

ADMINISTRACIÓN DE SISTEMAS OPERATIVOS

TEMA 2

Ejercicios de procesos

2CFSH

GERARDO CANO

ASO

Capítulo 8: Monitoreo y Gestión de Procesos Linux	2
1. Estados de los Procesos (Process States)	2
2. Comandos de Monitoreo y Visualización	3
a) ps (Process Status) - Vista Estática	3
b) top / htop - Vista Dinámica (Tiempo Real)	3
c) pstree - Jerarquía de Procesos.....	3
3. Comandos de Gestión y Control de Trabajos.....	4
a) kill, pkill, killall - Envío de Señales	4
b) Job Control (Trabajos en Segundo y Primer Plano)	4
c) nice y renice - Prioridad de Ejecución	4
TRABAJO DE LABORATORIO:	5

CAPÍTULO 8: MONITOREO Y GESTIÓN DE PROCESOS LINUX

1. ESTADOS DE LOS PROCESOS (PROCESS STATES)

El estado de un proceso (visible en columnas como STAT de ps o S de top) indica lo que el proceso está haciendo en un momento dado:

Estado	Símbolo	Descripción
Running	R	El proceso se está ejecutando o está listo para ejecutarse y esperando su turno en la CPU.
Sleeping	S	El proceso está inactivo, esperando que ocurra un evento (ej. entrada/salida de teclado o disco).
Uninterruptible Sleep	D	El proceso está en espera de una operación de E/S que no se puede interrumpir. Es un estado problemático (a menudo relacionado con fallas de hardware o red).
Stopped	T	El proceso ha sido suspendido, ya sea por una señal de Job Control (SIGTSTP / Ctrl+Z) o por un
Zombie	Z	El proceso ha terminado su ejecución, pero su entrada en la tabla de procesos aún existe. Está esperando que su proceso padre recoja su estado de salida.

2. COMANDOS DE MONITOREO Y VISUALIZACIÓN

Estos comandos permiten obtener instantáneas (estáticas) o vistas en tiempo real (dinámicas) del sistema:

a) PS (PROCESS STATUS) - VISTA ESTÁTICA

Muestra una instantánea de los procesos activos en el momento de la ejecución.

Comando	Descripción
<code>ps aux</code>	Muestra todos los procesos de todos los usuarios (formato BSD). Útil para ver el uso de CPU y memoria.
<code>ps -ef</code>	Muestra todos los procesos en formato estándar (formato System V). Útil para ver el PID y PPID.
<code>ps -u <usuario></code>	Muestra todos los procesos propiedad de un usuario específico.
<code>ps fax</code>	Muestra el árbol de procesos de forma jerárquica (requiere la opción f).

b) TOP / HTOP - VISTA DINÁMICA (TIEMPO REAL)

Muestran los procesos en tiempo real, ordenados por uso de CPU.

Comando	Descripción
<code>top</code>	Muestra información de procesos en tiempo real. Permite ordenar, matar procesos y ver el uso total de recursos.
<code>htop</code>	Una alternativa más amigable y visual que <code>top</code> . Permite desplazarse vertical y horizontalmente y usar el mouse.
Dentro de <code>top</code>	La tecla <code>k</code> permite "matar" (kill) un proceso, y la tecla <code>r</code> permite cambiar su prioridad (renice).

c) PSTREE - JERARQUÍA DE PROCESOS

Muestra los procesos en formato de árbol, revelando las relaciones padre-hijo.

3. COMANDOS DE GESTIÓN Y CONTROL DE TRABAJOS

Estos comandos se utilizan para manipular procesos, cambiar su prioridad o enviarlos a segundo plano.

a) KILL, PKILL, KILLALL - ENVÍO DE SEÑALES

Se utilizan para enviar señales a los procesos, siendo las más comunes SIGTERM (15) y SIGKILL (9).

Comando	Descripción
kill <PID>	Envía la señal por defecto (SIGTERM/15) al Proceso ID. Pide cortésmente al proceso que termine.
kill -9 <PID>	Envía la señal SIGKILL (9). Fuerza la terminación inmediata del proceso (no puede ser ignorada).
pkill <nombre>	Envía una señal a los procesos basándose en su nombre.
killall <nombre>	Envía una señal a <i>todos</i> los procesos que coincidan exactamente con el nombre.

b) JOB CONTROL (TRABAJOS EN SEGUNDO Y PRIMER PLANO)

Permite gestionar comandos que se ejecutan desde el *shell* actual (Bash).

