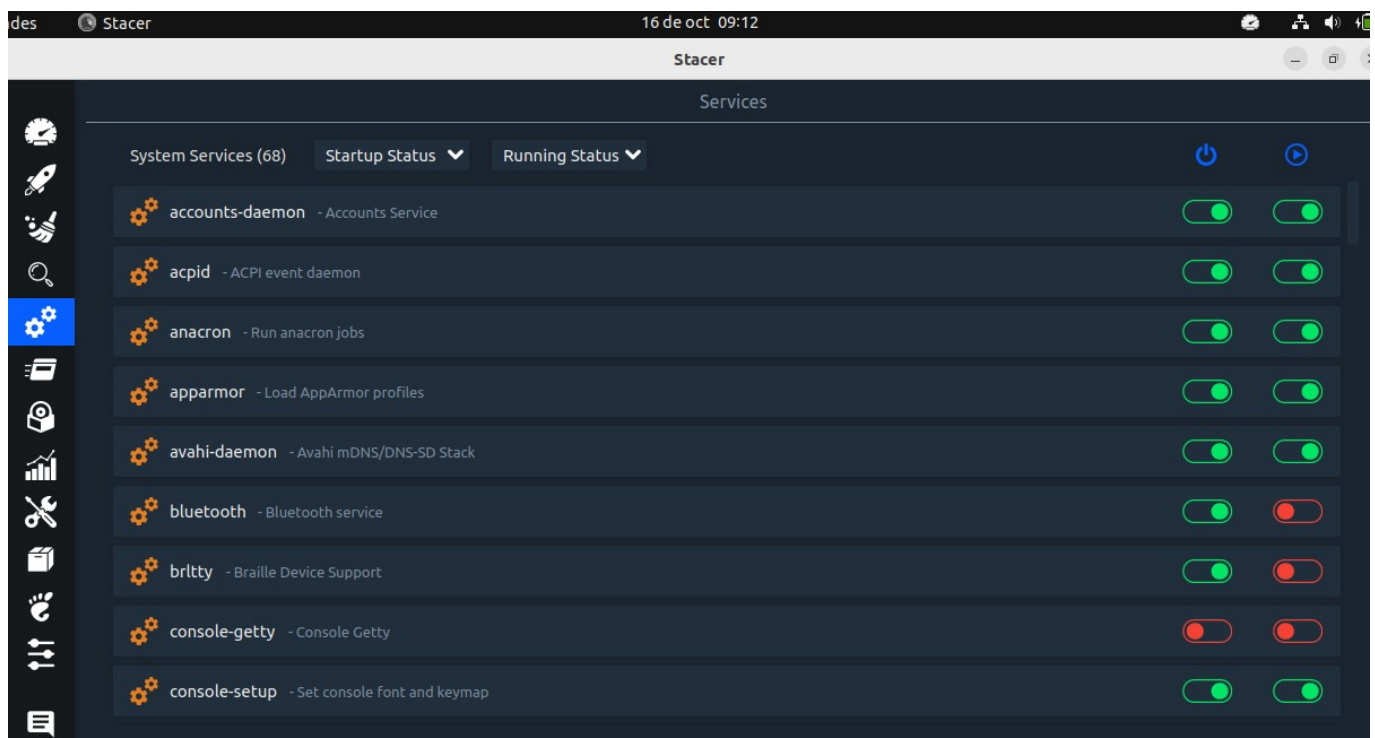
2.1 SCRIPTS DE SERVICIOS

Cuando se instala un programa que lleva asociado algún servicio, el **script** necesario para controlarlo se guarda en el directorio **/etc/init.d**.

```
usuario@server2:~$ ls /etc/init.d
acpid          hwclock.sh      pulseaudio-enable-autospawn
alsa-utils     irqbalance      rsync
anacron        iscsid          rsyslog
apparmor       kerneloops     saned
appport        keyboard-setup.sh screen-cleanup
atd            kmod           speech-dispatcher
avahi-daemon   lvm2           spice-vdagent
bluetooth      lvm2-lvmpolld  ssh
console-setup.sh multipath-tools udev
cron           network-manager ufw
cryptdisks     open-iscsi     unattended-upgrades
cryptdisks-early open-vm-tools  uidd
cups           openvpn        whoopsie
cups-browsed   plymouth       x11-common
dbus           plymouth-log   xrdp
gdm3           pppd-dns
grub-common    procs
usuario@server2:~$
```

2.3 SERVICIOS EN MODO GRÁFICO

Instalar la aplicación **Stacer** (en terminal o mediante Synaptic)



2.3 COMANDOS DE ADMINISTRACIÓN DE SERVICIOS

¡¡NECESITA sudo!!

SERVICE	SYSTEMCTL
Ver el estado de todos los servicios	
service --status-all	systemctl status
Inicia la ejecución de un servicio.	
service cron start	systemctl start ufw
Detiene un servicio que se está ejecutando.	
service cron stop	systemctl stop ufw
Reiniciar servicio(para e inicia)	
service cron restart	systemctl restart ufw
Ver el estado de un servicio.	
service cron status	systemctl status ufw

3. ADMINISTRAR APLICACIONES AL INICIO

