

CAPÍTULO 5

Sistemas de Ficheros

Crear un sistema de ficheros

Un sistema de ficheros es una forma de organizar los datos en un dispositivo físico; este dispositivo físico puede ser cualquier dispositivo de almacenamiento: un disquete, un disco duro, un CD-ROM, un llavero USB, ... En un mismo dispositivo pueden coexistir distintos sistemas de ficheros, aunque por lo general será uno sólo (a excepción de los discos duros).

Lamentablemente la manipulación de los sistemas de ficheros necesita realizarse desde una consola; existe una herramienta gráfica llamada “qtparted”, pero se limita a manipular particiones en discos duros (lo que por otro lado viene muy bien en algunos casos, por ejemplo a la hora de instalar).

En Linux, la mayoría de los dispositivos físicos (discos, impresoras, módems, ...) están representados por ficheros especiales de tipo “device” (o dispositivo); estos se encuentran dentro del directorio /dev del sistema y permiten acceder directamente al hardware que representan, haciendo operaciones de lectura, escritura y control sobre dichos dispositivos; pero para que nosotros podamos utilizar estos elementos necesitamos crear un sistema de ficheros, que no es otra cosa que crear en el dispositivo unas estructuras especiales que nos permitan leer y escribimos la información en ese dispositivo.

ADVERTENCIA: Cuando se crea un sistema de ficheros sobre un dispositivo, todo el contenido de dicho dispositivo se elimina y no se puede recuperar de ninguna forma (aunque existen algunas excepciones a esto). Ten esto en cuenta siempre presente porque no querrás comprobarlo por ti mismo.

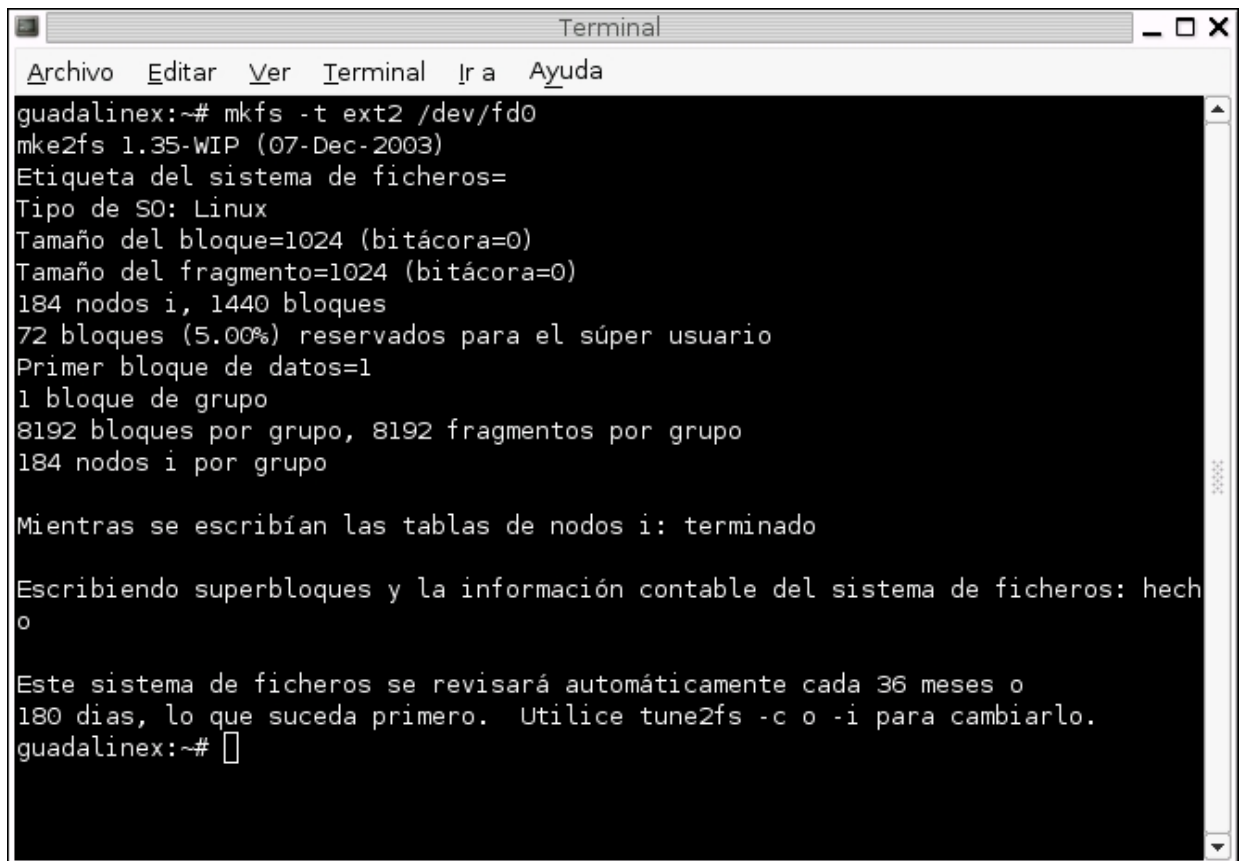
Existe una gran variedad de sistemas de ficheros, aunque en la práctica se usan sólo unos pocos; normalmente cada sistema operativo define uno o varios de ellos, que son los que utiliza para instalar dicho sistema o guardar sus datos. Los más utilizados en la actualidad son:

- ext2: es un sistema de ficheros creado expresamente para ser utilizado por Linux; ofrece el mejor rendimiento (en términos de velocidad y utilización de CPU) de todos los sistemas bajo Linux, permite protección de archivos por usuarios y grupos, no es necesario defragmentarlo, ...

