Ficheros y comandos avanzados

Sistemas Operativos

Enrique Soriano

GSYC

9 de marzo de 2020

Universidad
Rey Juan Carlos

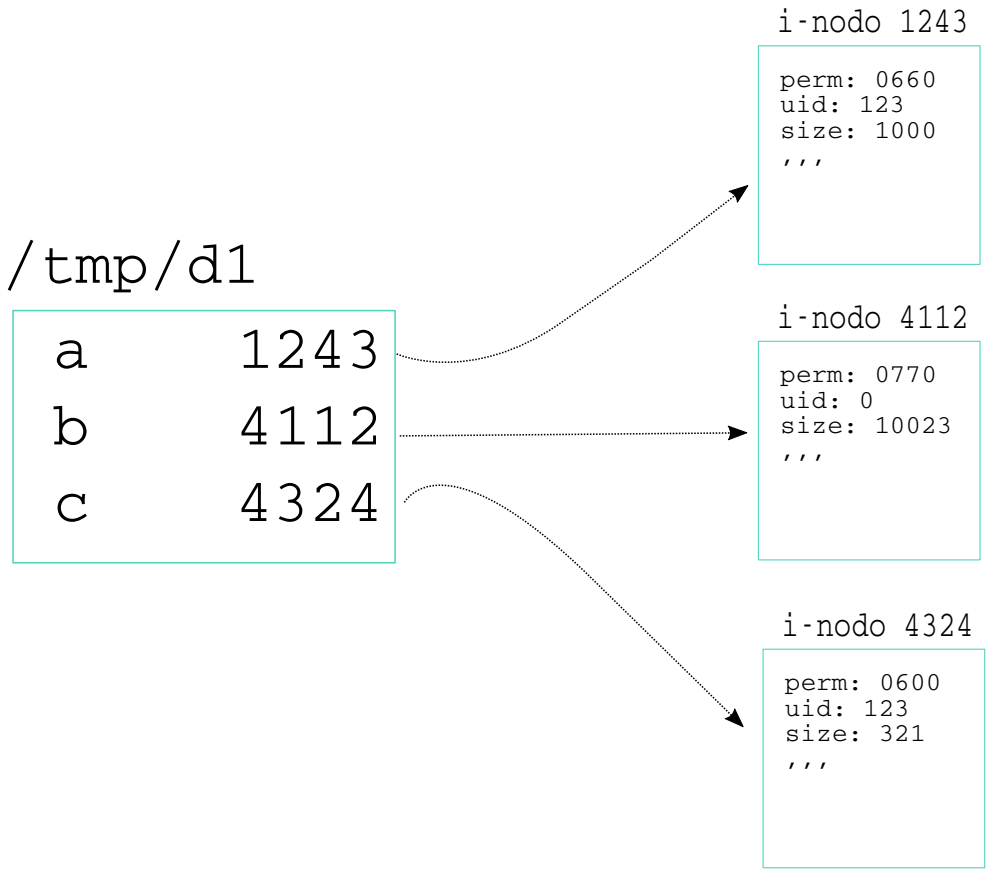
Metadatos

- Son los datos sobre un fichero, no los datos de un fichero.
- Ejemplos que ya conoces: el nombre del fichero y los permisos del fichero.
- Hay más metadatos.

I-nodo

- En realidad, en un sistema de tipo Unix, un directorio es una lista de **entradas**.
- Una entrada de directorio es básicamente el nombre de un fichero y un número que identifica dicho fichero: número de i-nodo.
- Distintas entradas pueden tener distintos nombres para el mismo i-nodo. Esto es, se pueden tener múltiples referencias al mismo fichero (i.e. un fichero puede tener distintos nombres).
- Entre las entradas, tiene la entrada de . (su i-nodo) y .. (i-nodo del padre).
- `ls -i`

I-nodo


/tmp/d1
a 1243
b 4112
c 4324
i-nodo 1243
perm: 0660
uid: 123
size: 1000
,,,
i-nodo 4112
perm: 0770
uid: 0
size: 10023
,,,
i-nodo 4324
perm: 0600
uid: 123
size: 321
,,,

I-nodo

```
 $ mkdir d
 $ cd d
 $ ls -di .
2371879 .
 $ ls -di ..
2359297 ..
 $ echo hola > a
 $ echo adios > b
 $ ls -i
2371884 a  2371885 b
```

I-nodos

- El directorio raíz siempre tiene el número de i-nodo 2.
- El i-nodo es una estructura que está en la partición del sistema de ficheros. El número de i-nodo se usa para localizar la estructura (es el índice en un array).
- La estructura i-nodo es la que contiene los metadatos del fichero y la lista de bloques de disco que contienen los datos del fichero.