Comando	Descripción
<comando> &	Ejecuta un comando inmediatamente en segundo plano (BG).
Ctrl + Z	Suspende un proceso que se ejecuta en primer plano, enviándolo a estado Stopped (T).
jobs	Lista los trabajos suspendidos o en segundo plano en el <i>shell</i> actual.
fg <jobID>	Trae un trabajo (listado por jobs) al primer plano (FG) para interactuar con él.
bg <jobID>	Envía un trabajo suspendido (T) al segundo plano (BG).

c) NICE Y RENICE - PRIORIDAD DE EJECUCIÓN

Controlan el valor de *niceness* (amabilidad), que influye en la prioridad que el kernel asigna a un proceso.

Concepto	Rango de Valor
Nice Value (VN)	De -20 (máxima prioridad, menos amable) a 19 (mínima prioridad, más amable). El valor por defecto es 0.

Comando	Descripción
nice -n <VN> <comando>	Ejecuta un nuevo comando con el valor de <i>niceness</i> especificado.
renice <VN> -p <PID>	Cambia el valor de <i>niceness</i> de un proceso que ya se está ejecutando.

Nota: Solo el usuario **root** puede establecer valores de *niceness* negativos (mayor prioridad).

TRABAJO DE LABORATORIO:

1. Script Base: process101

Este es el contenido exacto del script que genera la carga de CPU.

Ruta del archivo: /home/student/bin/process101 **Iteraciones:** 50,000

```
#!/bin/bash
while true; do
    var=1
    while [[ var -lt 50000 ]]; do
        var=$((var+1))
    done
    sleep 1
done
```

2. Instrucciones y Comandos

A. Preparación del Entorno (En workstation)

1. Iniciar sesión en workstation como student (contraseña: student).
2. *Iniciar el laboratorio:*

```
$ lab start processes-review
```

3. Abrir dos terminales y conectarse a serverb en ambas (serán izquierda y derecha).

```
$ ssh student@serverb
```

```
# Contraseña: student
```

B. Creación y Ejecución de process101 (En serverb)

Terminal Izquierda (Ejecución)

