Capítulo 4
Los discos y el sistema de archivos

1. Representación de los discos

Cada disco conocido por el sistema está asociado con un archivo especial en modo bloque. Lo mismo ocurre con las partituras.

1.1 Nomenclatura

En función del tipo de controlador e interfaz a los que se conectan los discos, Linux les asocia archivos especiales diferentes. Los nombres de estos archivos especiales dependen del tipo del controlador de disco.

1.1.1 Disco IDE

Este tipo de unidad se ha vuelto poco común, ya que el estándar SATA ha reemplazado al estándar IDE durante muchos años. Los discos conectados a controladores IDE (también llamados PATA (*Parallel ATA*) o ATAPI) están asociados a archivos especiales cuyo nombre es `hdx`, en el que `x` es una letra minúscula, dependiendo del controlador y la posición del disco en relación con su controlador:

- `hda`: IDE0, Principal
- `hdb`: IDE0, Secundario
- `hdc`: IDE1, Principal
- `hdd`: IDE1, Secundario
- etc.

No hay límite en cuanto al número de controladores IDE, más allá del número de los puertos de extensión de la máquina (slots PCI). Existen muchas tarjetas adicionales y convertidores que permiten leer antiguos discos IDE.

Linux entiende que los lectores de CD-Rom, DVD y grabadores de tipo IDE/ATAPI son discos IDE y respetan la nomenclatura citada.

El núcleo de Linux utiliza por defecto un API llamado `libata` para acceder al conjunto de los discos IDE, SCSI, USB, Firewire, etc. La nomenclatura sigue la de los discos SCSI, que trataremos en el apartado siguiente.

1.1.2 Discos SCSI, SATA, USB, FIREWIRE, etc

Los discos conectados a controladores SCSI, SATA, SAS, FibreChannel, USB, Firewire, Thunderbolt, etc., están asociados a archivos especiales denominados `sdx`. La letra minúscula `x` se asigna en el orden de detección de las tarjetas SCSI y los adaptadores asociados (*hosts*), y la conexión o desconexión de discos extraíbles (discos hotplug, memorias USB, etc.). Después de `sdz`, la enumeración continúa con `sdaa`, `sdab`, etc.

– `sda`: primer disco SCSI
– `sdb`: segundo disco SCSI
– `sdc`: tercer disco SCSI
– etc.

La norma SCSI distingue los diversos tipos de soportes. Así, los archivos especiales asociados a los lectores de CD-Rom, DVD, HD-DVD, BlueRay y los grabadores no siguen la misma regla de nomenclatura que los discos. En efecto, aplican la nomenclatura `srn` (`sr0`, `sr1`, etc.).

Observación

Los archivos asociados también se pueden nombrar `scd0`, `scd1`, etc., pero normalmente son enlaces simbólicos a `sr0`, `sr1`, etc.

El comando `lsscsi` permite enumerar los periféricos SCSI.

Ejemplo

```
# lsscsi
[0:0:0:0]    cd/dvd  PIONEER  DVDRW   DVR-K17  1.05  /dev/sr0
[2:0:0:0]    disk    ATA      ST9160821AS      D     /dev/sda
[6:0:0:0]    disk    USB      SanDisk 3.2Gen1 1.00   /dev/sdb
[7:0:0:0]    disk    USB      SanDisk 3.2Gen1 1.00   /dev/sdc
```

El lector de CD/DVD se asocia a `/dev/sr0`.

1.2 Casos particulares

Algunos controladores no aplican la nomenclatura anterior a sus archivos especiales.

Ejemplo

Un controlador Smart Array en un servidor HP, que utilice el controlador `cciss`, coloca sus archivos de periféricos en `/dev/cciss` con los nombres `cXdYpZ`, donde `X` es el slot, `Y` el disco y `Z` la partición.

