Guía de administración de Debian GNU/Linux

Jorge Juan Chico <jjchico@imse.cnm.es>
Manuel J. Bellido Díaz <bellido@imse.cnm.es>

Versión 0.4 (3 julio 2002)

Resumen

Este documento es una guía de introducción a la administración de Debian GNU/Linux, correspondiente a la versión 3.0 (woody) de la distribución.

Nota de Copyright

Índice General

Capítulo 1

Introducción

Este documento pretende ser una guía concisa para la administración básica de un sistema Debian GNU/Linux. Está enfocada a la resolución de los aspectos más frecuentes relacionados con la administración del sistema. Con esta guía, un usuario debe ser capaz de configurar el sistema que mejor se adapte a sus necesidades, de mantener en funcionamiento su sistema Debian y de localizar información más detallada en la documentación que se incluye con el propio sistema.

Se supone que el lector está familiarizado con los comandos básicos de UNIX/Linux, o al menos que es capaz de “moverse” por el sistema de ficheros y de editar ficheros de texto.

Salvo el apartado dedicado al gestor de arranque *lilo*, y que es específico de la plataforma i386, el resto debe ser aplicable a cualquier plataforma soportada por Debian.

Capítulo 2

Arranque y parada del sistema

El arranque de un sistema Linux consta de las siguientes fases:

- Ejecución del gestor de arranque (p.ej. *lilo*)
- Carga y ejecución del *kernel*
- Ejecución de *init* (proceso número 1)
- Ejecución de scripts de iniciación genéricos en `/etc/rcS.d`
- Entrada en el *runlevel* por defecto: ejecución de scripts del runlevel en `/etc/rcX.d`, donde *X* el el número del runlevel.

A grandes rasgos el proceso ocurre así: al conectar o reiniciar el ordenador, la BIOS busca en su configuración el dispositivo de arranque por defecto, el cual suele ser un disco duro, indicado generalmente en la configuración de la BIOS por *C*. Entonces carga en memoria el primer sector del dispositivo de arranque y le transfiere el control. Cuando se trata de un disco duro, este primer sector es el *Master Boot Record* (MBR) y contiene, además del código cargado por la BIOS, la tabla de particiones del disco. El código en el MBR, generalmente, lee en la tabla de particiones cual es la partición activa y carga en memoria el primer sector de la misma, transfiriéndole el control. En nuestro caso, este sector estará ocupado por un gestor de arranque (en adelante supondremos que es *lilo*) que nos permitirá arrancar Linux u otro sistema operativo. Opcionalmente, puede instalarse *lilo* en el MBR, tomando antes el control.

Una vez cargado *lilo*, éste se ejecuta, busca el kernel de Linux en una posición conocida del disco, carga el kernel en memoria y le cede el control. El kernel se almacena comprimido en disco, por lo que lo primero que hace el código del kernel que se ejecuta inicialmente es descomprimir el propio kernel y situarlo en memoria. Entonces se ejecuta realmente el kernel y empieza una lista de comprobaciones y activación de módulos internos del mismo. Una vez funcionando, el kernel *monta* el disco principal donde se almacena el sistema operativo (*root filesystem* y ejecuta el primer proceso: *init*. La misión de *init* es ejecutar el resto de procesos del sistema: comprobación de discos, detección/configuración de hardware adicional, apertura de terminales, servidores, etc.

En las siguientes secciones describiremos la configuración del gestor de arranque *lilo*, la operación de *init* y los *runlevels*.

2.1 El gestor de arranque *lilo*

Lilo es el gestor de arranque que nos permite, entre otras cosas, poder elegir el Sistema Operativo que queremos ejecutar al arrancar el ordenador. Durante la instalación, *Lilo* es configurado para arrancar tanto Linux como otros sistemas operativos que tengamos instalados en el sistema, por lo que en la mayoría de los casos no es necesario hacer modificaciones posteriores a menos que deseemos cambiar alguna de la opciones de arranque. De este modo, tras la instalación de Debian, cada vez que arranquemos el sistema se nos mostrará un menú con el que podremos elegir arrancar Linux u otro sistema operativo detectado durante el arranque, que estuviera instalado en otro disco o partición.

Lilo se configura mediante las opciones adecuadas en el fichero `/etc/lilo.conf`. Tras la instalación de *Debian* existe una versión de este fichero bastante completa y con comentarios suficientes para saber qué significa cada opción. A continuación mostramos este fichero con algunas modificaciones para permitir el arranque de un Sistema Operativo alternativo.

```
     # /etc/lilo.conf - See: ‘lilo(8)’ and ‘lilo.conf(5)’,
     # ---------------       ‘install-mbr(8)’, ‘/usr/share/doc/lilo/’,
     #                       and ‘/usr/share/doc/mbr/’.

     # +---------------------------------------------------------------
-+
     # |                         !! Reminder !!                        |
     # |                                                               |
     # | Don’t forget to run ‘lilo’ after you make changes to this     |
     # | conffile, ‘/boot/bootmess.txt’, or install a new kernel.  The |
     # | computer will most likely fail to boot if a kernel-image      |
     # | post-install script or you don’t remember to run ‘lilo’.      |
     # |                                                               |
     # +---------------------------------------------------------------
-+

     # Specifies the boot device.  This is where Lilo installs its boot
     # block.  It can be either a partition, or the raw device, in which
     # case it installs in the MBR, and will overwrite the current MBR.
     #
     boot=/dev/hda

     # Specifies the device that should be mounted as root. (‘/’)
     #
     root=/dev/hda2

     # Enable map compaction:
     # Tries to merge read requests for adjacent sectors into a single
     # read request. This drastically reduces load time and keeps the
     # map smaller.  Using ‘compact’ is especially recommended when
     # booting from a floppy disk.  It is disabled here by default
```

```
# because it doesn't always work.
#
# compact

# Installs the specified file as the new boot sector
#
install=/boot/boot.b

# Specifies the location of the map file
#
map=/boot/map

# You can set a password here, and uncomment the 'restricted' lines
# in the image definitions below to make it so that a password must
# be typed to boot anything but a default configuration.  If a
# command line is given, other than one specified by an 'append'
# statement in 'lilo.conf', the password will be required, but a
# standard default boot will not require one.
#
# This will, for instance, prevent anyone with access to the
# console from booting with something like 'Linux init=/bin/sh',
# and thus becoming 'root' without proper authorization.
#
# Note that if you really need this type of security, you will
# likely also want to use 'install-mbr' to reconfigure the MBR
# program, as well as set up your BIOS to disallow booting from
# removable disk or CD-ROM, then put a password on getting into the
# BIOS configuration as well.  Please RTFM 'install-mbr(8)'.
#
# password=tatercounter2000

# Specifies the number of deciseconds (0.1 seconds) LILO should
# wait before booting the first image.
#
#delay=20

# You can put a customized boot message up if you like.  If you use
# 'prompt', and this computer may need to reboot unattended, you
# must specify a 'timeout', or it will sit there forever waiting
# for a keypress.  'single-key' goes with the 'alias' lines in the
# 'image' configurations below.  eg: You can press '1' to boot
# 'Linux', '2' to boot 'LinuxOLD', if you uncomment the 'alias'.
#
message=/boot/bootmess.txt

prompt
```

```
single-key
delay=100
timeout=100

# Specifies the VGA text mode at boot time. (normal, extended, ask, < mode
#
# vga=ask
# vga=9
#
vga=normal

# Kernel command line options that apply to all installed images go
# here.  See: The 'boot-prompt-HOWO' and 'kernel-parameters.txt' in
# the Linux kernel 'Documentation' directory.
#
append="mem=256M hdc=ide-scsi"

# Boot up Linux by default.
#
default=Linux

image=/vmlinuz
 label=Linux
 read-only
# restricted
 alias=1

image=/vmlinuz.old
 label=LinuxOLD
 read-only
 optional
# restricted
 alias=2

# If you have another OS on this machine to boot, you can uncomment the
# following lines, changing the device name on the 'other' line to
# where your other OS' partition is.
#
other=/dev/hda1
 label=Windows
# restricted
 alias=3
```

