7.1 Procesos

Objetivos

Tras finalizar esta sección, los estudiantes deberían poder realizar lo siguiente:

- Definir el ciclo de vida de un proceso.
- Definir los estados de un proceso.
- Ver e interpretar listas de procesos.

Procesos

Un *proceso* es una instancia de un programa ejecutable que se inició y se encuentra en funcionamiento. Un proceso consta de lo siguiente:

- un espacio de direcciones que incluye la memoria asignada;
- características de seguridad, que incluyen credenciales y privilegios de propiedad;
- uno o más subprocesos de ejecución de código de programa; y
- el estado del proceso.

El *entorno* de un proceso incluye lo siguiente:

- variables locales y globales;
- un contexto de programación actual; y
- recursos asignados del sistema, como descriptores de archivos y puertos de red.

Un proceso (*principal*) existente duplica su propio espacio de direcciones (**bifurcación**) para crear una nueva estructura de proceso (*secundaria*). Se asigna una *identificación de proceso* (PID) única a cada proceso nuevo para su rastreo y por motivos de seguridad. La PID y la *identificación del proceso principal* (PPID) son elementos del entorno del proceso nuevo. Cualquier proceso puede crear un proceso secundario. Todos los procesos derivan del primer proceso de sistemas, que es **systemd**(1) en un sistema Red Hat Enterprise Linux 7.