1.2.1 Virtualización

La representación de discos de sistemas invitados (*guests*) virtualizados depende del tipo de controlador simulado. La mayoría son de tipo IDE o SCSI, con la librería `libata`; normalmente se ven como SCSI. Sin embargo, algunos sistemas como, por ejemplo, KVM o XEN (así como los utilizan los entornos cloud, como AWS), que ofrece paravirtualización, disponen de un controlador específico que presenta los discos con el nombre `vdx` (*virtual disk x*) o `xvdx`:

- `vda`: primer disco virtualizado, o `vxda`.
- `vdb`: segundo disco virtualizado, o `vxbd`.
- etc.

1.2.2 SAN, iSCSI, multipathing

Los discos conectados a través de una SAN (*Storage Area Network*) o mediante iSCSI se ven como discos SCSI y conservan esta nomenclatura. Sin embargo, los sistemas de gestión de rutas múltiples (*multipathing*) pueden generar otros nombres.

Ejemplo

Powerpath nombrará a los disqcos `emcpowerx` (`emcpowera`, `emcpowerb`, etc.) mientras que el sistema de Linux multipath los nombrará por defecto `mpathx` (`mpath0`, `mpath1`, etc.).

2. Operaciones de bajo nivel

Hay una variedad de comandos de Linux que se pueden usar para administrar dispositivos de disco a nivel de hardware, a través de su controlador.

Observación

Algunos de estos comandos o sus opciones permiten cambiar la configuración de un disco, por lo que se deben usar con precaución.

Información del disco

El comando `hdparm` permite efectuar un gran número de operaciones directamente en los discos duros gestionados por la librería `libata`, es decir, los discos de tipo SATA, ATA (IDE) y SAS.

El comando `sdparm` permite el mismo tipo de operaciones para los discos SCSI.

Las opciones `-i` o `-I` proporcionan la información de las características de un disco. La primera proporciona información del kernel, obtenido en el momento del arranque, la segunda consulta directamente el controlador de disco y proporciona información más detallada.

Ejemplo

```
# hdparm -I /dev/sda

/dev/sda:
/dev/sda:

ATA device, with non-removable media
        Model Number:       ST9160821AS
        Serial Number:      5MA3G4KT
        Firmware Revision:  3.BHD
Standards:
        Supported: 7 6 5 4
        Likely used: 8
[...]
Security:
        Master password revision code = 65534
                supported
        not     enabled
        not     locked
                frozen
        not     expired: security count
                supported: enhanced erase
        88min for SECURITY ERASE UNIT. 88min for ENHANCED SECURITY ERASE UNIT.
Checksum: correct
```

3. Elegir un sistema de archivos

El árbol global del sistema de archivos Linux consta de uno o más **sistemas de archivos** independientes.

Un sistema de archivos se utiliza para estructurar un espacio de almacenamiento, en forma de archivos y directorios. Esta organización interna del espacio de almacenamiento se hace accesible a las aplicaciones, en forma de directorios y archivos, mediante la operación de montaje en un directorio del árbol global.

Un sistema de archivos desmontado se ve como un conjunto no estructurado de bytes, al que se accede como a un todo, sin la posibilidad de usar su organización en directorios y archivos.

Hay diferentes tipos de sistemas de archivos, que se pueden combinar entre sí dentro del árbol general del sistema de archivos de Linux.

3.1 Fundamentos

Aunque los principios básicos suelen ser los mismos en los distintos tipos de sistemas de archivos Linux, las implementaciones y la organización lógica de los datos en los espacios de almacenamiento varían mucho.

Los sistemas de archivos Linux se ajustan a los estándares POSIX, pero pueden ofrecer extensiones (`ACL`, `selinux`, etc.).

Un sistema de archivos asocia uno o más nombres a un conjunto de datos y gestiona el acceso a su contenido: crear, modificar, eliminar, mover, abrir, leer, escribir, cerrar. Cada tipo de sistema de archivos está soportado por un controlador específico, que se comunica con el kernel para permitir que las aplicaciones accedan a los elementos contenidos en el espacio de almacenamiento. El driver y el kernel se encargan de las operaciones necesarias: mecanismos de protección de acceso (permisos, propietarios), accesos simultáneos, etc.

