

Unidad 9

Control de procesos

IFC01CM16. Administración avanzada en Linux y
virtualización
Curso 2016

Índice

- ▶ Procesos.
 - Introducción
 - Propiedades
 - Estados y transiciones
 - Acciones
 - Demonios vs Procesos de usuario
 - Creación de procesos
- ▶ Hilos de ejecución (Threads)
 - Características

Índice

- ▶ Control de procesos.
 - Background vs Foreground
 - Listado de procesos
 - Señalización de procesos
 - Modificación de la prioridad de procesos
 - Control de los recursos de un proceso
 - Otros comandos
 - El directorio /proc

Procesos

Introducción

- ▶ Un proceso es cada una de las tareas que se realizan en el sistema operativo.
- ▶ Un proceso se inicia cada vez que se lanza la ejecución de una aplicación o programa.
- ▶ La gestión de procesos puede hacerse mediante:
 - Monitor del sistema:
 - `gnome-system-monitor` (GNOME) y `ksysguard` (KDE)
 - `top` (consola)
 - Comandos: `ps`, `kill`, `jobs`, ...

Procesos

Introducción

- ▶ Concurrencia: capacidad de ejecutar varios procesos simultáneamente. Dos tipos:
 - Paralelismo real: cuando se ejecutan efectivamente al mismo tiempo en una arquitectura mutiprocesador o multi-core
 - Paralelismo virtual o concurrencia: cuando es el SO el que genera mediante compartición de tiempo y recursos una ilusión de ejecución simultánea
- ▶ La información relativa a un proceso se almacena en una estructura protegida llamada PCB (Process Control Block)

Procesos

Propiedades

- ▶ Nombre del proceso e identificador (PID)
- ▶ Credenciales: cada proceso está asociado a un UID y uno o varios GID que determinan sus permisos.
- ▶ Estado (ejecutando, detenido, ...)
- ▶ Prioridad de ejecución
- ▶ Memoria ocupada
- ▶ Uso de CPU
- ▶ ...