- ext3: es una versión de ext2 al que se le ha añadido un registro de transacciones (denominado “bitácora” o “journal”); esta bitácora es un registro donde se almacenan las operaciones que se realizan en el sistema de ficheros, de manera que ante una interrupción inesperada (por ejemplo un apagón) se evite en la medida de lo posible la pérdida de datos.
- reiserfs: otro sistema con bitácora, pero diseñado partiendo desde cero. Para manejarlas se necesita tener instalado el paquete “reiserfsprogs”.
- xfs: sistema de ficheros orientado a proporcionar alto rendimiento y escalabilidad (pensado para servidores). Para manejarlas es necesario instalar el paquete “xfsprogs”.
- swap: utilizado como memoria de intercambio o paginación en Linux.
- msdos: es el sistema usado en DOS y algunas versiones antiguas de Windows; los nombres de archivos están limitados a 8 caracteres, pudiendo estar seguidos de un punto y tres caracteres más. Está en desuso.
- vfat: es una extensión de msdos para soportar nombres largos de archivos que se utiliza en Windows 95 o NT y versiones superiores.
- iso9660: es un estándar utilizado primordialmente en los CD-ROM.
- nfs: sistema de ficheros de red utilizado para acceder discos localizados en máquina remotas.
- smb: es un sistema que soporta el protocolo SMB, utilizado en ordenadores Windows para compartir elementos por la red.

Para crear un sistema de ficheros hay que utilizar el comando “mkfs”; lo único que hay que hacer es saber cómo nombra Linux a ese dispositivo y decidir qué tipo de sistema de ficheros vamos a crear en él.

Coge un disquete que vamos a crear en él un sistema de ficheros; en Linux se suelen representar por /dev/fd0 (o /dev/fd1), así que para formatear un disquete con ext2 simplemente hay que escribir: “mkfs -t ext2 /dev/fd0”, como se muestra en la imagen (la opción “-t” es para especificar el tipo de sistema de ficheros que queremos). Recuerda que esto borrará todo el contenido del disquete.



```
Terminal
Archivo  Editar  Ver  Terminal  Ir a  Ayuda
guadalinux:~# mkfs -t ext2 /dev/fd0
mke2fs 1.35-WIP (07-Dec-2003)
Etiqueta del sistema de ficheros=
Tipo de SO: Linux
Tamaño del bloque=1024 (bitácora=0)
Tamaño del fragmento=1024 (bitácora=0)
184 nodos i, 1440 bloques
72 bloques (5.00%) reservados para el súper usuario
Primer bloque de datos=1
1 bloque de grupo
8192 bloques por grupo, 8192 fragmentos por grupo
184 nodos i por grupo

Mientras se escribían las tablas de nodos i: terminado

Escribiendo superbloques y la información contable del sistema de ficheros: hecho

Este sistema de ficheros se revisará automáticamente cada 36 meses o
180 días, lo que suceda primero.  Utilice tune2fs -c o -i para cambiarlo.
guadalinux:~#
```

Se pueden crear otros tipos de sistemas en un disquete, como msdos o vfat (ext3 y reiserfs están pensados para dispositivos de mayor capacidad).

Una lista de los dispositivos más comunes son:

disquetes	/dev/fd0			
discos IDE	/dev/hda	/dev/hdb	/dev/hdc	/dev/hdd
discos SCSI	/dev/sda	/dev/sdb	/dev/sdc	
cd-rom	/dev/cdrom			
llaveros	USB	como si fueran SCSI		

El orden de los discos IDE tal y como están nombrados se corresponden respectivamente con: máster primario, esclavo primario, máster secundario y esclavo secundario. Según esto el cd-rom puede ser accedido también mediante el dispositivo correspondiente, aunque por lo general se utiliza /dev/cdrom.

Para crear sistemas de ficheros en discos duros (tanto IDE como SCSI) hay que especificar además la partición en la que queremos crearla; esto se hace añadiendo el número de partición al nombre del disco. Por ejemplo, la primera partición del primer disco es /dev/hda1, mientras que la cuarta del tercero es /dev/hdc4. De esta forma, para crear un sistema de ficheros tipo ext3 en la segunda partición del segundo disco escribiríamos: mkfs -t ext3 /dev/hdb2. Ten mucho cuidado si estás como root de no equivocarte para no perder archivos.

Seguramente te estarás preguntando cómo de importante es la elección del sistema de ficheros para el dispositivo; pues bien, además de la velocidad va a determinar no sólo la forma de almacenar la información, sino también la forma de gestionar cómo se hace y la disponibilidad de los datos almacenados. Esto quiere decir que si aplicas el formato ext2 a un disquete, mantendrás los atributos de propiedad y permisos de los archivos, pero no podrás leerlo desde Windows, mientras que si lo formateas con msdos o vfat podrás leerlo pero perderás esa información asociada al archivo.

Lo mismo ocurrirá con las particiones del disco duro: Windows solo podrá usar particiones msdos o vfat, mientras que Linux podrá verlas todas, aunque sólo podrá utilizar los permisos de usuarios y grupos en particiones ext2, ext3, reiserfs o xfs.

El tipo iso9660 no plantea problemas; puede leerse tanto en Windows como en Linux. El resto de sistemas los veremos más adelante en el presente manual.