3.1.1 Representación

Además de la organización y el almacenamiento de la información, el sistema de archivos debe facilitar al usuario una visión estructurada de los datos, que permite distinguirlos, encontrarlos, tratarlos y trabajar con ellos en forma de archivos dentro de una estructura de directorios con los comandos asociados. Asimismo, cada sistema de archivos debe proporcionar lo necesario para que los programas puedan acceder a él.

3.1.2 Los metadatos

Un archivo se describe mediante propiedades denominadas metadatos. En los sistemas de archivos Linux, esta información se almacena en una estructura llamada inodo. Los datos del archivo se almacenan en diferentes bloques del medio de almacenamiento. Los metadatos difieren en función de los diferentes tipos de sistemas de archivos, pero deben contener al menos lo siguiente:

- los permisos de acceso;
- las últimas fechas de acceso y modificación;
- el propietario y el grupo;
- el tamaño;
- el número de bloques utilizados;
- el tipo de archivo;
- el contador de enlaces físicos;
- un árbol de direcciones de bloques de datos.

3.1.3 Los nombres de los archivos: los enlaces físicos

Los nombres de archivo pueden contener hasta 255 caracteres. El eventual sufijo (los caracteres después del último carácter . del nombre) es opcional y no tiene sentido para el sistema.

Los nombres de archivo de Linux no forman parte de los metadatos del inodo, sino que se mantienen en tablas de directorios. Un nombre se asocia con el número de inodo del archivo y permite que el sistema encuentre el archivo físico asociado a dicho nombre. Por lo tanto, es posible dar varios nombres al mismo archivo. Por lo tanto, cada nombre es un enlace físico al archivo. Cuando un archivo físico ya no tiene ningún enlace físico, el sistema lo elimina, su inodo se libera y se puede reutilizar.

3.1.4 Los sistemas de archivos de log cvf

Algunos tipos de sistemas de archivos tienen un mecanismo de registro para garantizar la integridad de los datos en caso de una interrupción abrupta del sistema. El sistema de archivos mantiene un registro en el que se conservan los cambios que se deben realizar antes de ejecutarlos y los confirma cuando se completa la operación. En caso de interrupción, esto permite que el sistema reproduzca los registros de log y volver a realizar o cancelar las operaciones que estaban en curso. Se garantiza la coherencia de los datos del sistema de archivos, ya sea mediante la ejecución o la vuelta atrás. Esto hace que la reparación sea más fiable y rápida que con un tipo de sistema de archivos sin registro.

3.2 Los tipos de sistema de archivos en Linux

Linux soporta muchos tipos de sistemas de archivos. La elección del tipo de sistema de archivos depende de la naturaleza del medio físico, sus capacidades, así como el tipo de uso (lectura, escritura, archivos pequeños y grandes, múltiples archivos pequeños, etc.) y las funcionalidades preferidas (velocidad de acceso de lectura o escritura, seguridad, atributos administrados, número y tamaño de objetos almacenados).

Describiremos los principales tipos de sistemas de archivos utilizados con Linux.

3.2.1 Sistemas de archivos de tipo ext*

El tipo de sistema de archivos ext ha evolucionado en varias versiones a lo largo del tiempo: `ext2`, `ext3` y `ext4`.

ext2

Los sistemas de archivos de tipo `ext2` fueron creados específicamente para Linux. Han sido durante mucho tiempo el tipo de sistema de archivos predeterminado para la mayoría de las distribuciones. Pueden manejar archivos de hasta 2 TB de tamaño, en volúmenes de hasta 32 TB.

Este tipo de sistema de archivos luego evolucionó a `ext3` y `ext4`, y rara vez se usa hoy en día, excepto por compatibilidad con el sistema existente. Su principal desventaja es que no se registra, lo que puede provocar una alteración de la propia estructura del sistema de archivos si se interrumpe bruscamente durante un cambio, dejándolo parcial o totalmente inutilizable.

ext3

Se trata de una evolución del tipo de sistema de archivos `ext2`, que permite configurar un registro de transacciones que garantiza la coherencia de los metadatos del sistema de archivos en caso de una interrupción abrupta de las escrituras.