I-nodos

- La estructura contiene:
  - permisos
  - tiempos
    - acceso a datos (`atime`)
    - modificación de los datos (`mtime`)
    - modificación del i-nodo (`ctime`)
  - tamaño
  - dueño
  - tipo
  - número de bloques
  - contador de referencias (links)
- **No** contiene el nombre de fichero.

Enlaces duros

- Un enlace duro es otro nombre para el fichero, esto es, otra entrada de directorio para ese i-nodo.
- El contador de referencias del i-nodo se incrementa cada vez que se crea un nombre nuevo para el fichero.
- El contador de referencias se decrementa cada vez que se elimina una entrada de directorio que hace referencia al i-nodo.

Enlaces duros

- Cuando el contador llega a 0, se puede eliminar el fichero (se libera el i-nodo y todos los bloques asociados a este, donde estaban los datos del fichero).
- En Linux no se permite crear enlaces duros para directorios: rompen la jerarquía, crean bucles y crea ambigüedad con .. (el padre).

Enlaces duros

- El comando `ln` permite crear un enlace duro.

```
 $ echo hola > a
 $ ls -i a
2371879 a
 $ ln a b
 $ ls
a  b
 $ ls -i
2371879 a  2371879 b
 $ cat a
hola
 $ cat b
hola
 $ rm a
 $ cat b
hola
```

Tipos de ficheros

- Ya sabes qué es un fichero normal y un directorio.
- Hay otros tipos:
  - Enlaces simbólicos (symlink, 'l').
  - Pipes con nombre (fifo, 'p').
  - Dispositivos (device, 'c' o 'b').
  - Conexiones de red (sockets, 's').

Enlaces simbólicos (blandos)

- NO se trata de otro nombre para el mismo i-nodo.
- Se trata de un fichero especial distinto (*symlink*), cuyos datos contienen la ruta al fichero enlazado.
- Pueden *romperse*: si el fichero enlazado se borra, el enlace está roto.
- También se crean con el comando `ln`, usando el modificador `-s`.

Enlaces simbólicos (blandos)

```
 $ echo hola > a
 $ ln -s a b
 $ ls -li
total 4
2371879 -rw-rw-r-- 1 esoriano esoriano 5 mar 21 13:42 a
2371883 lrwxrwxrwx 1 esoriano esoriano 1 mar 21 13:43 b -> a
 $ cat b
hola
 $ rm a
 $ cat b
cat: b: No such file or directory
 $ ls -li
total 0
2371883 lrwxrwxrwx 1 esoriano esoriano 1 mar 21 13:43 b -> a
```

Pipes con nombre

- Se llaman *fifo*.
- Es un pipe con nombre que persiste en el tiempo.
- Se crean con el comando `mkfifo`.
- Se borran con el comando `rm`.

Pipes con nombre

```
 $ mkfifo p
 $ $ echo uno dos > p  &  sed 's/o/X/g' p
[1] 4459
unX dXs
[1]+  Done                    echo uno dos > p
 $
```

Dispositivos

- En Unix, los dispositivos se presentan como ficheros.
- Ventaja: podemos usar las mismas herramientas que usamos con los ficheros convencionales.
- Ya conoces `/dev/null` etc.
- Los dispositivos están en `/dev/`. Siguen un esquema de nombrado.
- P. ej.: los discos comienzan por `sd` seguido de la posición en el bus (a, b, etc.) seguido del número de partición: /dev/sda2, /dev/sdb1, etc.