A continuación comentamos algunas de las líneas que aparecen en este fichero:

```
boot=/dev/hda
```

especifica la unidad en cuyo sector de arranque en el que debe instalarse lilo. En el ejemplo se ha especificado el primer disco IDE, por lo que lilo se intalará en el Master Boot Record (MBR) de ese disco. Opcionalmente puede indicarse que se instale en una partición, por ejemplo, `boot=/dev/hda2`, en cuyo caso la partición debe estar marcada como activa y ser accesible a la BIOS.

```
root=/dev/hda1
```

Especifica donde está la raiz (*root*) del sistema de ficheros de *Linux*, esto es, la partición en donde está instalado *Linux*. En este caso se trata de la segunda partición del primer disco IDE.

```
message=/boot/bootmess.txt
prompt
delay=100
timeout=100
```

Especifica un fichero que será mostrado al usuario al arrancar *lilo* junto con un tiempo de espera antes de arrancar automáticamente. Normalmente sirve para mostrarle las opciones que tiene, por ejemplo:

```
Elige un sistema operativo.

1 - Linux 2.2.16
2 - Linux 2.2.14
3 - MS-DOS/Windows

En 10 seg. se ejecutará el primero de la lista.
```

En las versiones actuales, *Lilo* muestra automáticamente un menú con las opciones de arranque, por lo que la creación de este fichero no suele ser necesaria y puede eliminarse o comentarse la línea con la opción `message`.

La siguiente línea:

```
append="mem=256M hdc=ide-scsi"
```

indica opciones adicionales que hay que pasar al kernel. En este caso, se indica la cantidad de RAM instalada para un sistema que no es capaz de detectarla automáticamente (puede pasar cuando la cantidad de RAM es superior a 64MB), y también se indica que debe usarse la emulación IDE–SCSI para el dispositivo `hdc`, opción necesaria, por ejemplo, si este dispositivo es una grabadora de CD's con conexión al puerto IDE.

```
image=/vmlinuz
label=Linux
read-only
# restricted
alias=1
```

Describe un kernel de *Linux* a ejecutar junto con varias opciones específicas. Se establece un nombre para esta entrada (`label`) y un alias (1). Aquí pueden establecerse opciones específicas como `root` o `append` cuando se ejecuta este kernel en concreto.

```
other=/dev/hda1
label = Windows
alias = 3
```

Describe la situación de otros sistemas operativos. En este caso se trata de una versión de MS–Windows(TM) situado en la primera partición del primer disco IDE (`/dev/hda1`).

2.2 Proceso `init` y *runlevels*

Una vez que el kernel se ha cargado en memoria, ejecuta el programa `init`, que se convierte en el proceso número 1 y se encarga de ejecutar el resto de programas que hacen que el sistema funcione. La operación de *init* se configura en el fichero `/etc/inittab`. Empezando en este fichero se puede seguir paso a paso el proceso de arranque (y parada) del sistema. Lo importante a saber aquí es que *init* no hace demasiado por si mismo, sino que se limita a ejecutar una serie de *scripts* que activan ordenadamente los diferentes servicios que hacen funcionar el sistema.

En primer lugar se ejecutan por orden alfabético todos los *scripts* que se encuentren bajo el directorio `/etc/rcS.d` que comiencen por 'S' con "start" como argumento. Por convenio estos *scripts* se nombran comenzando por un número de dos cifras para establecer el orden adecuado. Cada *script* tiene una función particular, por ejemplo:

- `S10checkroot.sh` comprueba el sistema de ficheros.
- `S20adjtimex` ajusta el reloj del sistema.
- `S40networking` activa los interfaces de red.

En realidad, y por convenio, estos *scripts* no se sitúan directamente en `/etc/rcS.d`, sino que se guardan en el directorio `/etc/init.d` y desde ahí se hacen enlaces simbólicos a `/etc/rcS.d`. De esta forma es más fácil añadir, eliminar y ordenar los componentes.

Las tareas realizadas por los *scripts* en `/etc/rcS.d` son las básicas para arrancar el sistema. Luego se ejecutan otra serie de *scripts* correspondientes a un *runlevel*. Los *runlevels* son un mecanismo para permitir que el ordenador trabaje con diferentes configuraciones de arranque (diferentes servicios, etc.). Los *runlevels* se numeran del 0 al 6. El 0 se ejecuta para parar el sistema (*halt*), el 6 para reiniciar (*reboot*) y el 1 para arrancar en modo *single user*, que viene a ser una cofiguración mínima para realizar tareas de administración. El resto de los *runlevels* son para funcionamiento normal. El *runlevel* por defecto es el 2 (se configura en `/etc/inittab`), empleando los otros sólo en situaciones especiales. En *Debian*, los *runlevels* del 2 al 5 se configuran inicialmente de forma idéntica.