El registro es opcional. Un sistema de archivos `ext3` permite la lectura y la escritura `ext2`.

Los límites de tamaño de archivo y volumen son los del tipo `ext2`.

ext4

Este tipo de sistema de archivos está destinado a reemplazar el tipo `ext3`. Puede manejar archivos de 16 TB y volúmenes de 1 EB. Una nueva técnica de gestión del espacio asignado a los archivos, las extensiones, puede reducir significativamente la fragmentación. Las fechas de los archivos se pueden administrar hasta el 2514.

Un sistema de archivos de tipo ext4 es compatible con `ext3` en modo de lectura y escritura.

Este es el tipo de sistema de archivos predeterminado para muchas distribuciones, especialmente Debian.

3.2.2 Sistemas de archivos de tipo XFS

XFS es un sistema de archivos diseñado por SGI (*Silicon Graphics Inc*) para sus sistemas Unix IRIX. Creada en 1994, se beneficia de una larga experiencia. Se utiliza principalmente en entornos de producción, con servidores que gestionan espacios de almacenamiento en disco muy grandes.

SGI licenció este sistema de gestión de archivos con licencia GPL en el año 2000, lo que permitió su portabilidad a Linux. Integrado en el kernel de Linux a partir de la versión 2.4, está disponible con la mayoría de las distribuciones.

Diseñado específicamente para administrar de manera eficiente grandes volúmenes de disco, las características clave incluyen:

- 64 bits: el límite teórico máximo para el tamaño de un archivo o de todo un sistema de archivos es de 8 exabytes (2 ^ 63 - 1). En la práctica, puede estar limitado por parámetros relacionados con la distribución (500 TB para RHEL7, 8 exabytes para RHEL8).
- Registro: las operaciones de escritura en metadatos se describen en el registro antes de ejecutarse, lo que ayuda a garantizar la coherencia del sistema de archivos en caso de que se produzca un error de hardware. Para optimizar el rendimiento, el registro se puede colocar en un volumen físico independiente.
- Entradas/Salidas paralelas: gracias a su organización en **grupos de asignación** (un sistema de archivos se divide en varios subconjuntos independientes), permite realizar múltiples operaciones de entrada/salida simultáneamente en el mismo sistema de archivos, lo que es especialmente adecuado para aplicaciones multithreads.
- Se pueden utilizar diferentes técnicas para asegurar un alto nivel de rendimiento y escalabilidad: uso de árboles indexados (**B-tree**) para metadatos, asignación diferida, preasignación para evitar la fragmentación, desfragmentación en línea, asignación por extensiones, etc.
- Las herramientas para comprobar y remediar los sistemas de archivos son muy rápidas.
- Atributos extendidos: permiten gestionar **listas de control de acceso** (ACL), además de los permisos de acceso clásicos del mundo Unix/Linux, así como atributos específicos de la aplicación.
- Sistema de cuotas muy flexible para el uso del espacio, por grupo, usuario o "proyecto" (cuota asociada a una rama del árbol).

– Gestión de **instantáneas** (*snapshots*): esta técnica de tomar una imagen del estado del sistema de archivos en un momento dado permite realizar copias de seguridad en línea.
– Expandible en línea: Un sistema de archivos se puede ampliar mientras está en uso (pero no minimizar).
– Herramientas de copia de seguridad/restauración específicas del sistema de archivos.

Este sistema de gestión de archivos es especialmente adecuado para servidores grandes, requiere mucha RAM e implica el uso de hardware fiable y seguro (caché de respaldo por una batería).

XFS es el sistema de archivos predeterminado para las distribuciones de Red Hat, a partir de la versión 7.

3.2.3 Sistemas de archivos de tipo BTRFS

BTRFS (*B-tree File System*, pronunciado "*butterfs*") es el sistema de archivos creado más reciente para Linux. Implementa conceptos modernos para ofrecer un conjunto de características de alto nivel, resultado de la colaboración de los principales actores del mundo Linux (Red Hat, Oracle, SUSE, Intel, etc.).