1. Crear, hacer ejecutable el script y ejecutarlo:

```
[student@serverb ~]$ vim ~/bin/process101 # Pegar contenido
```

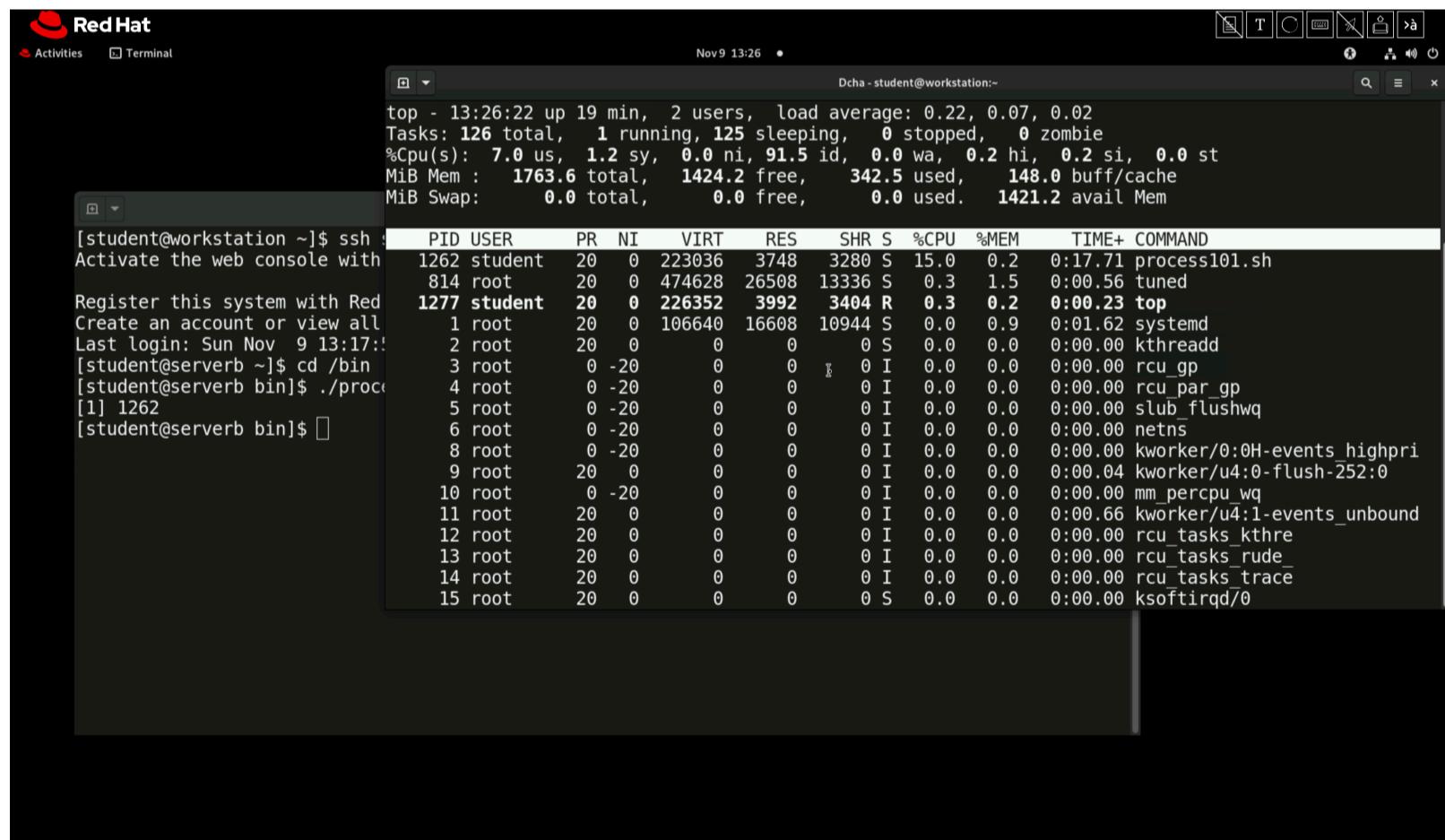
```
[student@serverb ~]$ chmod +x ~/bin/process101.sh
```

```
[student@serverb ~]$ ./process101.sh
```

El sistema mostrará el ID del proceso (PID).

C. Alternar Vistas y Guardar Configuración de top

(Realizar en la Terminal Derecha mientras top está activo)



Busca el proceso process101.

Verifica su PID y el uso de CPU (debería ser 10% - 15%).

Altera la visualización con las siguientes teclas (y vuelve a la pantalla inicial):

- Presiona **1** para alternar la línea de carga media/uptime.
- Presiona **H** para alternar la visualización de hilos (subprocesos).
- Presiona **m** para alternar la visualización de la memoria.

Desactiva la negrita y guarda la configuración:

- Presiona **y** para alternar la negrita de la columna de clasificación (si está activada, se verá un * a su lado).
- Presiona **w** (W mayúscula) para guardar la configuración en `~/.config/procps/toprc`.

D. Aumento de Carga (process102 y process103)

(Todos los comandos en la Terminal Izquierda)

Crear y ejecutar process102 (100,000 iteraciones):

```
$ cp ~/bin/process101 ~/bin/process102
$ sed -i 's/50000/100000/' ~/bin/process102.sh
[student@serverb ~]$ ./bin/process102.sh &
```

Verificar en top (Derecha): process102.sh

Crear y ejecutar process103 (800,000 iteraciones):

```
[student@serverb ~]$ cp ~/bin/process101 ~/bin/process103
[student@serverb ~]$ sed -i 's/50000/800000/' ~/bin/process103
[student@serverb ~]$ ./bin/process103 &
```

Verificar en top (Derecha): Esperar unos minutos hasta que el promedio de carga (Load Average) sea **superior a 1**.