La partición de sistema

El sistema de ficheros de Linux (independientemente del sistema utilizado) se compone de un único árbol de directorios que comienza en el directorio principal (designado por "/"); este árbol puede estar integrado por varios dispositivos físicos (e incluso dispositivos virtuales). A cada dispositivo integrado en el árbol de directorios se accede mediante un subdirectorio común del mismo llamado punto de montaje del dispositivo. De esta forma resulta transparente para los usuarios y las aplicaciones qué dispositivos forman el árbol de directorios y dónde están montados.

```
Terminal
Archivo  Editar  Ver  Terminal  Ir a  Ayuda
guadalinux:~# ls -l
total 262504
drwxr-xr-x  2 root    root      4096 2004-05-12 16:19 auto
drwxr-xr-x  2 root    root      4096 2004-05-12 16:23 bin
drwxr-xr-x  3 root    root      4096 2004-05-22 17:39 boot
drwxr-xr-x  2 root    root      4096 2004-05-12 16:19 cdrom
drwxr-xr-x  2 root    root      4096 2004-05-12 16:19 cdrom1
drwxr-xr-x  8 root    root     24576 2004-05-22 15:26 dev
drwxr-xr-x 95 root    root     8192 2004-05-22 18:59 etc
drwxr-xr-x  2 root    root      4096 2004-05-12 16:19 floppy
drwxrwsr-x  3 root    staff     4096 2004-05-21 19:49 home
drwxr-xr-x  2 root    root      4096 2004-05-12 16:19 initrd
drwxr-xr-x  6 root    root      4096 2004-05-12 16:19 lib
drwxr-xr-x  2 root    root      4096 2004-05-12 16:19 lost+found
drwxr-xr-x 10 root    root      4096 2004-05-16 02:13 mnt
drwxr-xr-x  2 root    root      4096 2004-05-12 16:19 opt
dr-xr-xr-x 98 root    root         0 2004-05-22 17:25 proc
drwxr-xr-x 10 root    root      4096 2004-05-22 18:59 root
drwxr-xr-x  2 root    root      4096 2004-05-22 13:20/sbin
-rw-r--r--  1 root    root    268435456 2004-05-12 16:00 swapfile
drwxrwxrwt 12 root    root      4096 2004-05-22 19:04 tmp
drwxr-xr-x 13 root    root      4096 2004-05-12 16:19 usr
drwxr-xr-x 16 root    root      4096 2004-05-12 16:19 var
lrwxrwxrwx  1 root    root         19 2004-05-12 16:04 vmlinuz -> boot/vmlinuz-2.4.23
guadalinux:~#
```

En la imagen se muestra el directorio raíz típico; este sistema cumple con una organización llamada FHS (Filesystem Hierarchy Standard, estándar jerárquico para el sistema de ficheros); los diferentes subdirectorios se suelen utilizar para tareas específicas:

/bin: aquí están los ejecutables necesarios para hacer funcionar el sistema y que tienen utilidad para todos los usuarios.

/boot: contiene los ficheros y datos necesarios para arrancar el sistema.

/cdrom y */floppy*: puntos de montaje del CD-ROM y la disquetera.

/dev: ficheros especiales que representan dispositivos hardware.

/etc: contiene los ficheros de configuración de diferentes programas.

/home: directorios de inicio de los usuarios.

/lib: librerías dinámicas esenciales del sistema.

/lost+found: aquí es donde aparecen los archivos o trozos recuperados tras ocurrir algún accidente; es una característica del sistema ext2.

/mnt: punto de montaje transitorio de sistemas de ficheros.

/proc: sistema de ficheros virtual generado por el núcleo y que permite

obtener y enviar información al hardware.

/root: directorio de inicio del superusuario.

/sbin: ejecutables de interés únicamente para el administrador.

/tmp: datos temporales empleados por diversos programas.

Hay dos directorios importante que nos queda por ver: */usr* y */var*; ambos tienen en común que no son esenciales para el funcionamiento del sistema, pero se diferencian en que mientras */usr* permanece invariable (salvo cuando se instalan programas), */var* cambia durante la operación normal del sistema. Por eso suelen residir en particiones independientes (y */usr* en modo sólo lectura). En ellos encontramos los siguientes subdirectorios:

/usr/X11R6: estructura similar a */usr* pero utilizado por los programas y componentes del entorno gráfico X-Window.

/usr/bin, */usr/sbin* y */usr/lib*: hacen la misma función que */bin*, */sbin* y */lib*, pero para programas que no son indispensables para el arranque y funcionamiento del sistema. Los programas que aquí se encuentran son comunes a todas las distribuciones del mismo grupo (todas las basadas en Debian, por ejemplo) mientras que los de */bin*, */sbin* y */lib* son comunes a todos los Linux.

/usr/doc: información adicional suministrada por algunos programas.

/usr/local: repite la estructura de */usr*, pero para programas y componentes particulares de nuestro sistema (que no estarán en otra Debian).

/var/log: ficheros que registran información generada por diversos procesos del sistema, como mensajes de error.

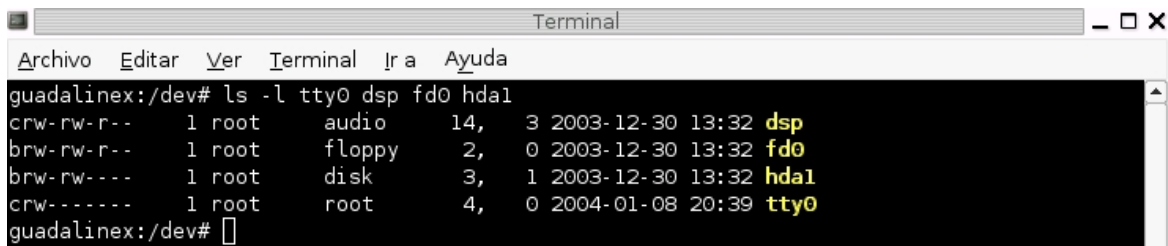
/var/spool: trabajos pendientes de realización, como por ejemplo las colas de impresión (*/var/spool/cups*) y los buzones correo (*/var/spool/exim*).

Existen más directorios, pero los principales los hemos expuesto aquí.