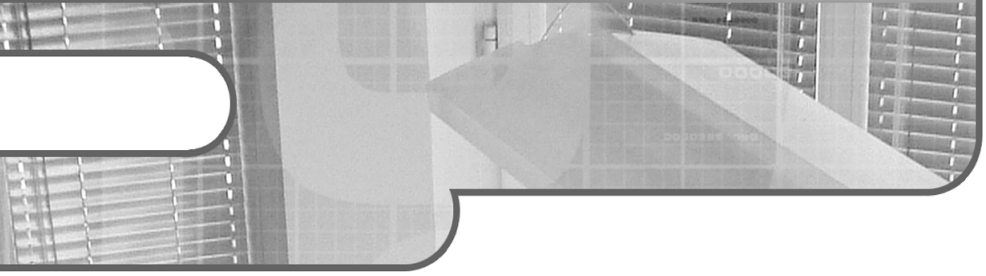
Es un sistema de gestión de archivos de 64 bits, que utiliza una **copy on write** (**CoW**: para garantizar la integridad de los datos, se realiza un cambio copiando los datos originales y luego modificando la copia), implementando capacidades de compresión, copia de seguridad en línea, redundancia de datos, snapshot, redimensionamiento en línea, etc.

Observación

Integrado en el kernel en 2008, se esperaba que eventualmente se convirtiera en el sistema de archivos predeterminado para Linux, pero no es compatible con todas las distribuciones. Red Hat ha decidido dejar de soportarlo a partir de la versión 8 de su distribución RHEL, a diferencia de SUSE, que lo utiliza por defecto para el sistema de archivos raíz. Algunas de sus funciones avanzadas aún no se pueden utilizar en producción.

Sus principales ventajas son las siguientes:

– Sistema de archivos de 64 bits: tamaño máximo de un sistema de archivos de 16 exabytes, tamaño máximo de un archivo de 8 exabytes.
– Optimización de la gestión de pequeños archivos.
– Consideración de las peculiaridades de los discos SSD.
– Mayor confiabilidad mediante la checksum de los datos y metadatos.
– Gestión de las instantáneas (snapshots) de modos lectura y lectura-escritura.



Capítulo 7
Planificación de la capacidad

1. Planificación de la capacidad

Este capítulo trata el monitoreo del uso de recursos del sistema y de la gestión provisional de la evolución de los recursos del sistema.

1.1 Medida y uso de los recursos y depuración

El objetivo de esta sección es de enseñarle a:

– medir el uso de los recursos materiales y del ancho de banda, identificar y resolver los problemas relativos a los recursos.

1.1.1 Competencias principales

– Medir el uso del procesador.
– Medir el consumo de la memoria.
– Medir las entradas/salidas de los discos.
– Medir las entradas/salidas de la red.
– Medir la capacidad de tratamiento del firewall y del enrutamiento.
– Evaluar el consumo de ancho de banda de los clientes.
– Asociar los síntomas y los problemas probables.
– Estimar la capacidad de tratamiento e identificar los cuellos de botella del sistema y de las redes.

1.1.2 Elementos empleados

- `iostat`
- `iotop`
- `vmstat`
- `netstat`
- `ss`
- `iptraf`
- `pstree`, `ps`
- `w`
- `lsof`
- `top`
- `htop`
- `uptime`
- `sar`
- `swap`
- Procesos bloqueados en entrada/salida.
- Bloques en entrada.
- Bloques en salida.

1.2 Planificación prospectiva de los recursos

El objetivo de esta sección es de enseñarle a:

- seguir la evolución del uso de los recursos para anticipar las necesidades en el futuro.

1.2.1 Competencias principales

- Saber utilizar las herramientas de supervisión y de toma de medidas para controlar el uso de la infraestructura informática.
- Determinar el umbral de saturación de una configuración.
- Seguir la progresión del consumo de los recursos.
- Mostrar gráficamente las tendencias de consumo de los recursos.
- Conocimiento básico de las soluciones de supervisión: Icinga2, Nagios, collectd, MRTG y Cacti.