Procesos

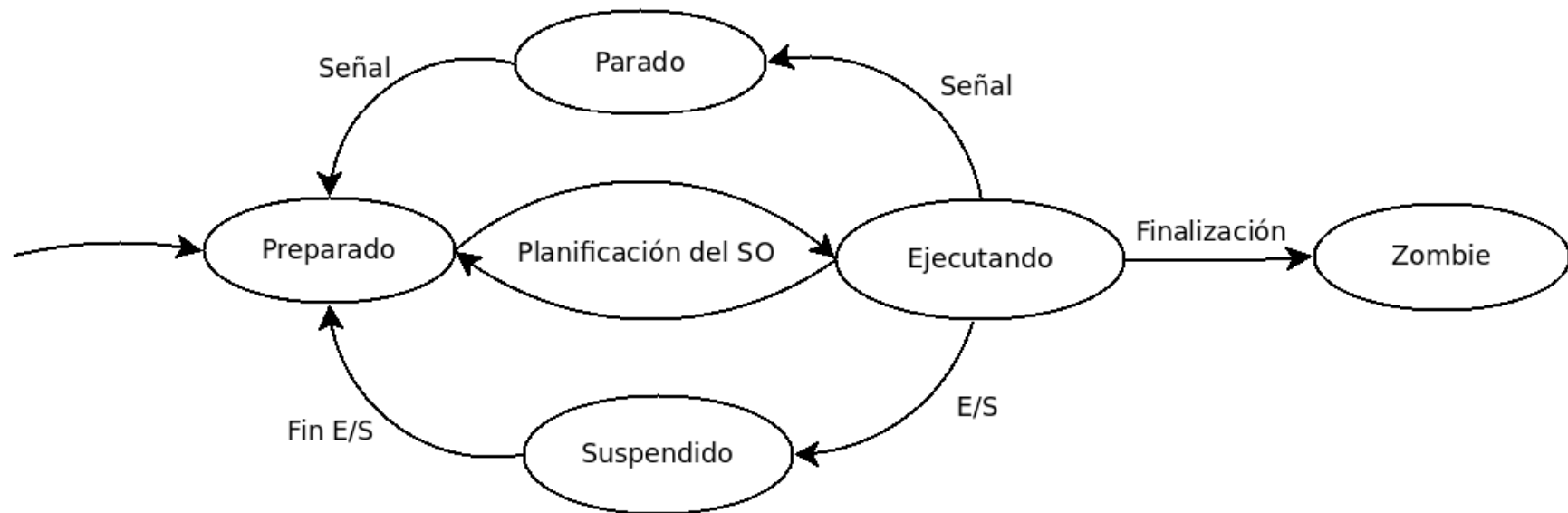
Estados y transiciones

- ▶ Un proceso puede pasar por diferentes estados durante su vida, de forma resumida son:
 - Preparado (ready): listo para ejecución y en espera de CPU.
 - Ejecutando (running (R)): el proceso está en ejecución. Puede ser en modo usuario o en modo kernel (más privilegios)
 - Suspendido (Uninterruptable sleep (D)): esperando un evento externo, generalmente de E/S, para poder continuar.
 - Parado (Interruptable sleep (S)): esperando un evento de otro proceso para poder continuar.
 - Zombie (zombie (Z)): el proceso ha terminado su ejecución pero aún no se ha eliminado de las estructuras de datos del kernel

Procesos

Estados y transiciones

- El paso de un estado a otro (transiciones) queda representado por este grafo:



Procesos

Acciones

- ▶ Algunas de las acciones básicas que un usuario puede realizar sobre los procesos son:
 - Detener proceso: para la ejecución de un proceso sin eliminarlo.
 - Finalizar o matar proceso: elimina completamente este proceso.
 - Ocultar proceso: no muestra el proceso en la lista de procesos.
 - Cambiar prioridad: darle mayor o menor prioridad de ejecución a un proceso.

Procesos

Demonios vs Procesos de usuario

- ▶ Un demonio es un proceso que se ejecuta en segundo plano (background) y que no es controlado directamente por el usuario.
 - Se emplea normalmente para prestar algún servicio del sistema. Normalmente están en ejecución permanente.
 - Hijos del proceso **init**.
 - `sshd`, `udev`, `syslogd`, `cron`...
- ▶ Los procesos de usuario son procesos generalmente interactivos que ejecuta un usuario normal dentro de su entorno.

Procesos

Creación de procesos

- ▶ Los procesos se crean mediante la llamada al sistema `fork` y `exec` que generan un proceso hijo (child) del proceso padre llamante (parent).
 - `fork`: genera una copia del proceso padre.
 - `exec`: lanza en el nuevo proceso la ejecución de un nuevo programa.
- ▶ El proceso hijo es inicialmente una copia del proceso padre y se ejecuta en una copia de la memoria del proceso padre y con una copia del entorno de ejecución. Por tanto si el proceso hijo modifica el entorno de ejecución esto no afectará al proceso padre (y viceversa).

Procesos

Creación de procesos

- ▶ Los procesos huérfanos (orphans) son procesos que pierden a su padre ya sea intencionadamente (como algunos demonios) o accidentalmente.
- ▶ Los procesos huérfanos son “adoptados” inmediatamente por el proceso **init**.

Hilos de ejecución (Threads)

Características

- ▶ Los threads al igual que los procesos permiten hacer varias cosas simultáneamente.
- ▶ Los threads se ejecutan concurrentemente según la planificación del sistema operativo de la misma forma que lo hacen los procesos.
- ▶ Los threads existen dentro de los procesos y son una unidad de ejecución de grano más fino que los procesos.

Hilos de ejecución (Threads)

Características

- ▶ Cuando se crea un proceso, este se ejecuta dentro de un único thread que se ejecuta de forma secuencial.
- ▶ Este thread puede crear nuevos threads que ejecutan el mismo programa dentro del mismo proceso y por tanto comparten espacio de memoria, entorno de ejecución, etc.
 - En esto difieren de la creación de procesos hijos como hemos visto anteriormente.
 - Las modificaciones que haga un thread serán vistas por los otros threads del mismo proceso.
- ▶ Diferentes threads de un mismo proceso pueden estar ejecutando diferentes partes de un mismo programa al mismo tiempo.