E. Gestión de Procesos (Suspender y Finalizar)

(Todos los comandos en la Terminal Izquierda)

Cambiar a root:

```
[student@serverb ~]$ su -
# Contraseña: redhat
```

Izquierda - student@serverb:-

```
sed: couldn't open temporary file ./sedlPk8re: Permission denied
[student@serverb bin]$ sudo !!
sudo sed -i 's/50000/100000/' process102.sh
[student@serverb bin]$ cat process102.sh
#!/bin/bash
while true; do
    var=1
    while [[ var -lt 100000 ]]; do
        var=$((var+1))
    done
    sleep 1
done
[student@serverb bin]$ sudo sed -i 's/50000/80000/' process103.sh
[student@serverb bin]$ cat process103.sh
#!/bin/bash
while true; do
    var=1
    while [[ var -lt 80000 ]]; do
        var=$((var+1))
    done
    sleep 1
done
[student@serverb bin]$
```

Derecha - student@workstation:-

```
top - 12:26:21 up 29 min, 2 users, load average: 0.03, 0.02, 0.0
Tasks: 229 total, 1 running, 227 sleeping, 1 stopped, 0 zomb
%Cpu(s): 0.3 us, 0.1 sy, 0.0 ni, 99.6 id, 0.0 wa, 0.0 hi, 0.
MiB Mem : 5657.4 total, 1774.0 free, 2014.0 used, 2190.4 b
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 3643.4 a
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM
1543	student	20	0	5078256	299580	135588	S	0.7	5.2
1279	student	20	0	840228	97860	58532	S	0.3	1.7
1943	root	20	0	271764	40416	8520	S	0.3	0.7
2609	student	20	0	693340	57132	38700	S	0.3	1.0
3114	student	20	0	226484	4260	3416	R	0.3	0.1
1	root	20	0	173120	17228	10840	S	0.0	0.3
2	root	20	0	0	0	0	S	0.0	0.0
3	root	0	-20	0	0	0	I	0.0	0.0
4	root	0	-20	0	0	0	I	0.0	0.0
5	root	0	-20	0	0	0	I	0.0	0.0
6	root	0	-20	0	0	0	I	0.0	0.0
7	root	20	0	0	0	0	I	0.0	0.0
8	root	0	-20	0	0	0	I	0.0	0.0
9	root	20	0	0	0	0	I	0.0	0.0
10	root	0	-20	0	0	0	I	0.0	0.0
12	root	20	0	0	0	0	I	0.0	0.0

*Suspender **process101** (asumiendo que es el trabajo 1):*

```
[root@serverb ~]# kill -STOP <PID>
```

Observar en `top` (Derecha): El estado (columna S) de `process101` cambia a T (Stopped).

```
Dcha - student@workstation:~
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1242	student	20	0	223036	3712	3244	S	28.2	0.2	2:54.45	process102.sh
1251	student	20	0	223036	3716	3252	S	22.9	0.2	2:45.89	process103.sh
1765	student	20	0	223036	3744	3280	S	19.3	0.2	0:59.52	process101.sh

```
lizq - student@workstation:~
```

```
[student@serverb bin]$ jobs
[1]+  Stopped                  ./process101.sh
[2]  Running                   ./process102.sh &
[3]  Running                   ./process103.sh &
[4]-  Running                   ./process101.sh &
```

```
[student@serverb bin]$
```

```
systemd
kthreadd
rcu_gp
rcu_par_gp
slub_flushhwq
netns
kworker/0:0H-events_highpri
mm_percpu_wq
kworker/u4:1-events_unbound
rcu_tasks_kthre
rcu_tasks_rude_
rcu_tasks_trace
ksoftirqd/0
pr/ttyS0
```

Listar trabajos restantes:

```
[root@serverb ~]# jobs
```

Reanudar process101:

```
[root@serverb ~]# kill -CONT <PID>
```

Observar en top (Derecha): El estado de process101 vuelve a R (Running).