1.2.2 Elementos empleados

- Diagnosticar los problemas de recursos.
- Anticipar el aumento del consumo.
- Prevenir el agotamiento de los recursos.

2. Medida de uso de los recursos y depuración

El administrador del sistema tiene que ser capaz de hacer un censo de los recursos que se encuentran a disposición del sistema y vigilar su uso por el sistema y por las aplicaciones.

2.1 Tipos de recursos

Existen cuatro tipos principales de recursos:

- El o los procesadores.
- La memoria viva.
- El espacio de almacenamiento.
- La red.

Cada una de estas categorías de recursos sirve para permitir que las aplicaciones se ejecuten correctamente, presentando un rendimiento y funcionalidades que puedan satisfacer a los usuarios. Si uno de estos aspectos no fuera suficiente, podría hacer que el funcionamiento del conjunto no fuera correcto, provocando un "cuello de botella".

2.2 Fuentes de información sobre los recursos

El administrador del sistema debe poder cuantificar cada tipo de recurso y vigilar en tiempo real su uso. Para ello, Linux ofrecen diferentes fuentes de información, como las interfaces de comunicación con el núcleo, los comandos o los archivos de registro.

2.2.1 Los pseudo-sistemas de archivos proc y sysfs

El núcleo gestiona los recursos de tipo material y los pone a disposición de las aplicaciones. Para ello, se ocupa de los recursos principales (memoria y procesador) y coordina los pilotos de dispositivos encargados de los otros recursos de tipo material. Por lo tanto, sigue en tiempo real los recursos disponibles y su uso.

Linux dispone de un mecanismo muy potente que permite una comunicación dinámica con el núcleo: los pseudo-sistemas de archivos proc y sysfs (también llamados procfs y sys). Se trata de una interfaz, bajo la forma de un arborescencia de directorios

y de archivos especiales, gestionada por el núcleo. Esto permite obtener información sobre el estado del sistema incluyendo los procesos activos, e incluso enviar información al núcleo con el objetivo de modificar dinámicamente algunos de sus parámetros.

Estos pseudo-sistemas de archivos son una fuente esencial de información sobre los recursos de la máquina así como sobre su uso.

Los elementos que ofrecen información general se encuentran directamente en el directorio de montaje de `proc`, o en su subdirectorio `sys`.

El pseudo-sistema de archivos sysfs permite la gestión unificada de los buses, controladores y dispositivos a partir de una arborescencia virtual. Sobre todo es usado por los controladores de los dispositivos y por aplicaciones especializadas.

En general, el pseudo-sistema de archivos proc está montado en `/proc`, el pseudo-sistema de archivos sysfs está montado en `/sys`, lo podemos comprobar gracias al comando `mount`.

Ejemplo

```
mount
[...]
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
[...]
```

Dos pseudo-sistemas de archivos están montados, en lectura y escritura: proc y sysfs.

El directorio de montaje del pseudo-sistema de archivos proc, `/proc`, contiene numerosos archivos y directorios.

Ejemplo

```
ls /proc
1     15    1904  23    asound       modules
10    1530  1906  24    buddyinfo    mounts
1004  1562  1910  25    bus          mpt
1036  1571  1914  26    cgroups      mtd
1038  1572  1917  27    cmdline      mtrr
1039  16    1928  270   cpuinfo      net
1040  162   1931  271   crypto       pagetypeinfo
1041  1620  1947  28    devices      partitions
1042  1621  2     29    diskstats    sched_debug
1043  1624  20    294   dma          schedstat
1046  1625  2017  295   driver       scsi
1048  163   2018  3     execdomains  self
1049  164   2020  30    fb           slabinfo
1053  1649  2023  303   filesystems  softirqs
1057  1667  2027  31    fs           stat
1059  17    2045  32    interrupts   swaps
```

```
11    1738  2047  37    iomem         sys
12    177   2049  386   ioports       sysrq-trigger
13    178   2056  39    ipmi          sysvipc
1334  179   2057  4     irq           timer_list
1353  18    2058  40    kallsyms      timer_stats
1360  1818  2059  5     kcore         tty
14    1824  2061  6     keys          uptime
141   1825  2062  7     key-users     version
1410  1842  2071  71    kmsg          vmallocinfo
142   1852  2075  8     kpagecount    vmstat
1425  1853  2079  850   kpageflags    zoneinfo
1436  1861  21    851   loadavg
1440  1872  2118  896   locks
1455  1884  2122  897   mdstat
1467  1899  2188  9     meminfo
1483  19    22    acpi  misc
```