Así, el proceso de arranque suele continuar ejecutando los *scripts* del *runlevel* correspondiente situados en `/etc/rcX.d`, donde *X* es el número del *runlevel*. La forma de ejecutar estos *scripts* es análoga

al caso anterior, salvo que ahora se ejecutan primero los *scripts* que comiencen con 'K' con argumento "stop" y luego los que comienzan con 'S' con argumento "start". La idea es que cada *script* gestione una tarea o servicio. Este servicion se inicia cuando el *script* se ejecuta con el argumento "start" y se detiene cuando se usa el argumento "stop". De esta forma en cada *runlevel* puden detenerse los servicios que no se necesiten y activarse aquellos que interese. Este sistema también facilita el arranque y parada de servicios, ejecutando estos *scripts* manualmente con el argumento apropiado.

En cualquier momento, el administrador puede hacer que el sistema cambie a otro *runlevel* ejecutando el comando `telinit` con un argumento numérico indicando el nuevo *runlevel*. Por ejemplo:

```
# telinit 3
```

En general, no hay que preocuparse por la configuración del arranque ya que el sistema de instalación se ocupa automáticamente de actualizar la configuración en función de los paquetes instalados.

2.3 Parada del sistema

La parada del sistema se produce cuando se entra en los *runlevel* 0 ó 6. Aunque esto puede hacerse empleando el comando `telninit`, la forma más adecuada es empleando el comando `shutdown`, que además incluye múltiples opciones para programar la parada en un instante dado, avisando a los usuarios previamente. Algunos ejemplos son:

```
# shutdown -r now
```

reinicia el sistema inmediatamente (*reboot*).

```
# shutdown -h now
```

para el sistema inmediatamente (`halt`).

```
shutdown -h +10 "Vamos a parar el sistema en 10
 min."
```

para el sistema dentro de 10 minutos enviando un aviso a todos los terminales.

```
shutdown -r 20:00
```

reinicia el sistema a las 20:00 horas.

El sistema también puede detenerse pulsando la combinación de teclas `CTRL-ALT-SUPR`. Esto indica a *init* que ejecute un *shutdown* inmediatamente y reinicie el sistema. Este comportamiento pude cambiarse editando el fichero `/etc/inittab` para, por ejemplo, hacer una parada en vez de un reinicio. Normalmente, esta operación funciona cuando se ejecuta desde un terminal virtual, no desde X–Window, por lo tanto, si estamos en el entorno gráfico y queremos usar esta opción, primero debemos cambiarnos a un terminal virtual pulsando `CTRL-ALT-F1` y luego reiniciar el sistema con `CTRL-ALT-SUPR`.

Capítulo 3

Sistema de ficheros

En este capítulo se describe tanto el sistema de ficheros como su administracion. Ademas, se incluyen las operaciones básicas que se pueden hacer con el sistema de ficheros, especialmente al manejar medios extraibles (diskettes y CD–ROM's).

3.1 Estructura del sistema de ficheros

El sistema de ficheros en UNIX se compone de un único árbol de directorios que comienza en el directorio principal (/). 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 formas el arbol de directorios y dónde están montados.

Durante el proceso de instalación de Debian, si se realizan varias particiones tipo "Linux", el usuario podrá indicar en qué directorios desea montar cada una de ellas y se le mostrarán las opciones más comunes.

A continuación mostramos un breve esquema de la jerarquía de directorios en un sistema Linux. Una descripción completa de esta jerarquía se encuentra en el *Filesystem Hierarchy Standard* (FHS) que en los sistemas Debian se encuentra, junto con otra información, en el paquete `debian-policy`.

Del directorio raiz cuelgan los siguientes directorios:

```
/
|-bin
|-boot
|-cdrom
|-dev
|-etc
|-floppy
|-home
|-lib
```

```
|-mnt
|-proc
|-root
|-sbin
|-tmp
|-usr
‘-var
```

- `/bin` contiene 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, como imágenes del kernel y datos generados por el gestor de arranque.
- `/cdrom` y `/floppy` son los puntos de montado del CD–ROM y de la disketera, como veremos posteriormente.
- `/dev` contine ficheros especiales que representan dispositivos del hardware y sirven para interaccionar con ellos.
- `/etc` contiene los ficheros de configuración de los diferentes programas.
- `/home` contiene los directorios de inicio de los usuarios.
- `/lib` contiene las librerías dinámicas esenciales del sistema.
- `/mnt` es el punto de montado transitorio de sistemas de ficheros.
- `/proc` es el directorio de acceso a un sistema de ficheros virtual generado por el kernel que permite obtener y enviar información al hardware.
- `/root` es el directorio de inicio del superusuario.
- `/sbin` similar a `/bin`, pero contiene ejecutables de interés principalmente para el administrador: herramientas de detección y reparación de errores, etc.
- `/tmp` contiene datos temporales empleados por diversos programas. Se suele borrar cada vez que se reinicia el sistema.
- `/usr` y `/var` contienen todo lo demás, esto es, todo aquello que no es estrictamente necesario para hacer funcionar el sistema. De esta forma, `/usr` y `/var` pueden estar físicamente en particiones distintas a `/`, lo cual tiene importantes ventajas en cuanto a seguridad y organización. La diferencia fundamental entre `/usr` y `/var` es que `/usr` contiene datos estáticos que no cambian salvo cuando se instalan o desinstalan programas, mientras que `/var` contiene datos dinámicos que sufren alteraciones durante la operación normal del sistema. Esta distinción permite montar `/usr` en modo de sólo lectura, y compartirlo entre distintas máquinas a través de la red.