Dispositivos

Hay dos tipos de dispositivos:

- Un dispositivo de **caracteres** trabaja con flujos de bytes.
- Un dispositivo de **bloques** trabaja con trozos de datos de cierta longitud (bloque). El dispositivo tiene un tamaño determinado y se tiene acceso aleatorio a sus bloques (i.e. discos duros).
- Tienen asociado dos números que sirven al núcleo para identificar el dispositivo dentro del kernel:
  - Major number: indica la clase del dispositivo (i.e. su driver), todos los de la misma clase tienen el mismo.
  - Minor number: indica la instancia concreta dentro de esa clase de dispositivo.

Dispositivos

```
 $ cd /dev
 $ ls -l sda1 sda2
brw-rw---- 1 root disk 8, 1 mar 21 16:39 sda1
brw-rw---- 1 root disk 8, 2 mar 21 16:39 sda2
 $ ls -l tty1 tty2
crw--w---- 1 root tty 4, 1 mar 21 16:39 tty1
crw--w---- 1 root tty 4, 2 mar 21 16:39 tty2
```

Dispositivos

- Los dispositivos se crean con el comando `mknod`. Hay que proporcionar el tipo (b,c) y los dos números (major y minor).
- No es común tener que crearlos, el sistema los crea automáticamente.
- El demonio `udev` se encarga de esta gestión.

dd

El comando `dd` es muy útil para copiar datos entre dispositivos de bloques. Sus modificadores habituales son:

- `if=file` fichero de entrada.
- `of=file` fichero de salida.
- `bs=size` tamaño de un bloque (p. ej. bs=1k o bs=1024).
- `count=num` número de bloques a transferir.
- `skip=num` número de bloques a saltar de la entrada.
- `seek=num` número de bloques a saltar en la salida.

Inspección del sistema

Desde el shell, tenemos dos formas de inspeccionar el sistema:

- Navegando por sistemas de ficheros sintéticos como `/proc`.
- Ejecutando comandos especializados como `ps`.

/proc

- Es un sistema con ficheros *generados al vuelo*.
- Tiene un directorio por cada proceso en ejecución (p. ej. /proc/323 para el proceso con PID 323).
- Además tiene otros ficheros y directorios importantes.
- `man 5 proc`

/proc

Dentro del directorio para un proceso tenemos (entre otros):

- `cmdline`: línea de comandos que ejecuta
- `cwd`: enlace simbólico a su directorio actual
- `environ`: variables de entorno
- `exe`: enlace simbólico al ejecutable
- `fd`: ficheros que tiene abiertos
- `io`: información sobre entrada/salida
- `maps`: regiones de memoria
- `mem`: memoria del proceso

/proc

/proc tiene otros ficheros y directorios de interés:

- `cpuinfo`: información sobre la CPU
- `kmsg`: log del kernel
- `meminfo`: información sobre el uso de la memoria
- `modules`: modulos cargados
- `net`: directorio con información sobre la red
- `uptime`: tiempo que lleva levantado el sistema
- ...

/sys

- Es otro directorio con ficheros sintéticos como /proc, es la misma idea.
- Es una interfaz para acceder a las estructuras internas del kernel.
- `man 5 sysfs`

/sys

Entre otros, ofrece:

- `block`: enlaces simbólicos para cada dispositivo de bloques
- `class`: directorios por cada clase de dispositivo
- `devices`: representación del árbol de dispositivos
- `firmware`: interfaz para manipular objetos y atributos del firmware
- `fs`: interfaz para controlar los sistemas de ficheros
- `kernel`: manipulación variada del kernel
- `module`: módulos cargados
- `power`: manipulación de la gestión de energía
- ...

ps

- `ps` lista los procesos del sistema.
- Comúnmente se ejecuta con los modificadores `aux`: mostrar todos los procesos.
- Hay muchos otros modificadores para mostrar distintas columnas, hilos de los procesos, etc.

top

- `top` muestra los procesos refrescando sus datos.
- Es útil para ver el consumo de CPU y de memoria de los procesos.
- Las filas se pueden ordenar de distinta forma.
- Pulsando `h` (también ?) nos ofrece algunas de las opciones interactivas. Se sale pulsando `q`.
- Por ejemplo: `P` ordena por uso de CPU, `M` ordena por consumo de memoria, etc.
- Hay alternativas más fáciles de usar, como `htop`.