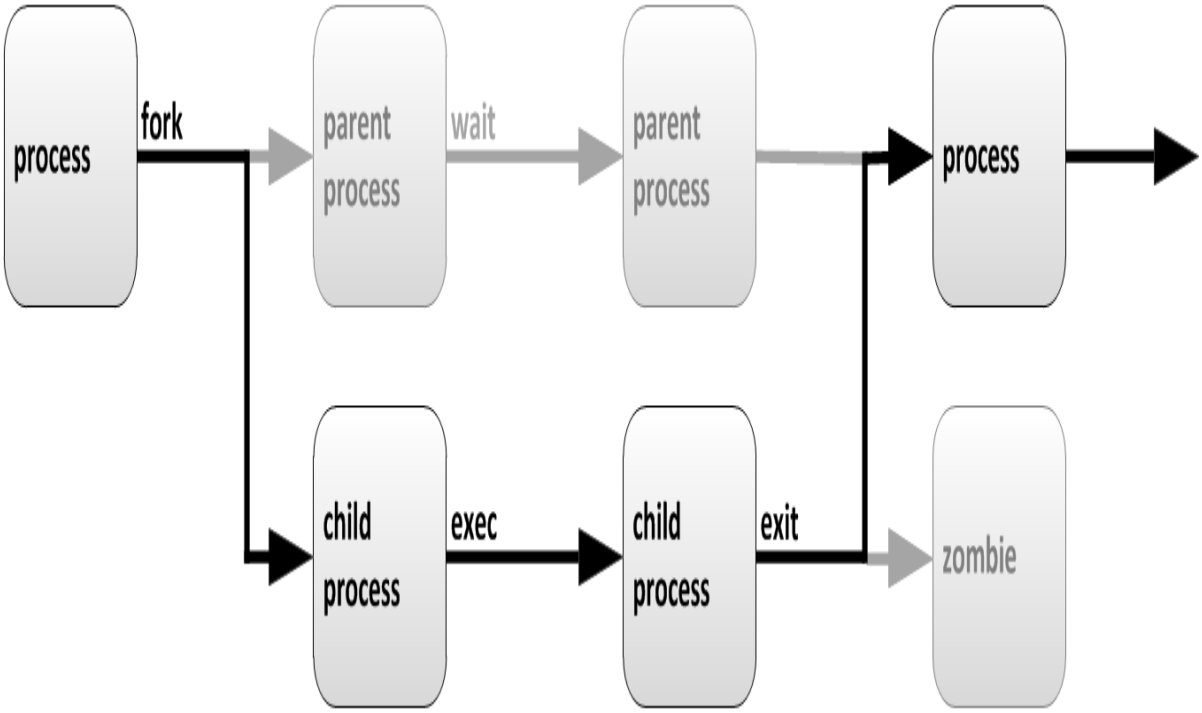

Figura 7.1: Ciclo de vida de un proceso

A través del procedimiento de **bifurcación**, un proceso secundario hereda identidades de seguridad, descriptores de archivos actuales y anteriores, privilegios de recursos y puertos, variables de entorno y código de programa. Luego, un proceso secundario puede **ejecutar** su propio código de programa. Normalmente, un proceso principal se encuentra *durmiendo* mientras se ejecuta el proceso secundario, lo que establece que una solicitud (**espera**) se señale cuando finalice el proceso secundario. Tras su finalización, el proceso secundario ya ha cerrado o descartado sus recursos y su entorno; el resto del proceso se conoce como *zombie*. El proceso principal, que se señala como activo una vez que finaliza el proceso secundario, limpia la estructura restante y continúa con la ejecución de su propio código de programa.

Estados de los procesos

En un sistema operativo de funciones múltiples, cada CPU (o núcleo de CPU) puede trabajar en un proceso en un momento dado. Mientras se ejecuta un proceso, sus requisitos inmediatos en cuanto a asignación de recursos y tiempo de la CPU cambian. Los procesos reciben un *estado*, que cambia en la medida en que las circunstancias así lo exijan.