Veamos la estructura y los componentes más importantes de `/usr` y `/var`:

```
/
‘-usr
  |-X11R6
  |-bin
  |-doc
  |-info
  |-lib
  |-local
  |-man
  |-sbin
  ‘-share
```

- `/usr/X11R6` contiene una estructura similar a `/usr` pero para los programas y componentes del entorno gráfico X–Window.
- `/usr/bin` y `/usr/sbin` hacen la misma función que `/bin` y `/sbin` pero para aquellas instrucciones que no son indispensables para el arranque y funcionamiento del sistema
- `/usr/doc` contiene información adicional suministrada con los distintos programas y en formatos especiales como html, PostScript, etc. En Debian, cada paquete instala un directorio en `/usr/doc` con el mismo nombre del paquete y que contiene información sobre el mismo.
- `/usr/lib` contiene librerías dinámicas usadas por diversos programas.
- `/usr/local` contiene una estructura similar a `/usr` pero para programas instalados fuera de la distribución (instalación manual u otros procedimientos). Ninguna distribución de Linux debe instalar programas o ficheros en este directorio. `/usr/local` es un buen candidato para hacer que resida en un disco o partición propios.
- `/usr/man` y `/usr/info` contienen las páginas de manual y las páginas info respectivamente.

Algunos componentes de la estructura `/usr/var` son:

```
/
‘-var
  |-log
  ‘-spool
    |-lpd
    ‘-mail
```

- `/var/log` contiene los ficheros que registran información generada por diversos procesos del sistema, como mensajes de error, etc. El sistema gestiona automáticamente este directorio, empaquetando y eliminando los mensajes antiguos.
- `/var/spool` contine datos correspondientes a trabajos pendientes de realización. Destacan las colas de impresión en `/var/spool/lpd` y los buzones de entrada del correo en `/var/spool/mail`.

3.2 Ficheros especiales

En UNIX, aparte de los directorios y los ficheros convencionales, existe una serie de ficheros "especiales". Describiremos brevemente cuales son y que función tienen.

3.2.1 Enlaces "duros"

No se trata en sí de un tipo especial de ficheros, sino más bien de la capacidad de UNIX de que un mismo fichero pueda aparecer en más de una entrada de directorio. Esto es, el fichero puede aparecer en varios sitios a la vez, pero hace referencia a los mismos datos. El número de enlaces de un fichero se se puede averiguar con el comando `ls`:

```
$ ls -l
drwxr-xr-x    4 jjchico  profes       4096 ene 17 00:32 ./
drwxr-xr-x    3 jjchico  profes       4096 ene 14 21:34 ../
drwxr-xr-x    2 jjchico  profes       4096 ene 17 00:29 latex/
drwxr-xr-x    2 jjchico  profes       4096 ene 23 09:51 sgml/
-rw-r--r--    1 jjchico  profes        458 ene 21 23:19 Makefile
-rw-r--r--    1 jjchico  profes      84093 ene 22 20:14 admin.ps
```

Puede verse que los ficheros sólo tienen un enlace mientras que los directorios siempre tienen al menos dos, ya que siempre aparecen en el directorio que los contiene y en ellos mismos en forma de "`.`". Además, poseen un enlace adicional por cada subdirectorio que contienen, ya que en estos aparece como "`..`".

Para los ficheros se pueden crear enlaces con el comando `ln`. Por ejemplo:

```
$ ln Makefile Makefile2
```

crea un nuevo enlace al fichero `Makefile` con nombre `Makefile2`. El comando `ln` usado de esta forma funciona como el comando `cp`, salvo que la nueva copia no es un nuevo fichero, sino una nueva entrada de directorio para el fichero que ya existía. Todos los enlaces "duros" a un fichero son equivalentes entre si.

Los enlaces "duros" están limitados a un mismo sistema de ficheros, esto es, no se puede hacer un enlace de un fichero a un directorio que se encuetre en un dispositivo físico distinto (disco, partición, etc.).

3.2.2 Enlaces "simbólicos"

Una variante a los enlaces "duros" son los enlaces "simbólicos". Éstos no son realmente nuevas entrada de directorio para ficheros existentes, sino unos ficheros especiales que indican que los accesos a los mismos debe redirigirse a otro fichero diferente. En la práctica funcionan como los enlaces "duros" y no tienen sus limitaciones, por lo que su uso es más frecuente.

Los enlaces simbólicos también se crean con el comando `ln`, pero esta vez usando la opción `-s`. Por ejemplo:

```
$ ln -s Makefile Makefile2
$ ls -l
drwxr-xr-x    2 jjchico  profes       4096 ene 23 10:24 ./
drwxr-xr-x    4 jjchico  profes       4096 ene 23 10:14 ../
-rw-r--r--    1 jjchico  profes        458 ene 21 23:19 Makefile
lrwxrwxrwx    1 jjchico  profes          8 ene 23 10:24 Makefile2 -
> Makefile
```

Vemos que el enlace simbólico se marca con una `l` en el campo de tipo y se indica cual es el fichero real al que apunta. Hay que tener en cuenta que el enlace se realiza a la ruta que se indique en el primer argumento del comando `ln`. Si esta ruta es relativa, es conveniente ejecutar el comando desde el directorio donde vaya a estar el enlace, para que éste se cree correctamente.

3.2.3 Ficheros de dispositivo

En UNIX, la mayoría de los dispositivos físicos están representados en el sistema de ficheros por ficheros especiales de tipo dispositivo (*device*). Estos ficheros se agrupan dentro del directorio `/dev`. Veamos un ejemplo:

```
$ cd /dev
$ ls -l ttyS0 fd0 hda hda1 dsp
crw-rw----    1 root     audio     14,   3 abr 22  2000 dsp
brw-rw----    1 root     floppy     2,   0 abr 22  2000 fd0
brw-rw----    1 root     disk       3,   0 abr 22  2000 hda
brw-rw----    1 root     disk       3,   1 abr 22  2000 hda1
crw-rw----    1 root     dialout    4,  64 ene  9 12:38 ttyS0
```

Vemos que hay dos tipos de dispositivos: tipo carácter (marcados con `c`) y tipo bloque (marcados con `b`). Con los dispositivos 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 tamaño determinado. Cada dispositivo se caracteriza por dos números: el número mayor y el número menor. El número mayor indica el tipo de dispositivo (disco, puerto serie, etc.) y el menor identifica un dispositivo concreto dentro de un tipo.

Existen ficheros de dispositivo de muchos tipos y generalmente no es necesario crear ficheros nuevos. Gran cantidad de ellos representan hardware que no tiene por qué estar instalado. Si se necesitan nuevos ficheros de dispositivo, el script `/dev/MAKEDEV` puede ejecutarse con los argumentos adecuados para crear los dispositivos o grupos de dispositivos que se necesiten. Para más información sobre tipos de dispositivos y su creación ver `MAKEDEV(8)`. Como puede verse en el ejemplo, el control a los dispositivos puede establecerse asignando los permisos adecuandos.