Ficheros especiales

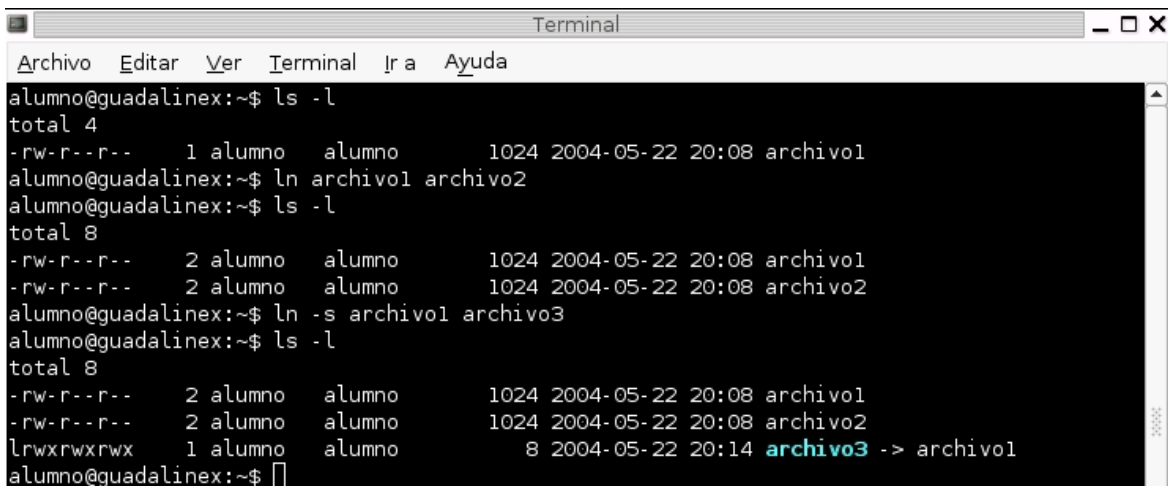
En Linux, aparte de los directorios y ficheros convencionales, existen una serie de ficheros “especiales”. Un grupo lo constituyen los ficheros de dispositivos que nombramos anteriormente, que representan ciertos dispositivos físicos. Los hay de dos tipos: de tipo carácter (marcados con una c) y de tipo bloque (marcados con una b). Con los de tipo carácter la transferencia se realiza byte a byte, mientras que con los de tipo bloque la transferencia se realiza por bloques completos de un determinado tamaño.

En la imagen puedes ver que los dispositivos “dsp” y “tty0” son de tipo carácter (se corresponden con el procesador digital de la tarjeta de sonido y la primera consola virtual, respectivamente), mientras que “fd0” y “hda1” (la disquetera y la primera partición del primer disco IDE) son de tipo bloque.



```
Terminal
Archivo Editar Ver Terminal Ir a Ayuda
guadalinux:/dev# ls -l tty0 dsp fd0 hda1
crw-rw-r-- 1 root audio 14, 3 2003-12-30 13:32 dsp
brw-rw-r-- 1 root floppy 2, 0 2003-12-30 13:32 fd0
brw-rw-r-- 1 root disk 3, 1 2003-12-30 13:32 hda1
crw----- 1 root root 4, 0 2004-01-08 20:39 tty0
guadalinux:/dev#
```

Existen otros tipos de ficheros, denominados enlaces; los hay también de dos tipos, duros y blandos. Los enlaces duros resultan en que un mismo fichero aparezca más de una vez, es decir, aparezca en dos sitios a la vez, pero haciendo referencia a los mismo datos. Sin embargo los enlaces simbólicos indican que los accesos deben redirigirse a otro fichero diferente. Ambos se crean con el comando “ln”, añadiendo el parámetro “-s” para el caso del enlace simbólico.



```
Terminal
Archivo Editar Ver Terminal Ir a Ayuda
alumno@guadalinux:~$ ls -l
total 4
-rw-r--r-- 1 alumno alumno 1024 2004-05-22 20:08 archivo1
alumno@guadalinux:~$ ln archivo1 archivo2
alumno@guadalinux:~$ ls -l
total 8
-rw-r--r-- 2 alumno alumno 1024 2004-05-22 20:08 archivo1
-rw-r--r-- 2 alumno alumno 1024 2004-05-22 20:08 archivo2
alumno@guadalinux:~$ ln -s archivo1 archivo3
alumno@guadalinux:~$ ls -l
total 8
-rw-r--r-- 2 alumno alumno 1024 2004-05-22 20:08 archivo1
-rw-r--r-- 2 alumno alumno 1024 2004-05-22 20:08 archivo2
lrwxrwxrwx 1 alumno alumno 8 2004-05-22 20:14 archivo3 -> archivo1
alumno@guadalinux:~$
```

Los duros están restringidos al mismo sistema de ficheros (los simbólicos no).

En la imagen puedes ver cómo al fichero “archivo1” se le ha creado un enlace duro primero y simbólico después. El enlace duro aparece de forma idéntica a como lo hace el original; aunque aparezca con su mismo tamaño, la información no está duplicada, y de hecho podemos eliminar uno de los dos, y

el otro seguiría teniendo la información original; la información no se elimina realmente hasta que no se borra el último enlace duro (con el comando “rm”). El número delante del usuario propietario del archivo indica el número de enlaces duros que posee.

Sin embargo el simbólico aparece de forma distinta; la “l” del principio lo identifica como lo que es, y ocupa un pequeño espacio en disco, independientemente del tamaño del archivo al que apunta; dicho archivo se muestra marcado con el símbolo “->”, y representa el archivo que hay que buscar cuando se acceda al fichero “archivo3”. Los enlaces simbólicos se borran con el comando “unlink”.

Montar y desmontar dispositivos

Antes de poder utilizar un sistema de ficheros es necesario realizar una operación conocida como “montado”; igualmente antes de extraerlo de la unidad correspondiente hay que “desmontarlo”. De esta forma el sistema es informado de la presencia de la unidad y puede optimizar la transferencia de datos con él.

Para montar una partición hay que escribir: “mount -t tipo dispositivo directorio”, siendo “tipo” el tipo de sistema de ficheros, “dispositivo” su nombre en /dev y directorio el punto de montaje. Pero seguramente ya conozcas que desde el entorno gráfico es posible montar y desmontar ciertas unidades, pero no es posible configurarlas ni añadir unidades nuevas. Pulsa con el botón derecho sobre una zona vacía del escritorio y selecciona la opción “Discos”, verás la lista de unidades disponibles (no intentes montar ninguna que esté vacía porque dará un error).