Control de procesos

- ▶ Como ya vimos las principales acciones que podemos realizar sobre los procesos son:
 - Listar procesos en ejecución.
 - `ps, pstree`
 - Señalizar procesos: detener y matar procesos
 - `kill`
 - Modificar la prioridad de ejecución de un proceso.
 - `nice, renice`

Control de procesos

Background vs Foreground

- ▶ Por defecto los comandos se ejecutan en primer plano (foreground), es decir, se espera a la finalización del mismo para mostrar de nuevo el shell
- ▶ Ejecutar un comando en background
 - `sleep 20 &`
- ▶ Terminar proceso en foreground: `Ctrl-C`
- ▶ Pausar proceso en foreground: `Ctrl-Z`
- ▶ pasar un proceso a background: `bg`
- ▶ pasar un proceso a foreground: `fg`

Control de procesos

Listado de procesos

► Comando `ps`

- Muestra ingente información estática de los procesos (UID, PID, PPID, TTY, CPU, memoria, ...) → `man ps`
- `ps -ef` // todos los procesos en formato largo
- `ps --user usuario1` // los procesos de `usuario1`
- `ps f` // jerarquía procesos, similar a `pstree`

► Comando `top`

- Muestra información dinámica de los procesos.
 - `h` → muestra interfaz interactivo
 - `q` → salir

Control de procesos

Listado de procesos

▶ Comando `strace`

- Muestra las llamadas al sistema que se hacen al ejecutar un comando
 - `strace ls`

▶ Comando `jobs`

- Muestra los procesos en ejecución en background y su estado (parado, ejecutando, etc)

Control de procesos

Señalización de procesos

- ▶ El comando **kill** se emplea para mandar diferentes señales a los procesos (`kill -l` muestra el conjunto de señales).
- ▶ Tipos de señales (las más importantes, véase `man 7 signal`)
 - SIGHUP (1): cuelgue detectado en el terminal o muerte del proceso controlador
 - SIGTERM (15): mata al proceso dejándole terminar correctamente. Es la señal enviada por defecto (`kill -15 PID ~ kill PID`)
 - SIGKILL (9): mata al proceso sin permitirle terminar.
 - SIGSTOP (19): para el proceso
 - SIGCONT (18): continúa el proceso si estaba parado
 - SIGINT (2): interrupción procedente del teclado (equivale a Ctrl-C)
 - SIGTSTP (20): para el proceso desde el teclado (equivale a Ctrl-Z)
- ▶ Formato:

```
kill -tipo PID ~ kill -número PID  
kill -KILL 123 ~ kill -9 123
```

Control de procesos

Señalización de procesos

► Ejemplos:

- En bash, si enviamos SIGHUP a una shell, esta señal se reenvía todos sus hijos.
- Se emplea esta señal al cerrar una terminal en entorno gráfico o hacer un cierre de sesión
- Los demonios responden a SIGHUP releyendo sus ficheros de configuración (`kill -HUP sshd` por ejemplo)

Control de procesos

Señalización de procesos

► Otros comandos:

- `pgrep`: equivale a `ps | grep` permite buscar procesos por su nombre en lugar de por el PID.

- `pgrep -u alumno bash`

- `pkill`: envía señales a procesos indicando su nombre (admite comodines).

`pkill -KILL nano // mata todos los nano`

- `exec`, `execve`: ejecuta un comando o script en un nuevo shell

Control de procesos

Modificación de la prioridad de procesos

- ▶ Un proceso se ejecuta con un valor de prioridad (**nice**) que lo sitúa antes o después en la lista de tareas planificadas por el SO.
 - Valores: -20 (máxima prioridad) → 19 (mínima prioridad)
 - Prioridad por defecto: 0
- ▶ Un usuario solo puede reducir la prioridad de sus propios procesos.
- ▶ root puede cambiar la prioridad de cualquier proceso.
- ▶ Comandos: `nice`, `renice`

Control de procesos

Modificación de la prioridad de procesos

► nice

- Inicia la ejecución de un nuevo comando con la prioridad corregida respecto a la prioridad por defecto.
- La prioridad por defecto se modifica por ajuste indicando el `anice -n ajuste comando`
- **Aumento o disminución de prioridad**

`nice // muestra la prioridad actual`

`nice -n 5 bash`

`//disminuye 5 la prioridad del nuevo bash`

`nice -n -1 bash`

`//aumenta 1 grado la prioridad de bash`

Control de procesos

Modificación de la prioridad de procesos

- ▶ `renice`
 - Modifica la prioridad de una tarea o conjunto de tareas que están ya en ejecución.
 - `renice pri [-p pid] [-g pgrp] [-u user]`
 - `-p pid` cambia la prioridad para el proceso especificado
 - `-g pgrp` cambia la prioridad para los procesos ejecutados por los usuarios que pertenecen al grupo con `GID=pgrp`
 - `-u user` cambia la prioridad para los procesos del usuario especificado
 - `renice 5 -u alumno`
 - `//` aumenta 5 la prioridad de todos los procesos de alumno

Control de procesos

Control de los recursos de un proceso

- ▶ Es posible controlar los recursos del SO que emplea un proceso con el comando `ulimit`
- ▶ `ulimit [opciones] [limite]`
- ▶ Algunas opciones:
 - `-a` muestra los límites actuales
 - `-f` máximo tamaño de los ficheros creados por el shell (opción por defecto)
 - `-t` máximo tiempo de CPU
 - ...