Los ficheros de dispositivo permiten acceder directamente al hardware que representan, haciendo operaciones de lectura, escritura y control sobre los ficheros de dispositivo como si fueran ficheros convencionales. En realidad, los ficheros de dispositivo es una de la grandes ideas de UNIX. Posteriormente veremos cómo emplear directamente los dispositivos de disco.

3.2.4 Tuberías (fifo's)

Las tuberías son ficheros especiales que no almacenan información en el disco, sino que la guardan en memoria en espera de que algún programa la lea. En este sentido funcionan como un buffer tipo FIFO (*First In First Out*). Permiten conectar la salida de un programa con la entrada de otro, de forma parecida a como lo hace el shell con el operador "`|`". Last tuberías se crean con el comando `mkfifo(1)`. Por ejemplo, primero creamos una tubería llamada `mififo`:

```
$ mkfifo mififlo
$ ls -l
total 0
prw-r--r--    1 jjchico  profes          0 ene 23 12:05 mififo|
```

Vemos como las tuberías se marcan con `p` en el campo de tipo y con `|` en el nombre (aquí la opción `-F` de `ls` está implícita). Ahora ejecutamos un proceso que escriba en la tubería. El proceso quedará bloqueado en espera de que alguien lea de la tubería, por eso lo ejecutamos en el *background*:

```
$ ls > mififo &
$ ls -l
prw-r--r--    1 jjchico  profes          0 ene 23 12:05 mififo|
```

Ahora leemos de la tubería. Esto hace que el proceso `ls` pueda escribir los datos y termine:

```
$ cat mififo
mififo|
[2]+  Done                    ls $LSOPTIONS >mififo
```

3.3 Administracion del sistema de ficheros

3.3.1 Montar y desmontar dispositivos.

En general, en los sistemas UNIX es necesario *montar* los dispositivos extraibles antes de que puedan ser usados por el sistema, y también es necesario "desmontarlos" antes de extraerlos de las unidades correspondientes. De esta forma, el sistema es informado de la presencia del medio y puede optimizar la transferencia de datos con el mismo.

En muchos entornos gráficos (como GNOME o KDE) existen iconos que permiten un acceso directo a los dispositivos, esto es, el dispositivo se monta automáticamente al pulsar en el icono y aparece una ventana con el contenido de la unidad. En estos casos suele ser necesario desmontar el dispositivo antes de extraer el medio, lo cual se realiza con la opción adecuada del menú desplegable del dispositivo (accesible a menudo pulsando la tecla derecha del ratón sobre el icono). Esta operación es especialmente importante en diskettes, ya que los CD–ROM's no podrán ser extraidos hasta que se desmonten, pero los diskettes pueden ser extraidos sin desmontar, lo cual puede tener consecuencias desagradables como

pérdida de datos o incluso bloqueo de la unidad. En este caso, es necesario volver a introducir el medio y hacer la operación de desmontar.

En realidad, las operaciones de montado y desmontado las llevan a cabo los comandos `mount` y `umount`, por ejemplo:

```
# mount /dev/hdc /cdrom
```

montaría el dispositivo de CD–ROM situado en `/dev/hdc` en el directorio `/cdrom`. Antes del montado, el directorio `/cdrom` debe estar creado. Tras el montado, el contenido del dispositivo montado "aparece" en `/cdrom` como si se tratara de un directorio más del sistema. De la misma forma de haría para un diskette:

```
# mount /dev/fd0 /floppy
```

y el contenido del diskette aparecerá en `/floppy`. De esta forma, se accede a un dispositivo montado como si se tratara de un directorio más del sistema. Los directorios `/cdrom` y `/floppy` son creados por el proceso de instalación precisamente para el el propósito descrito.

En cualquier momento puede saberse que dispositivos están montados en el sistema ejecutando el comando `mount` sin argumentos:

```
$ mount
/dev/hda3 on / type ext2 (rw,errors=remount-ro,errors=remount-ro)
proc on /proc type proc (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/hda1 on /dos type vfat (rw,umask=007,gid=101)
/dev/fd0 on /floppy type vfat (rw,noexec,nosuid,nodev,user=jjchico)
/dev/hdc on /cdrom type iso9660 (ro,unhide,user=jjchico)
```

Puede verse que `/dev/hda3` (la tercera partición del primer disco IDE) es el sistema de ficheros principal (/), y que la primera partición de este mismo disco es de formato *vfat* y está montada en `/dos`. En este caso se trata de una partición en la que hay instalado otro sistema operativo. También vemos que hay un diskette montado en `/floppy` y un CD–ROM en `/cdrom`. Los dispositivos *proc* y *devpts* son dispositivos virtuales pero que se comportan como si fueran dispositivos reales. Aunque su descripción sale del ámbito de este apartado, diremos, por ejemplo, que `/proc` contiene ficheros que permiten obtener información sobre el hardware del sistema y su configuración. El contenido de estos ficheros es generado dinámicamente por el sistema cuando se leen. Es divertido pasearse por `/proc` y curiosear el contenido de sus ficheros y directorios.

Linux reconoce muchos formatos posibles de discos, diskettes y CD–ROM. En la mayoría de los casos el sistema identifica automáticamente el formato del dispositivo, pero cuando esto no es posible, podemos indicar el formato con la opción `-t` de `mount`, por ejemplo:

```
# mount -t vfat /dev/fd0 /floppy
```

montaría un diskette con formato MS–DOS y soporte para nombres largos. Otros formatos comunes son: msdos (formato MS–DOS normal), iso9660 (formato de los CD–ROM's) y ext2 (formato nativo de Linux). Para más información sobre formatos y opciones de ver `mount(8)`

3.3.2 Fichero /etc/fstab

Para facilitar el uso de `mount` y para indicar al sistema que dispositivos forman el sistema de ficheros, existe el fichero `/etc/fstab`. En él se indican que dispositivos han de montarse en qué lugar y con que opciones. Por ejemplo, con un `/etc/fstab` correctamente configurado es posible montar el CD–ROM indicando simplemente el punto de montaje:

```
$ mount /cdrom
```