Esa lista que aparece se controla desde un archivo, en concreto el /etc/fstab, que puedes ver en la imagen siguiente.

```
Terminal
Archivo  Editar  Ver  Terminal  Ir a  Ayuda
guadalinux:~# cat fstab
# /etc/fstab: static file system information.
#
# The following is an example. Please see fstab(5) for further details.
# Please refer to mount(1) for a complete description of mount options.
#
# Format:
#<file system> <mount point> <type> <options> <dump> <pass>
/dev/hda2 / ext3 defaults,errors=remount-ro 0 1
proc /proc proc defaults 0 0
/swapfile none swap sw 0 0
none /proc/bus/usb usbdevfs rw 0 0
/dev/fd0 /floppy vfat defaults,user,noauto,showexec,umask=022 0 0
/dev/cdrom /cdrom iso9660 defaults,ro,user,noexec,noauto 0 0
/dev/sda1 /mnt/USB1 vfat defaults,rw,noauto,user 0 0
/dev/sdb1 /mnt/USB2 vfat defaults,rw,noauto,user 0 0
/dev/hda1 /mnt/WindowsXP1 auto noauto,user,exec,uid=1000,gid=1000 0 0
guadalinux:~#
```

Cada fila representa una unidad diferente; la primera columna identifica el fichero de dispositivo asociado a la unidad; por ejemplo vemos las particiones del disco duro (hda1 y hda2), la disquetera y el CD-ROM (fd0 y cdrom), entradas para dispositivos USB como llaveros, reproductores MP3, cámaras, ... (sda1 y sdb1), y un par de entradas especiales (necesarias para el archivo swap y el sistema USB).

La segunda columna indica el subdirectorio a través del cual estará disponible el contenido de la unidad o “punto de montaje”, y debe estar creado con anterioridad (aunque su contenido aparecerá como vacío hasta que se monte la unidad correspondiente).

La tercera columna define el sistema de ficheros a utilizar para acceder a la unidad, y evidentemente debería coincidir con el que realmente existe en la unidad; podemos ver los ya conocidos ext3, iso9660, swap y vfat. El parámetro “auto” indica que debe autodetectarse cada vez que se vaya a proceder al montaje; en este caso se realiza esta operación sobre hda1, aunque no es mala idea hacer lo mismo con la disquetera (fd0) si planeamos utilizar alternativamente discos formateados en vfat y ext2.

La siguiente columna, la de opciones, se utiliza para modificar de alguna manera el comportamiento estándar al montar la unidad. Por ejemplo el parámetro “noauto” indica que esa unidad no debe montarse automáticamente al iniciar el sistema, “user” si la puede montar un usuario, “ro” si se monta para sólo lectura, “rw” si se hace en modo lectura-escritura,... No todas las opciones son válidas para todos los sistemas de ficheros (cada uno tiene las suyas propias).

Las dos últimas columnas son utilizadas por dos herramientas, “dump” y “fsck” respectivamente, y generalmente se encuentran siempre con el valor 0, salvo para la partición de sistema, que vale 1 en la última columna.

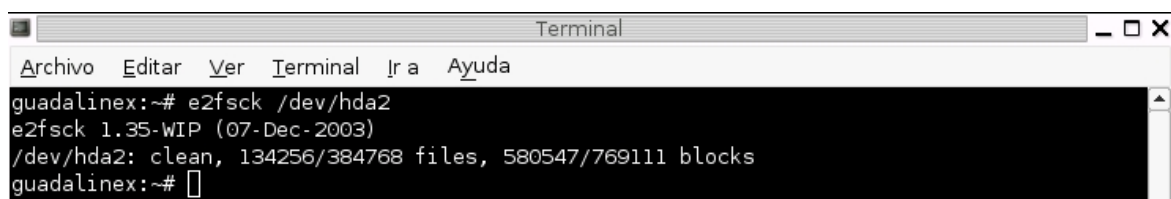
Con este archivo configurado correctamente los dispositivos se pueden montar simplemente con “mount dispositivo”, siendo dispositivo la entrada en la primera columna.

Para desmontar una unidad siempre puedes escribir “umount dispositivo” o pulsar con el botón derecho sobre él si estás en el entorno gráfico y seleccionar “Desmontar el volumen”. Asegúrate de hacerlo antes de sacar la unidad porque puedes perder algún dato en el peor de los casos estropear el sistema de ficheros de esa unidad.

Manejando ext2

A diferencia de otros sistemas operativos, Linux no posee herramientas para defragmentar el sistema de ficheros, realmente porque por las propias características de control que posee ext2 no suele ser necesario realizar este proceso.

Sin embargo sí existen herramientas de reparación que, incluso, se ejecutan automáticamente cada vez que al arrancar se detecta que en la última sesión no fueron cerrados apropiadamente. El comando para reparar el sistema de ficheros es e2fsck, y tan sólo hay que decirle el sistema a reparar (o simplemente comprobar), en este caso /dev/hda2. Tras la comprobación se nos muestra un pequeño resumen del número de archivos y bloques respecto al máximo posible. Hay que tener en cuenta que para realizar esta comprobación el sistema de ficheros debe estar desmontado o montado como sólo lectura, ya que de otra forma podría producirse la pérdida de datos.

A screenshot of a terminal window titled "Terminal". The window has a menu bar with "Archivo", "Editar", "Ver", "Terminal", "Ir a", and "Ayuda". The terminal output shows the command "e2fsck /dev/hda2" being executed. The output text is: "e2fsck 1.35-WIP (07-Dec-2003)", "/dev/hda2: clean, 134256/384768 files, 580547/769111 blocks", and the prompt "guadalinux:~#".

```
guadalinux:~# e2fsck /dev/hda2
e2fsck 1.35-WIP (07-Dec-2003)
/dev/hda2: clean, 134256/384768 files, 580547/769111 blocks
guadalinux:~#
```

Los sistemas de archivos ext2 pueden convertirse de forma sencilla a ext3, sin tener que formatear particiones ni perder la información en ella contenida. Lo único que hay que hacer es añadir el registro de transacciones al sistema de ficheros que ya existe; tampoco es necesario desmontar el sistema de ficheros. Si la partición que se quiere convertir es hda2, la orden es:

```
# tune2fs -j /dev/hda2
```

No hay que olvidar modificar el archivo `/etc/fstab` para reflejar este cambio (modificar `ext2` por `ext3`).