Control de procesos

Otros comandos

- ▶ `uptime`
 - Entre otros muestra el tiempo de encendido del sistema, usuarios conectados y carga de proceso del sistema en los últimos (1, 5, 15) min.
- ▶ `w`
 - Muestra los usuarios con sesión iniciada en el sistema y qué están haciendo.
 - Opciones:
 - PCPU: tiempo ocupado por el proceso actual (indicado en WHAT)
 - JCPU: tiempo total ocupado por todos los procesos desde ese terminal (TTY)
 - ...

Control de procesos

Otros comandos

- ▶ `free`

- Memoria libre y memoria usada por el sistema.

- ▶ `uname`

- Ofrece información general del sistema

`uname -r` // versión del kernel

Control de procesos

El directorio /proc

- ▶ Directorio virtual que contiene información sobre la ejecución del SO y los procesos.
- ▶ Se inicializa durante el arranque del sistema
- ▶ No se guarda en disco duro
- ▶ Sus estructura depende de la versión de kernel empleada
- ▶ Muchos comandos de esta sección extraen su información de /proc
 - `lsmod ~ cat /proc/modules`
 - `lspci ~ cat /proc/pci`
- ▶ Permite con el comando `sysctl` modificar parámetros del kernel en tiempo de ejecución
 - `sysctl` accede a las variables del sistema contenidas en `/proc/sys/`

Control de procesos

El directorio /proc

- ▶ Entre otros hay estos ficheros:
 - cpuinfo: información de la CPU.
 - meminfo: uso de memoria
 - version: versión del kernel
 - modules: módulos del kernel
 - ioports: puertos del sistema
 - interrupts: interrupciones del sistema
- ▶ Directorios:
 - Cada proceso crea un directorio identificado con su PID que contiene información sobre estado del proceso, entorno, mapa de memoria, ficheros abiertos, etc.
 - `/proc/sys`: variables del kernel

Práctica

► Práctica 9.1

- Gestión de procesos

```
alumno@ServidorUbuntu01:~$ pstree
init--acpid
      |
      |--atd
      |--cron
      |--dbus-daemon
      |--4*[getty]
      |--login--bash--pstree
      |--login--bash--sleep
      |--rsyslogd--3*[rsyslogd]
      |--udev--2*[udev]
      |--upstart-socket-
      |--upstart-udev-br
      |--whoopsie--{whoopsie}
alumno@ServidorUbuntu01:~$
```

Bibliografía

- ▶ Curso de administración de servidores GNU/Linux 2013 - Universidad de Sevilla. Jorge Juan Chico, Enrique Ostúa.
- ▶ Linux Kernel – Processes. Luís Fernando Friedrich (www.inf.ufsc.br/~fernando) – Universidad Federal de Santa Catarina, Brasil)
- ▶ <http://www.advancedlinuxprogramming.com>
- ▶ <http://www.tldp.org>
- ▶ <http://theurbanpenguin.com>
- ▶ <https://help.ubuntu.com>

Bibliografía

- ▶ <https://help.ubuntu.com>
- ▶ <http://www.gnu.org>
- ▶ <http://upstart.ubuntu.com>
- ▶ <http://mitchtech.net/>
- ▶ <http://mogaal.com>
- ▶ <http://www.vicente-navarro.com>
- ▶ <http://www.alexonlinux.com>

Bibliografía

- ▶ <https://www.linux.com>
- ▶ <http://debian.org>
- ▶ <https://fedoraproject.org>
- ▶ <http://0pointer.de/blog/projects/systemd-for-admins-1.html>
- ▶ <http://www.wikipedia.org>
- ▶ TimeSys Corporation Newsletter. Rob Landley 2006
- ▶ Guía rápida de Upstart. Juan Hernando Vieites (<http://laurel.datsi.fi.upm.es> , 2012)