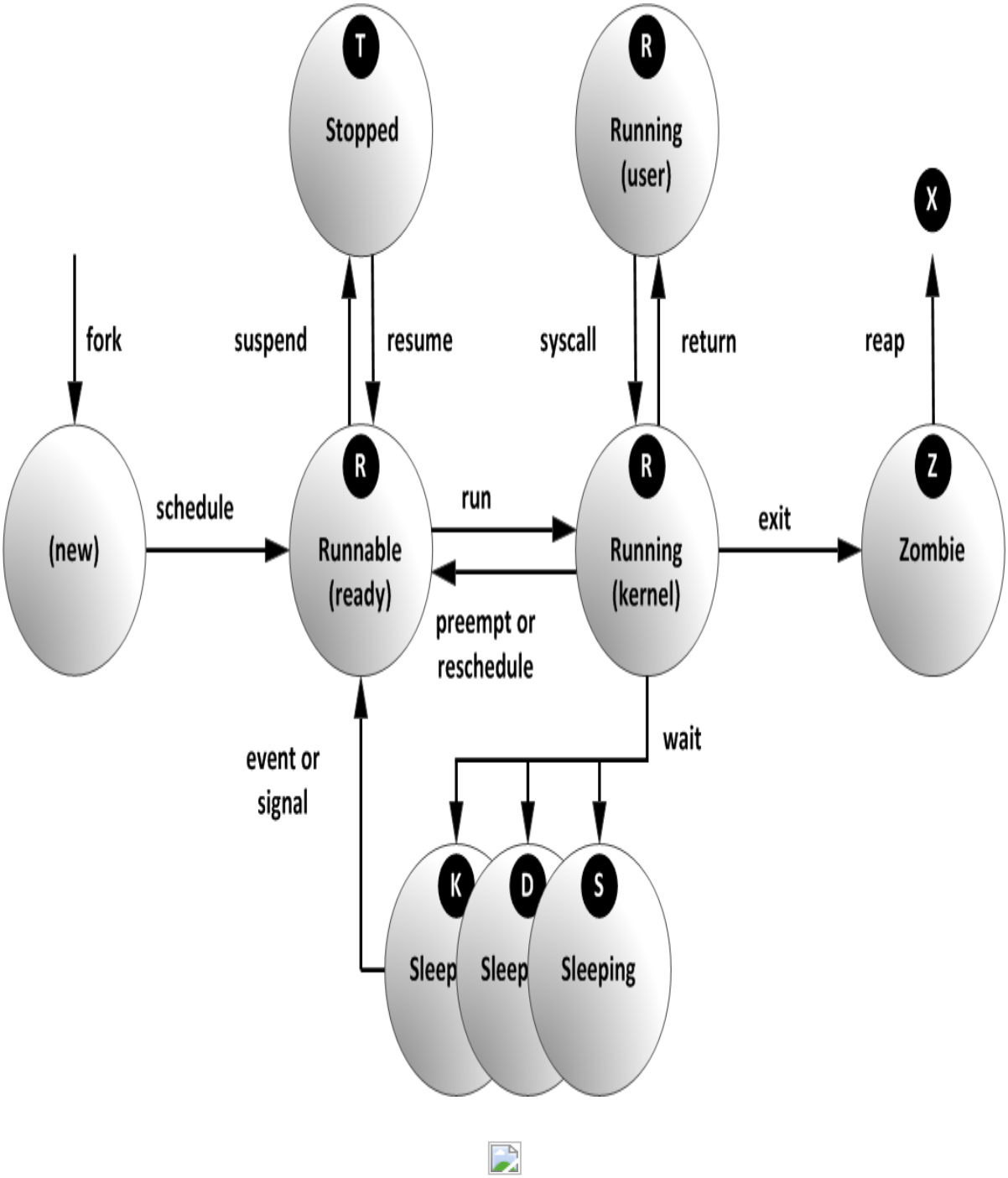

Figura 7.2: Estados de los procesos de Linux

Los estados de los procesos de Linux se ilustran en el diagrama anterior y se describen en la siguiente tabla.

Estados de los procesos de Linux

Nombre	Indicador	Nombre y descripción del estado definido por el núcleo

En ejecución	R	TASK_RUNNING: El proceso se está ejecutando en una CPU o se encuentra en espera de su ejecución. El proceso puede estar ejecutando rutinas del usuario o rutinas del núcleo (llamadas del sistemas); también puede estar en cola y listo cuando esté en el estado *En ejecución* (o *Ejecutable*).
En espera	S	TASK_INTERRUPTIBLE: El proceso se encuentra en espera de que se dé cierta condición, como una solicitud de hardware, el acceso a un recurso del sistema o una señal. Cuando un evento o una señal cumple con la condición, el proceso regresa al estado *En ejecución*.
	D	TASK_UNINTERRUPTIBLE: Este proceso también se encuentra *En espera*, pero a diferencia del estado `S`, no responderá a las señales enviadas. Solo se utiliza en ciertas condiciones en las que la interrupción del proceso puede dar lugar a un estado imprevisto del dispositivo.
	K	TASK_KILLABLE: Igual al estado `D` ininterrumpido, solo que modificado para permitir que la tarea en espera responda a una señal de anulación (salida completa). Las utilidades con frecuencia muestran procesos *anulables* como procesos con estado `D`.
Detenido	T	TASK_STOPPED: El proceso se ha *Detenido* (suspendido), generalmente porque otro usuario u otro proceso lo señalizó. Otra señal puede hacer que el proceso continúe (se reanude) y regrese al estado *En ejecución*.

	T	TASK_TRACED: Un proceso que se está depurando también se encuentra temporalmente *Detenido* y comparte el mismo indicador de estado T.
	Z	EXIT_ZOMBIE: Un proceso secundario señala su proceso principal cuando finaliza. Todos los recursos, menos la PID, se liberan.
Zombie	X	EXIT_DEAD: Cuando el proceso principal limpia (*obtiene*) la estructura del proceso secundario restante, el proceso se libera completamente. Este proceso nunca se observará en utilidades de listas de procesos.