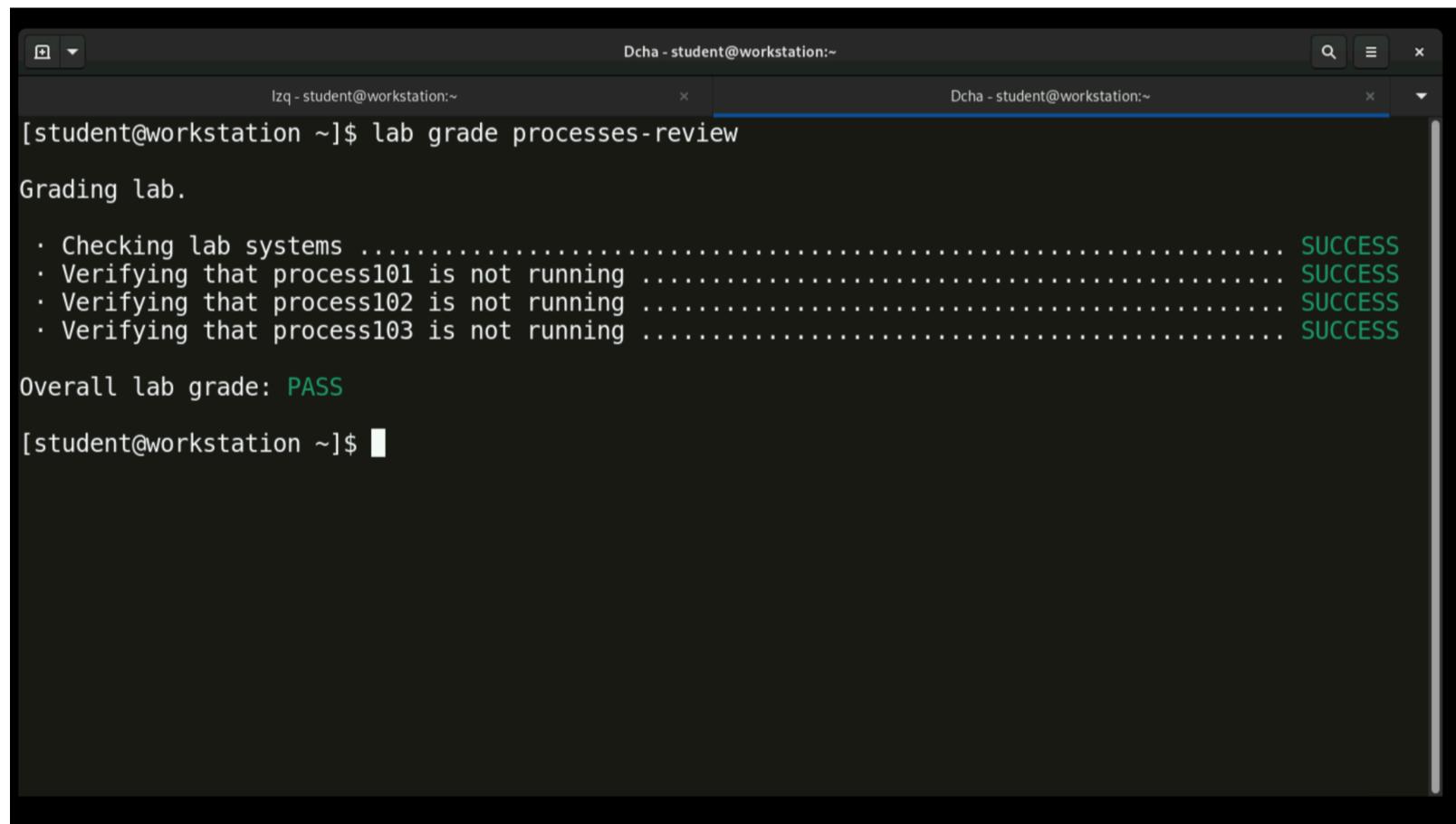
Finalizar los procesos (trabajos 1, 2 y 3):

```
[root@serverb ~]# kill %1 %2 %3
```

Verificar en top (Derecha): Los procesos deben desaparecer.

Regresar a student:

```
[root@serverb ~]# exit
```



The screenshot shows a terminal window with three tabs. The active tab is titled 'Dcha - student@workstation:~'. The command [student@workstation ~]\$ lab grade processes-review is run, followed by the output:

```
[student@workstation ~]$ lab grade processes-review
Grading lab.

· Checking lab systems ..... SUCCESS
· Verifying that process101 is not running ..... SUCCESS
· Verifying that process102 is not running ..... SUCCESS
· Verifying that process103 is not running ..... SUCCESS

Overall lab grade: PASS
[student@workstation ~]$
```