Los directorios cuyo nombre es un número están asociados a los procesos activos del sistema, este número corresponde al PID del proceso.

Para acceder a una información, basta con leer el archivo correspondiente. No es un archivo de disco, sino un archivo especial gestionado por el núcleo. Una solicitud de lectura de este archivo envía un mensaje al núcleo y este responde bajo la forma de un conjunto de caracteres.

Ejemplo

Para obtener la versión del núcleo cargado en memoria, basta con leer el contenido del archivo `/proc/version`.

```
cat /proc/version
Linux version 4.18.0-348.23.1.el8_5.x86_64 (mock-build@x86-vm-
07.build.eng.bos.redhat.com) (gcc version 8.5.0 20210514 (Red Hat 8.5.0-4)
(GCC)) #1 SMP Tue Apr 12 11:20:32 EDT 2022
```

El directorio `/proc/sys` contiene especialmente, bajo la forma de pseudo-archivos, información sobre la configuración del núcleo, organizada en directorios por categoría de recursos. Algunos de estos pseudo-archivos pueden ser editados, para poder modificar dinámicamente la configuración del núcleo.

Ejemplos

```
ls /proc/sys
abi   crypto  debug  dev  fs  kernel  net  sunrpc  user  vm
```

Para saber si el enrutamiento IPv4 está activo:

```
cat /proc/sys/net/ipv4/ip_forward
0
```

El núcleo devuelve cero como respuesta a una solicitud de lectura: el enrutamiento no está autorizado. Para activarlo, basta con escribir `1` en el archivo:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
cat /proc/sys/net/ipv4/ip_forward
1
```

Observación

Esta modificación de la configuración es dinámica, pero como se hace directamente en la memoria viva, se perderá en el siguiente reinicio del sistema.

Los directorios dinámicos de procesos

Cada proceso está representado por un directorio en el pseudo-sistema de archivos proc, cuyo nombre es el PID del proceso. Dentro de este directorio se encuentran archivos especiales, gestionados en tiempo real por el núcleo y que dan información sobre las procesos.

En cuanto el proceso se termina, el sistema suprime este directorio.

Ejemplo

En su directorio de conexión, el usuario `becario` abre un archivo `archivo` con Vim. El proceso que ha sido creado tiene como PID 1989.

```
ls -lcd /proc/1989
dr-xr-xr-x. 9 stage stage 0  8 septiembre 13:42 /proc/1989
```

Vemos que el proceso ha sido creado el `08/09` a las `13:42` (fecha de creación del directorio).

```
cd /proc/1989
ls
attr                 fdinfo      oom_adj         stack
autogroup            gid_map     oom_score       stat
auxv                 io          oom_score_adj   statm
cgroup               limits      pagemap         status
clear_refs           loginuid    patch_state     syscall
cmdline              map_files   personality     task
comm                 maps        projid_map      timens_offsets
coredump_filter      mem         root            timers
cpu_resctrl_groups   mountinfo   sched           timerslack_ns
cpuset               mounts      schedstat       uid_map
cwd                  mountstats  sessionid       wchan
environ              net         setgroups
exe                  ns          smaps
fd                   numa_maps   smaps_rollup
```

Todos los archivos y directorios proporcionan información en tiempo real sobre el proceso.

```
cat cmdline
vim archivo
```

El comando ejecutado es: `vim archivo`.