lsof

- `lsof` muestra los ficheros que tienen abiertos los procesos.
- Como ya sabemos, hay ficheros de múltiples tipos (sockets, pipes, etc.).
- Por omisión nos muestra columnas con el comando, PID, usuario, descriptor de fichero, tipo, dispositivo que lo sirve, tamaño, posición y nombre del fichero.

mount

- `mount` sin argumentos muestra los sistemas de ficheros que están montados.
- Te dice: dispositivo, punto de montaje, tipo de sistema de ficheros y opciones de montaje.
- El fichero `/etc/fstab` tiene algunos de los sistemas de ficheros que se pueden montar.

df

- `df` muestra el espacio libre de los distintos sistemas de ficheros montados.
- La opción `-h` da los tamaños en un formato más legible.

netstat

- `netstat` ofrece información sobre las conexiones de red.
- Tiene muchas opciones, las más habituales:
  - `-a`: lista todas las conexiones
  - `-at`: lista solo conexiones TCP
  - `-au`: lista solo conexiones UDP
  - `-tnl`: lista los puertos de nuestra máquina en los que están escuchando servicios

dmesg

- `dmesg` muestra el *kernel ring buffer*, esto es, los mensajes que escribe el kernel.
- El kernel escribe mensajes de diagnóstico de distinto nivel de importancia (información, avisos, errores de distinta gravedad, etc.).
- El opción `-w` hace que se quede esperando nuevos mensajes.

Usuarios

- `w` muestra la lista de usuarios que están usando el sistema ahora mismo, en qué terminal están, etc. El comando `who` sirve para lo mismo (es menos completo).
- `last` muestra la lista de los últimos usuarios que han entrado en el sistema. Muestra tanto entradas locales como remotas.

Logs

- Los fichero de bitácora (logs) están en `/var/log`.
- Ojo: algunos son binarios, otros son de texto.
- Se van rotando: cuando un fichero llega a un tamaño (o edad) se comprime y se renombra (se le asigna un número). P. ej.: `kern.log.2.gz`.
- `systemd` mantiene un log también, se puede inspeccionar con el comando `journalctl`.

Logs

Algunos logs:

- `syslog`: información general del sistema, todo menos temas de autenticación. Un demonio `syslog` recopila los mensajes de distintos servicios y programas y los escribe en este log. Syslog es configurable (`/etc/syslog.conf`).
- `auth.log`: información sobre autenticaciones (correctas y fallidas)
- `kern.log`: mensajes del kernel
- `wtmp`: contiene la información que muestra el comando `last`
- `dpkg.log`: mensajes sobre la instalación y gestión de paquetes Debian
- `Xorg.N.log`: mensajes de la interfaz gráfica en la sesión N

ssh

- `ssh` permite trabajar con un shell en un sistema remoto de forma segura.
- Es la versión segura de otros programas antiguos (inseguros) como `telnet`, `rsh`, etc.

```
$ ssh esoriano@zeta01.aulas.gsyc.urjc.es
```

scp

El comando `scp` permite copiar un fichero desde el servidor remoto a la máquina local (por ejemplo, del laboratorio a casa):

scp tu-login@servidor:fichero-origen fichero-destino

Ejemplos:

```
scp pepito@alpha.aulas.gsyc.urjc.es:expr.p expr.p
```

(copia el fichero expr.p de tu directorio personal del servidor al directorio actual en tu PC)

```
scp pepito@alpha.aulas.gsyc.urjc.es:Escritorio/a.txt a.txt
```

(copia el fichero a.txt de tu directorio Escritorio en el servidor al directorio actual en tu PC)

scp

Para copiar un fichero local a la máquina remota (por ejemplo al laboratorio desde casa):

scp fichero-origen tu-login@servidor:fichero-destino

Ejemplos:

```
scp expr.p pepito@alpha.aulas.gsyc.urjc.es:expr.p
```

(copia el fichero expr.p del directorio actual de tu PC en tu directorio personal en el servidor)

```
scp func.p pepito@alpha.aulas.gsyc.urjc.es:Documentos/func.p
```

(copia el fichero func.p del directorio actual de tu PC en tu directorio Documentos en el servidor)