Listas de procesos

El comando **ps** se utiliza para elaborar una lista de los procesos actuales. El comando puede proporcionar información detallada de los procesos, que incluye:

- la identificación del usuario (UID) que determina los privilegios del proceso;
- la identificación del proceso (PID) única;
- la CPU y el tiempo real empleado;
- la cantidad de memoria que el proceso ha asignado en diversas ubicaciones;
- la ubicación del proceso STDOUT, conocido como *terminal de control*; y
- el estado del proceso actual.

Importante

La versión de **ps** de Linux admite tres formatos de opciones, a saber:

- opciones UNIX (POSIX), que pueden agruparse y deben estar precedidas por un guión;
- opciones BSD, que pueden agruparse y no deben usarse con un guion; y

- opciones extensas GNU, que están precedidas por dos guiones.

Por ejemplo, **ps -aux** no es igual a **ps aux**.

Una lista de visualización común (opciones `aux`) muestra todos los procesos, con columnas que serán de interés para los usuarios, e incluye procesos sin una terminal de control. Una lista extensa (opciones `lax`) proporciona detalles más técnicos, pero puede visualizarse más rápidamente porque no realiza la búsqueda de nombre de usuario. La sintaxis de UNIX similar usa las opciones `-ef`para la visualización de todos los procesos.

```
[student@serverX ~]$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.1  0.1  51648  7504 ?        Ss   17:45   0:03 /usr/lib/systemd/syst
root         2  0.0  0.0      0     0 ?        S    17:45   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    17:45   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   17:45   0:00 [kworker/0:0H]
root         7  0.0  0.0      0     0 ?        S    17:45   0:00 [migration/0]
-- salida truncada --
[student@serverX ~]$ ps lax
F   UID   PID  PPID PRI  NI    VSZ   RSS WCHAN  STAT TTY        TIME COMMAND
4     0     1     0  20   0  51648  7504 ep_pol Ss   ?          0:03 /usr/lib/systemd/
1     0     2     0  20   0      0     0 kthrea S    ?          0:00 [kthreadd]
1     0     3     2  20   0      0     0 smpboo S    ?          0:00 [ksoftirqd/0]
1     0     5     2   0 -20      0     0 worker S<   ?          0:00 [kworker/0:0H]
1     0     7     2 -100   -      0     0 smpboo S    ?          0:00 [migration/0]
-- salida truncada --
[student@serverX ~]$ ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 17:45 ?        00:00:03 /usr/lib/systemd/systemd --switched-ro
root         2     0  0 17:45 ?        00:00:00 [kthreadd]
root         3     2  0 17:45 ?        00:00:00 [ksoftirqd/0]
root         5     2  0 17:45 ?        00:00:00 [kworker/0:0H]
root         7     2  0 17:45 ?        00:00:00 [migration/0]
-- salida truncada --
```

De manera predeterminada, el comando **ps** sin opciones selecciona todos los procesos que tienen la misma *identificación de usuario efectivo* (EUID) que el usuario actual y que están asociados con la misma terminal en la que se invocó **ps**.

- Los procesos entre corchetes (normalmente en la parte superior) son subprocesos del núcleo programados.

- Los procesos zombies aparecen en una lista ps como *finalizado* u *obsoleto*.

- **ps** se muestra una vez. Utilice la opción **top**(1) para una visualización de procesos de actualización repetitiva.
- **ps** puede mostrar los resultados en formato de árbol para ver las relaciones de procesos primarios y secundarios.
- El resultado predeterminado no está ordenado. El orden de visualización coincide con el de la tabla de procesos del sistema, que reutiliza las filas de la tabla, ya que ciertos procesos finalizan y otros nuevos se crean. Los resultados pueden aparecer en orden cronológico, pero esto no es seguro a menos que se usen las opciones explícitas `-O` o `--sort`.

Referencias

info libc signal (*Manual de referencias de la biblioteca GNU C*)

- Sección 24: Manejo de señales

info libc processes (*Manual de referencias de la biblioteca GNU C*)

- Sección 26: Procesos

Páginas de manual **ps**(1) y **signal**(7)

Next