```
cat environ
[...]
PWD=/home/stage
[...]
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpassHOME=/home/
stageSSH_CLIENT=192.168.1.141 65470
22SELINUX_LEVEL_REQUESTED=XDG_DATA_
DIRS=/home/stage/.local/share/flatpak/exports/share:/var/lib/flatpak/
exports/share:/usr/local/share:/usr/shareSSH_TTY=/dev/pts/0MAIL=/var/
spool/mail/stageTERM=xtermSHELL=/bin/bash
```

Este archivo especial da acceso (en solo lectura) a todas las variables de entorno del proceso. Vemos, por ejemplo, el contenido de la variable `PWD: /home/becario`.

```
ls fd
0  1  2  3  4
cat fd/4
3210#"! Utad hello a todos
```

El directorio `fd` contiene la información sobre los archivos abiertos del proceso: las tres entradas/salidas estándares (`0`, `1` y `2`), así como un archivo abierto por vim. El comando `cat` muestra el "contenido" de este archivo abierto, el archivo temporal de vim.

```
cat status
Name:   vim
Umask:  0002
State:  S (sleeping)
Tgid:   1989
Ngid:   0
Pid:    1989
PPid:   1910
TracerPid:      0
Uid:    1001    1001    1001    1001
Gid:    1001    1001    1001    1001
FDSize: 256
Groups: 1001
[...)
```

El archivo `status` muestra la información sobre el estado del proceso.

Cuando el usuario cierra vim, el pseudo-directorio se suprime automáticamente:

```
pwd
/proc/1989
ls
ls: no se puede acceder a .: No existe el fichero o el directorio
```

2.2.2 Los registros del sistema

El sistema y las aplicaciones usan los servicios de un daemon de gestión de registros `rsyslogd` (o de otro equivalente). Se puede encontrar mucha información acerca del uso de los recursos y de los problemas que pueden ser ocasionados consultando los archivos de registro generados por este daemon.

Durante el inicio del sistema, el núcleo debe determinar los recursos materiales de los que dispone. Para ello, activa un componente de software encargado de esta detección, `udev`. Después mostrará en consola los mensajes relativos a estas detecciones y los almacena también en memoria viva, porque el daemon de gestión de registros todavía no está activo y los sistemas de archivos no están posiblemente accesibles en escritura todavía.

Esta información de inicialización podrá ser leída posteriormente, usando el comando `dmesg`. Sin embargo, como el buffer en el cual están almacenados los mensajes es circular, es posible que los mensajes iniciales hayan desaparecido, si el sistema funciona desde hace mucho tiempo o si un controlador de un periférico señala errores de manera recurrente.

Observación

La mayoría de estos mensajes son recuperados de hecho por `rsyslogd` a partir del momento en el que es iniciado y son escritos en el archivo de registro principal (generalmente `/var/log/messages`).

Ejemplo

```
dmesg | more
[    0.000000] microcode: microcode updated early to revision 0x2f,
date = 2019-02-17
[   0.000000] Linux version 4.18.0-348.23.1.el8_5.x86_64 (mock-build@x86-vm-07.build.
eng.bos.redhat.com) (gcc version 8.5.0 20210514 (Red Hat 8.5.0-4) (GCC)) #1 SMP Tue A
pr 12 11:20:32 EDT 2022
[   0.000000] Command line: BOOT_IMAGE=(hd0,msdos1)/vmlinuz-4.18.0-348.23.1.el8_5.x8
6_64 root=/dev/mapper/cl-root ro crashkernel=auto resume=/dev/mapper/cl-swap rd.lvm.l
v=cl/root rd.lvm.lv=cl/swap rhgb quiet
[   0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers
'
[   0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[   0.000000] x86/fpu: Enabled xstate features 0x3, context size is 576 bytes, using
 'standard' format.
[   0.000000] BIOS-provided physical RAM map:
[...]
[   0.000000] Memory: 2589412K/4127088K available (12293K kernel code, 2261K
rwdata,7880K rodata, 2492K init, 13944K bss, 434124K reserved, 0K cma-reserved)
[...]
```