Este comando también puede usarse para establecer otras opciones del sistema de ficheros `ext2` (o `ext3`), por ejemplo cada cuánto tiempo realizar una comprobación automática. Para realizar esta operación automáticamente cada vez que se monte 30 veces:

```
# tune2fs -c 30 /dev/hda2
```

Y para realizarlo cada 7 días:

```
# tune2fs -i 7d /dev/hda2
```

Si se le añade la letra “w”, la cuenta se realiza en semanas (si es la “m” en meses).

Gestión de la memoria virtual

La memoria es uno de los bienes más preciados de una máquina, ya que es en ella donde se ejecutan los programas y se almacenan temporalmente ciertos datos. Por otro lado es un recurso que aunque puede ser más o menos grande, es limitado. Esto significa que una vez agotada la memoria no se pueden ejecutar más programas.

Sin embargo los sistemas operativos desde hace tiempo proveen un mecanismo para que esta situación no ocurra (o al menos suceda con menos frecuencia): la memoria virtual. Este tipo de memoria no es memoria real, sino que se utiliza un espacio del disco duro para que haga las funciones de memoria, de manera que datos o programas que están abiertos pero no se están utilizando pasan a un segundo plano (a la memoria virtual) dejando libre la memoria principal para otro programa. Cuando estos datos vuelven a ser necesarios se intercambian con otros que ya no lo son. Aunque se pierde algo de tiempo en estos intercambios, se obtiene un mejor funcionamiento en conjunto, sobre todo si se trabaja con varias aplicaciones que necesitan una gran cantidad de memoria para funcionar.

La memoria virtual se denomina genéricamente en Linux como “swap” y generalmente se crea durante el proceso de instalación. Si más adelante se necesita mayor cantidad puede crearse otro espacio virtual.

```
top - 21:09:38 up 5:38, 2 users, load average: 0.62, 0.41, 0.32
Tasks: 69 total, 1 running, 68 sleeping, 0 stopped, 0 zombie
Cpu(s): 26.0% user, 5.6% system, 0.0% nice, 68.4% idle
Mem: 256452k total, 251900k used, 4552k free, 9400k buffers
Swap: 377488k total, 49220k used, 328268k free, 69000k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 607 root        5   -10 101m  63m 5880 S   6.6  25.3   8:10.00 XFree86
21411 alfabet   15    0  6616  6616 5372 S   4.3   2.6   0:00.16 screenshot
  836 alfabet   16    0  9896  9296 7860 S   3.6   3.6   0:25.30 wnck-applet
  717 alfabet   15    0 72204   66m  31m S   1.3  26.4   5:46.43 soffice.bin
20998 root       16    0 1024 1024  820 R   1.0   0.4   0:00.34 top
  715 alfabet   19    0  5788  4536 4080 S   0.7   1.8   2:45.54 gkrellm
 1003 root       15    0 10252  8520 7084 S   0.7   3.3   0:11.97 gnome-terminal
  647 alfabet   15    0   624   600  556 S   0.3   0.2   0:44.28 busymouse
  689 alfabet   15    0  2568  2188 2080 S   0.3   0.9   0:01.34 gnome-smproxy
  709 alfabet   16    0  6932  6356 5556 S   0.3   2.5   0:14.57 metacity
    1 root       19    0   488   460  436 S   0.0   0.2   0:18.64 init
    2 root       15    0     0     0    0 S   0.0   0.0   0:00.51 keventd
    3 root       34   19     0     0    0 S   0.0   0.0   0:00.00 ksoftirqd_CPU0
    4 root       15    0     0     0    0 S   0.0   0.0   0:00.75 kswapd
    5 root       25    0     0     0    0 S   0.0   0.0   0:00.00 bdflood
    6 root       15    0     0     0    0 S   0.0   0.0   0:00.14 kupdated
    7 root       16    0     0     0    0 S   0.0   0.0   0:00.34 kjournald
```

En la imagen puedes ver el comando “top” en acción, que monitoriza la carga del sistema; observa que tengo 256 MB de memoria, de los cuales sólo tengo libres menos de 5 MB; sin embargo tengo unos 377 MB de swap de los que se utilizan unos 50 MB, con lo que podría seguir arrancando aplicaciones sin que me saliera el mensaje “Memoria insuficiente”.

Hay dos alternativas para establecer un espacio de memoria virtual: crear una partición independiente en el disco duro, o crear un archivo en alguna partición ya existente. La primera opción resultará en un mejor rendimiento, pero la segunda es más cómoda y fácil de hacer.

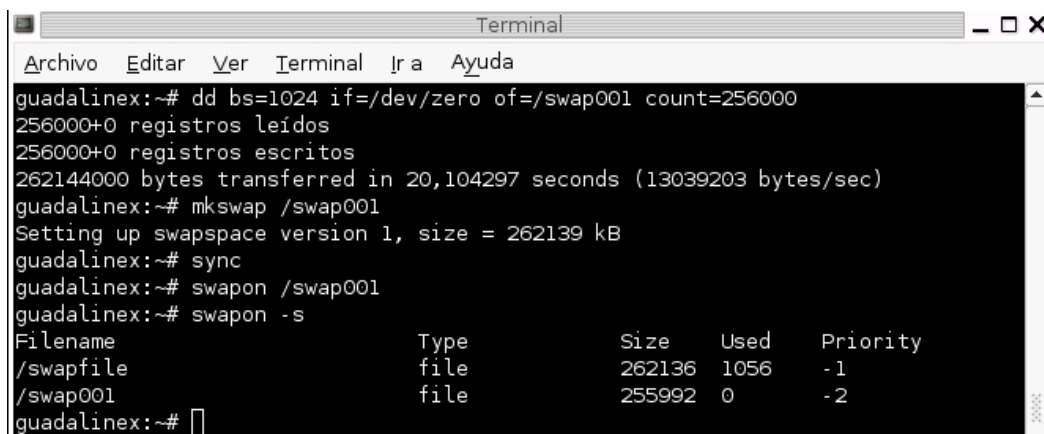
Para establecer una partición como swap primero hay que crearla; puede utilizarse el programa “qtparted” a través del lanzador “Gksu” (Aplicaciones -> Herramientas del sistema”) para el usuario root.

Lo único que hay que hacer es crear una partición en el espacio libre o seleccionar una que ya exista y formatearla como “Linux swap”. El tamaño puede ser cualquiera, pero se suele poner entre 1,5 y 2,5 veces la cantidad de RAM, dependiendo del uso que se le de al ordenador y la cantidad de ésta.

ADVERTENCIA: Al realizar este proceso se eliminarán todos los datos de la partición seleccionada. Piensa dos veces la partición que vas a destinara a swap.

La otra opción es crear un archivo en la partición del sistema; la explicación excede las pretensiones de este curso, aunque los pasos son muy sencillos, como puedes ver en la imagen. Si queremos un archivo swap de unos 256 MB aproximadamente (se especifica en el parámetro “count”):

```
# dd bs=1024 if=/dev/zero of=/swap001 count=256000
# mkswap /swap001
# sync
# swapon /swap001
```



```
Terminal
Archivo  Editar  Ver  Terminal  Ir a  Ayuda
guadalinux:~# dd bs=1024 if=/dev/zero of=/swap001 count=256000
256000+0 registros leídos
256000+0 registros escritos
262144000 bytes transferred in 20,104297 seconds (13039203 bytes/sec)
guadalinux:~# mkswap /swap001
Setting up swapspace version 1, size = 262139 kB
guadalinux:~# sync
guadalinux:~# swapon /swap001
guadalinux:~# swapon -s
Filename                                Type      Size      Used      Priority
/swapfile                               file      262136    1056      -1
/swap001                                file      255992    0         -2
guadalinux:~#
```

Si ahora escribes “swapon -s” verás en la lista el nuevo espacio swap, además del que ya existía. De esta forma podríamos añadir cuantos archivos o particiones swap vayamos necesitando.

Quotas y límites

En los sistemas Linux existe la posibilidad de limitar los recursos a usuarios o grupos, por ejemplo el máximo número de logins que pueden realizar simultáneamente, el máximo tiempo de CPU, ... Estos límites se controlan a través del fichero /etc/security/limits.conf. Concretamente, en este fichero se controlan los límites sobre los procesos de un usuario.

Abre el programa “nautilus” desde el menú Aplicaciones -> Accesorios y localiza el archivo limits.conf. Aunque el archivo está en inglés es muy fácil modificarlo; en primer lugar todas las líneas que empiezan por “#” son comentarios (es decir, se ignoran), que como puedes ver son todas (no hay establecido ningún límite todavía). Cada línea que vamos a poner tiene cuatro campos:

- domino (domain): será el nombre de usuario o de grupo (precedido por una “@”) al que le vamos a poner algún tipo de restricción; el “*” significa todo el mundo.

- tipo (type): puede tomar dos valores, "soft" y "hard". Representan respectivamente el límite "suave" y el límite "duro". El suave indica la cantidad que no debe ser sobrepasada, pero que se puede superar de forma temporal; el duro es el límite que nunca puede superarse. Por decirlo de algún modo, la diferencia entre ellos nos da un cierto margen que puede traspasarse durante algún tiempo, pasado el cual debemos bajar del límite suave.

- objeto (item): es el elemento que vamos a limitar; de entre las opciones disponibles las más utilizadas son:

- fsize: tamaño máximo de archivos (en KiloBytes).
- memlock: máxima cantidad de memoria (en KiloBytes).
- cpu: máximo tiempo de CPU (en minutos).
- nproc: número máximo de procesos.
- maxlogins: número máximo de logins simultáneos.
- priority: la prioridad con la que ejecutar los procesos de este usuario.

De esta forma, si queremos que nadie realice más de cuatro logins podemos añadir:

```
*      hard      maxlogins      4
```

Y si queremos limitar el tamaño máximo de archivo para el grupo "internet" a 2 MB pero que pueda llegar a 4 de forma puntual:

```
@internet      soft      fsize      2048
@internet      hard      fsize      4096
```

Ten en cuenta que no es posible (ni conveniente) establecer límites para root.

Existe otro mecanismo para establecer límites, pero está pensado para limitar y controlar el uso del espacio en disco disponible en un sistema. Se pueden establecer límites en la cantidad de espacio y el número de ficheros de que puede disponer un usuario o grupo. Para poder utilizarlo se necesitan cuatro elementos:

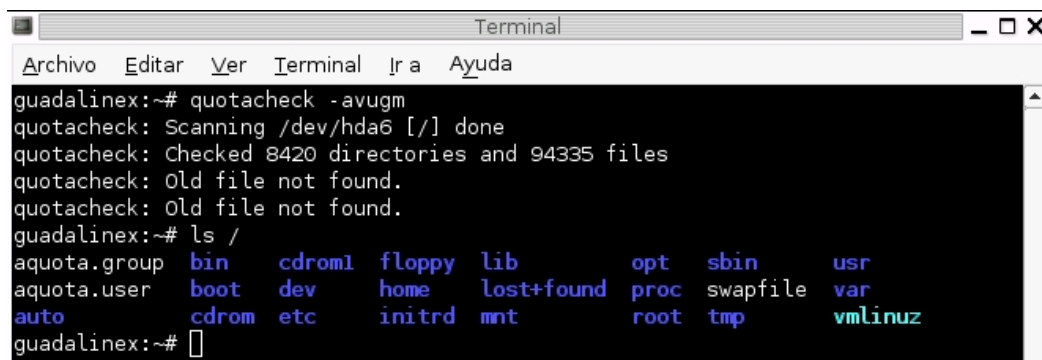
- Configurar el núcleo con la opción "Disk QUOTA support"; los núcleos que se distribuyen ya disponen de esta opción activada.
- Disponer de herramientas para manipular las cuotas; el paquete se denomina "quota" y puede instalarse mediante apt (apt-get install quota). Si no lo tienes instalado ya ahora es el momento de hacerlo.
- Habilitar las cuotas sobre uno o más sistemas de ficheros; para ello hay que editar el archivo /etc/fstab y añadir las opciones "usrquota" y "grpquota" al sistema de ficheros en el que queramos activar las cuotas (por ejemplo el

montado como “/”). Antes de que se te olvide modifica ese archivo ahora; para activar los cambios y no tener que reiniciar escribe “mount -o remount /” (o la partición que corresponda).

- Configurar las cuotas. Para realizar este último punto hay que ejecutar:

```
# quotacheck -avugm
```

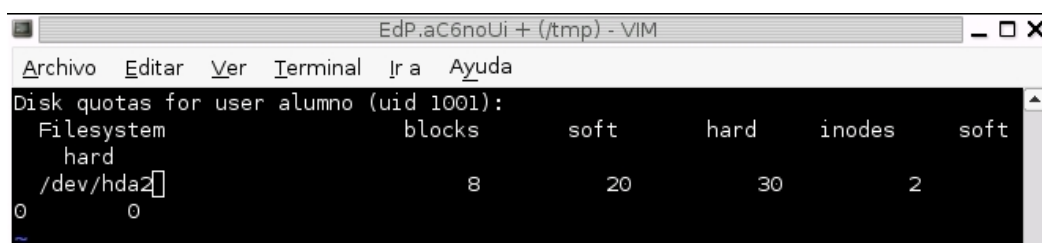
La primera vez que se ejecuta va a crear dos archivos en el directorio raíz: aquota.user y aquota.group, como puedes ver en la imagen.



```
guadalinux:~# quotacheck -avugm
quotacheck: Scanning /dev/hda6 [/] done
quotacheck: Checked 8420 directories and 94335 files
quotacheck: Old file not found.
quotacheck: Old file not found.
guadalinux:~# ls /
aquota.group  bin      cdrom1  floppy  lib      opt      sbin     usr
aquota.user   boot     dev     home    lost+found  proc     swapfile var
auto          cdrom    etc     initrd  mnt      root     tmp      vmlinuz
guadalinux:~#
```

Para editar la cuota de un usuario o grupo se usa el programa “edquota” con la opción “-u” para usuarios y “-g” para grupos; lo único que hay que modificar son los límites “soft” y “hard”, que tienen el mismo significado que antes. Pueden limitarse el número de bloques y el número de nodos-i; los números que aparecen son los que está utilizando el usuario en ese momento.

Este editor es similar a “vi”, por lo que si no estás familiarizado con él te costará un poco manejarlo. De cualquier manera sigue este orden: pulsa la tecla “i”, muévete con las flechas del cursor hasta la cifra del límite soft, escribe 20, colócate sobre el límite hard y escribe 30; después pulsa la tecla de escape (esc) y teclea “:qw” (sin las comillas); si lo has hecho bien volverás al símbolo del sistema.



```
EdP.aC6noUi + (/tmp) - VIM
Archivo  Editar  Ver  Terminal  Ir a  Ayuda
Disk quotas for user alumno (uid 1001):
Filesystem blocks      soft      hard      inodes      soft
  hard
/dev/hda2  8          20        30         2
0          0
```

Se puede cambiar de la misma forma el periodo de gracia con “edquota -t”.

Puedes ver las cuotas establecidas para un usuario llamado “alumno” con “quota -u alumno”, y la de todos los usuarios con “repquota

sistema_de_ficheros”.

Para eliminar las cuotas basta con establecer sus límites soft y hard a cero.

Ejercicios

- 1) Introduce un disquete vacío y crea en él un sistema de archivos vfat.
- 2) Ya conoces el nombre del explorador de carpetas “nautilus”; ¿en qué directorio crees que estará instalado el programa encargado de lanzarlo?
- 3) Introduce un CD-ROM y móntalo
- 4) Crea un enlace simbólico dentro de tu directorio principal que apunte al CD-ROM. Entra en él y comprueba que puedes ver los archivos.
- 4) Establecer una comprobación automática del sistema principal cada 15 días.
- 5) Crea un usuario llamado “ejemplo” y limita el número de procesos que puede ejecutar a 30, el tiempo de uso de la CPU a 1 hora y el tamaño de la memoria que puede utilizar a 8 MB.
- 6) Establece una cuota de disco en la partición principal del sistema (raíz) para el usuario “ejemplo” de 200 bloques y 40 nodos-i, los cuales podrá sobrepasar en igual cantidad durante 2 semanas.
- 7) Comprueba el resultado visualizando las cuotas para el usuario “ejemplo”

Soluciones

- 1) El comando es “mkfs -t vfat /dev/fd0”
- 2) Ese programa es propio de distribuciones como GuadaLinux, pero no es común a todos los Linux, así que empezaríamos por buscarla en /usr; por otro lado, aunque es una aplicación gráfica, no es una aplicación que pertenezca al sistema gráfico (no es indispensable), así que no estará en /usr/X11R6. Utiliza la orden “which nautilus” para conocer su ubicación exacta.
- 3) Como ya existe la entrada en /etc/fstab, puedes poner simplemente “mount

/cdrom".

4) `ln -s /cdrom /home/alumno/cdrom`, suponiendo que el usuario sea "alumno".

5) El comando es: `"tune2fs -i 2w /dev/hda2"`; alternatively se puede utilizar `15d` en lugar de `2w`.

6) En `limits.conf` debes escribir:

ejemplo	hard	nproc	30
ejemplo	hard	cpu	60
ejemplo	hard	memlock	8192

Recuerda que el tiempo de CPU hay que ponerlo en minutos y `memlock` en KB.

7) Se supone que ya tienes activada la cuota para `/`; debes escribir `"edquota ejemplo"` y poner:

/dev/hda2	x	200	400	x	40	80
-----------	---	-----	-----	---	----	----

Las "x" depende de tu sistema y no hay que modificarlas. Por otro lado debes ejecutar `"edquota -t"` y poner un valor de 14 días.

8) Tan sólo escribe `"quota -u ejemplo"`