De esta forma, el usuario se puede olvidar de en qué dispositivo se encuentra el CD–ROM, etc. Esta información está pre–configurada en `/etc/fstab`. Un ejemplo de fichero `/etc/fstab` es el siguiente:

```
# /etc/fstab: static file system information.
#
#<file system> <mount point> <type> <options> <dump> <pass>
/dev/hda3   /        ext2     defaults,errors=remount-ro  0    1
/dev/hda2   none     swap     sw                          0    0
proc        /proc    proc     defaults                    0    0
/dev/hda1   /dos     vfat     defaults,umask=007,gid=101
/dev/cdrom  /cdrom   iso9660  ro,user,noauto,unhide
/dev/fd0    /floppy  vfat     defaults,noauto,user
```

Algunos puntos interesantes de este fichero son los siguientes:

- En `/dev/hda3` se encuentra la partición principal de Linux (root file system)
- La línea que hace referencia a `/dev/hda2` indica donde se encuentra la partición de swap
- El CD–ROM y la disketera se montarán en `/cdrom` y `/floppy` respectivamente, pero no se montarán automáticamente al arrancar (opción noauto), aunque si podrán ser montados posteriormente por usuarios “normales” (opción user)
- En `/dev/hda1` hay una partición tipo vfat que se montará automáticamente al arrancar

Para finalizar esta sección, mencionar que existe un sistema para montar automáticamente dispositivos extraibles. Este sistema requiere del módulo apropiado del kernel (autofs.o) y de los programas que lo controlan. En los sistemas Debian, estos programas se encuentran en el paquete `autofs`. El usuario que esté interesado por esta opción puede instalar el paquete y consultar la documentación que adjunta.

3.3.3 Distribuyendo el sistema de ficheros a través de varias particiones

En general es aconsejable instalar todo el sistema de ficheros de Linux en varias particiones. Como mínimo es dos separando estrictamente lo que es el sistema Linux en si de los datos personales. En este apartado pretendemos mostrar como se puede llevar a cabo este proceso una vez que se ha instalado el sistema.

Creando nuevas particiones

Lo primero que es necesario es crear o disponer de una nueva partición en disco duro. Para ello es necesario, o bien disponer de un nuevo disco duro o de espacio libre en el actual. Si se dispone de este espacio el primer paso consiste en crear una o mas particiones tipo Linux extended 2. Para crear particiones se utiliza el comando `fdisk` o `cfdisk`:

```
# cfdisk /dev/hda
```

Tras la ejecución de `cfdisk`, el programa muestra una pantalla como la siguiente:

```
                                   cfdisk 2.10s

                              Unidad de disco: /dev/hda
                              Tamaño: 20020396032 bytes
                 Cabezales: 255   Sectores por pista: 63   Cilindros: 2434

         Nombre      IndicadoresTipo de parTipo de sistema d[Etiqueta]     Tamañ
       ------------------------------------------------------------------
--------------
         hda1        Inicio      Primaria  Win95 FAT32                      524
         hda5                    Lógica    Linux ext2                       500
         hda6                    Lógica    Linux swap                        24
         hda7                    Lógica    Win95 FAT32                      952









          [Iniciable] [Suprimir]  [ Ayuda  ]  [Maximizar] [Imprimir]
          [ Salir  ]  [  Tipo  ]  [Unidades]  [Escribir]
```

Dentro de este programa se pueden hacer cambios en las particones del disco duro. Este proceso es delicado y, por ello, es necesario tener en cuenta varias cosas:

- Los cambios propuestos no se hacen efectivos hasta que se escribe la tabla de particiones.
- Si el sistema esta funcionando combiene conocer previamente que particiones estan montadas y activas (con el comando `mount` o `df`) para no modificarlas pues se dañaría el sistema.
- Una vez escrita la tabla de particiones, se ha asignado un tipo de sistema de fichero a cada una de las particiones establecidas en dicha tabla pero no se les ha dado el formato adecuado.

Formateando nuevas particiones

Una vez creadas las nuevas particiones es necesario formatearlas. Esto crea un sistema de ficheros tipo *second extended* (ext2) en la particion. Vamos a describir de forma breve las caracteristicas basicas de los sistemas de ficheros Linux ext2.

En la particion de disco duro libre denominada ext2, se realiza un proceso de division por bloques de tamaño entorno a 4KB numerandolos secuencialmente. Estos bloques se agrupan en, como minimo, 6 grupos distintos:

- Grupo 1: Bloque 1: Bloque donde se coloca el sector de arranque
- Grupo 2: Bloque 2: Se especifica el tamaño de los restantes bloques
- Grupo 3: Bloque 3 hasta n1: Se especifica el mapeado de los i–nodos
- Grupo 4: Bloque n1+1 hasta n2: Se especifica el mapeado de los datos
- Grupo 5: Bloque n2+1 hasta n3: Espacio de i–nodos
- Grupo 6: Bloque n3+1 hasta n4: Espacio de datos

Los i–nodos almacenan informacion de los datos guardados (tipo de dato nombre del fichero, tamaño que ocupa, etc.) mientras que los datos se almacenan en el espacio de datos (grupo 6).

Tanto los ficheros como los directorios consisten en un i–nodo con la informacion del mismo y uno o varios bloques de datos almacenando los datos.

Una descripcion mas completa del sistema de ficheros Linux ext2 se puede encontrar en el documento `/usr/doc/HOWTO/en-txt/Filesystems-HOWTO.txt`

Para formatear una particion en Linux se utiliza el comando `mkfs(8)`, como en el siguiente ejemplo:

```
# mkfs -t ext2 /dev/hda3
mke2fs 1.20, 25-May-2001 for EXT2 FS 0.5b, 95/08/09
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
244320 inodes, 487974 blocks
24398 blocks (5.00%) reserved for the super user
First data block=0
15 block groups
32768 blocks per group, 32768 fragments per group
16288 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Writing inode tables: done
```

```
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 34 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

Reparando el sistema de ficheros ext2

A diferencia del sistema Windows/dos, en Linux no existe una herramienta propia del sistema de defragmentacion del sistema de ficheros. Realmente por las propias caracteristicas que posee el control del sistema de ficheros extended 2 no sule ser necesario realizar este proceso.

Si existen herramientas de reparaci´on del sistema de ficheros que, incluso, se ejecutan automaticamente cada vez que al arrancar el sistema se detecta que en la ultima sesion no fueron cerrados apropiadamente. El comando para reparar el sistema de ficheros es `fsck` o `e2fsck` en el caso de sistemas de ficheros ext2:

```
# e2fsck /dev/hda3
e2fsck 1.20, 25-May-2001 for EXT2 FS 0.5b, 95/08/09
/dev/hda3: clean, 11/244320 files, 7682/487974 blocks
```

Es importante tener en cuenta que este proceso solo debe llevarse a cabo con la particion no montada o bien montada de solo lectura lo cual plantea un problema con la particion raiz del sistema.

No obstante, lo habitual es que si la particion raiz tiene errores se detecte en el proceso de arranque y se intente correguir automaticamente. Si no es posible la correccion automatica el sistema se queda en modo single–user y solo lectura, pidiendo al administrador del sistema que, antes de arrancar debe entrar y ejecutar manualmente el comando de reparacion del sistema de ficheros. Lo habitual es responder si a todas las preguntas que plantea la correccion de ficheros daÑados. Po esto, es posible automatizar la ejecucion de comando de reparacion para que el solo responda que si a todas las preguntas que se plantea. Para automatizar esta tarea hay que cambiar en el fichero `/etc/default/rcS` la opcion FSCKFIX a yes:

```
#
#       Defaults for the boot scripts in /etc/rcS.d
#

# Time files in /tmp are kept in days.
TMPTIME=0
# Set to yes if you want sulogin to be spawned on bootup
SULOGIN=no
# Set to no if you want to be able to login over telnet/rlogin
# before system startup is complete (as soon as inetd is started)
DELAYLOGIN=yes
# Set UTC=yes if your system clock is set to UTC (GMT), and UTC=no if not.
UTC=yes
```

```
# Set VERBOSE to "no" if you would like a more quiet bootup.
VERBOSE=yes
# Set EDITMOTD to "no" if you don't want /etc/motd to be editted automatica
EDITMOTD=yes
# Set FSCKFIX to "yes" if you want to add "-y" to the fsck at startup.
FSCKFIX=yes
```

Por ultimo, respecto de la reparacion de ficheros es importante tener en cuenta que si hay ficheros o trozos que estan dañados y son corregidos quedan almacenados en el directorio `lost+found/`de la particion correspondiente

Accediendo a la nueva particion

Una vez formateada una nueva particion, por ejemplo, `/dev/hda3`, se puede acceder a ella montándola en algun directorio designado para contenerla:

```
# mount -t ext2 /dev/hda3 /mnt
```

Si se pretende disponer automaticamente de esta particion al arrancar el sistema se necesita añadir una linea al fichero `/etc/fstab`. Si por ejemplo se quisiera montar la partición `/dev/hda3` en el directorio `/home` la línea sería la siguiente:

```
/dev/hda3       /home      ext2    defaults,errors=remount-ro  0  1
```

Distribuyendo el sistema en varias particiones

Como ya mencionamos antes, es interesante tener separados los datos de usuarios de los datos del sistema. Por ello se recomienda dividir el sistema completo en varias particiones.

Como minimo seria aconsejable 2 particiones:

- Particion raiz (`/`): Tamaño aconsejable 1–2GB
- Espacio para los datos de los usuarios (`/home`

En algunos sistemas incluso es conveniente tener mayor numero de particiones que permitan separar diferentes partes del propio sistema:

- `/`: Particion raiz que almacena los comandos básicos(300–500MB).
- `/usr`: Particion del sistema que almacena las aplicaciones instaladas de la propia distribución.
- `/usr/local`: Particion que almacena las aplicaciones que no son de la distribucion.
- `/home`: Espacio para datos de usuarios.
- `/var`: Datos de las aplicaciones.

Creando particiones swap

Las particiones swap son creadas normalmente durante el proceso de instalación. Si por necesidades de las aplicaciones fuera necesario disponer de mayor espacio de swap hay dos alternativas: crear una nueva partición de swap en dico duro libre o, si no se dispone de espacio libre en el disco duro, montar un fichero dentro de una partición activa como swap.

En el primer caso, el proceso comienza por crear una nueva particion con el comando `cfdisk` siguiendo el mismo procedimiento que contamos previamente. Lo unico que hay que tener en cuenta es que el tipo de la nueva particion debe ser *Linux swap*. Para formatear esta particion se utiliza el siguiente comando:

```
# mkswap /dev/hdxx
```

Para activarla:

```
# swapon /dev/hdxx
```

El comando `swapon` con la opcion `-s` da informacion sobre las particiones swap activas.

En el segundo caso, la ventaja esta en que no es necesario disponer de espacio libre en el disco duro, sino de espacio libre en una particion ya activa. El inconveniente es que es mas lento el acceso al fichero swap que a una particion.

El proceso de creacion y activacion de un fichero swap es como sigue:

1. Se crea un fichero relleno de ceros de un tamaÑo adecuado:

   ```
   # dd bs=1024 if=/dev/zero
      of=/dev/swapfile count=1000
   ```

   Con esta orden se crea un fichero */dev/swapfile*de tamaÑo 1024X1000 (aprox. 1M)

2. Se formatea el fichero swap:

   ```
   # mkswap /dev/hdxx 1000
   ```

3. Activar el fichero swap:

   ```
   # sync
   # swapon -v /dev/swapfile
   ```

Una particion o fichero swap puede ser incluido en el fichero `/etc/fstab` para su activacion en el proceso de arranque:

```
# Montando una particion tipo swap
/dev/swapfile       none       swap    sw  0  0
```

3.3.4 Sistemas de ficheros avanzados

Uno de los principales problemas con los sistemas de ficheros, entre ellos el *ext2*, es su poca robustez ante paradas súbitas del sistema, como las que ocurren cuando hay un fallo en el suministro eléctrico. En estos casos, el sistema de ficheros no puede desmontarse a tiempo y un número variable de datos almacenados en la memoria en espera de pasar al disco se perderán. Si estos datos se refieren a la propia organización del disco, el sistema de ficheros puede quedar en un estado *incoherente*, lo que puede provocar la pérdida total de todos los datos almacenado en el sistema de ficheros. Las posibilidades de que ocurra ésto son tanto mayores cuanto mayor sea la actividad del disco en el momento de la caida del sistema. Como puede deducirse, el problema es lo bastante serio como para que nadie haya pensado en una solución. La solución consiste en mantener un *journal* o registro de transacciones realizadas al sistema de ficheros. Esto no evita la pérdida de datos en tránsito, pero permite en la moyoría de los casos, devolver el sistema de ficheros a un estado *coherente* tras una caida del sistema, evitando el problema más grave de la pérdida total de datos del disco.

En la actualidad, hay tres sistemas de ficheros con esta carácterística disponibles para Linux, todo ellos en fase experimental aunque siendo usado en múltiples servidores en todo el mundo con buenos resultados:

- *ext3*: actualización de *ext2* incluyendo un registro de transacciones. Completamente compatible con *ext2*.
- *reiserfs*: sistema de ficheros de propósito general de alto rendimiento con diferentes optimizaciones.
- *xfs*: avanzado sistema de ficheros desarrollado originalmente por SGI para la plataforma IRIX y orientado a proporcionar alto rendimiento y escalabilidad.

Los dos primeros están soportados por las versiones actuales del nucleo de Linux, como las imágnes de la serie 2.4 incluidas con Debian. *ext3* es idéntico a *ext2* salvo por el *journal* y está soportado por las utilidades estándard suministradas con *ext2*. Para manipular sistemas de ficheros *reiserfs* son necesarias las utilidades incluidas en el paquete `reiserfsprogs`. Las utilidades para *xfs* se encuentran en el paquete `xfsprogs`, pero en este caso, es necesario parchear y compilar un nuevo kernel, pues el soporte para este sistema de ficheros aún no se incluye con el kernel oficial de Linux. El parche para el kernel se incluye con Debian en el paquete `kernel-patch-xfs`, que puede aplicarse al fuente del kernel incluido en el paquete `kernel-source-2.4.18`. Para más información sobre como compilar un kernel consultar el capítulo ‘Instalando nuevos modulos y compilando un nuevo *Kernel*’ en la página 61.

La preparación de sistemas de ficheros de un formato diferente a *ext2* no tiene ninguna dificultad, y sigue los mismos pasos explicados en apartados anteriores. Basta tener en cuenta los siguientes puntos:

- Es necesario tener soporte en el kernel para el sistema de ficheros a usar. Los sistemas de ficheros soportados se listan en `/proc/filesystems`, salvo para aquellos compilados como módulos y que no esten cargados en ese momento. Si los módulos correspondientes están disponibles, estos serán cargados automáticamente o pueden añadirse con el programa `modconf`.
- Si el sistema de ficheros se va a usar como raiz, el soporte ha de estar compilado en el kernel, no vale que esté como módulo.

- Para manipular el sistema de ficheros hay que instalar los programas correspondientes, mencionados arriba.
- Los comandos para manipular los distintos sistemas de ficheros son análogos, por ejemplo, para *reiserfs* son `mkfs.reiserfs`, `fsck.reiserfs`, etc.
- `/etc/fstab` debe actualizarse en caso de cambiar el tipo del sistema de ficheros.

Actualización de sistemas *ext2* a *ext3*

Como *ext3* es sólo una actualización a *ext2*, los sistemas de ficheros *ext2* pueden convertirse de forma sencilla a *ext3* de forma muy sencilla, sin tener que formatear particiones ni copiar discos. Supongamos un sistema con las siguientes particiones de Linux, según figura en `/etc/fstab`:

```
...
/dev/hdb1  /      ext2 defaults,errors=remount-ro   0      1
/dev/hdb3  /home  ext2 defaults,errors=remount-ro   0      1
...
```

Para convertirlas a *ext3*, añadimos primero un registro de transacciones a los sistemas de ficheros. No hace falta desmontar los sistemas de ficheros ni nada:

```
# tune2fs -j /dev/hdb1
# tune2fs -j /dev/hdb3
```

Luego alteramos `/etc/fstab` para reflejar el cambio en el sistema de ficheros:

```
...
/dev/hdb1  /      ext3 defaults,errors=remount-ro   0      1
/dev/hdb3  /home  ext3 defaults,errors=remount-ro   0      1
...
```

Listo. La próxima vez que arranquemos el sistema, los sistemas de ficheros funcionarán como *ext3*. Podemos comprobarlo en los mensajes de arranque del kernel:

```
# dmesg | less
...
kjournald starting.  Commit interval 5 seconds
EXT3-fs: mounted filesystem with ordered data mode.
VFS: Mounted root (ext3 filesystem) readonly.
...
```

3.4 Manipulación de diskettes

Hemos visto como se pueden usar diskettes formateados montándolos en el sistema de ficheros. En este apartado veremos otras formas de acceder a los diskettes sin necesidad de montarlos, como formatear diskettes y como manipular imágenes de diskettes.

3.4.1 Manipular diskettes como en MS–DOS: mtools

Cuando se desea copiar datos de forma rápida a/desde un diskette, la opción de montar el dispositivo puede ser un poco engorrosa e incluso arriesgada si olvidamos desmontar el dispositivo antes de sacar un diskette. Para estas ocasiones, existe un paquete de programas que simulan los comando empleados por MS–DOS para acceder a los diskettes, llamado `mtools`. Cuando este paquete está instalado, se dispone de los mismos comando que existen en MS–DOS, con la salvedad de que ahora sus nombre comienzan con `m`, por ejemplo: `mdir`, `mcopy`, `mdel`, `mcd`, `mmd`, `mrd`, etc. Hay información general sobre mtools en `mtools(3)` y sobre cada programa en la página de manual correspondiente.

Algunas consideraciones generales sobre el uso de estos comandos son:

- A la unidad de diskette se hace referencia con `a:` como en MS–DOS.
- Muchos comandos como `mdir` y `mcd` operan por defecto sobre `a:`
- En el comando `mcopy`, las rutas que comienzan con `a:` se refieren al diskette, las que no, a directorios UNIX
- Cuando se usan comodines (`*` o `?`) haciendo referencia a una unidad, es necesario "escaparlos" del shell usándolos, por ejemplo, dentro de comillas: `"a:*.*"`

Algunos ejemplos de uso de las *mtools* son:

```
$ mdir
```

muestra el contenido del directorio actual en el diskette.

```
$ mcopy a:prog.tar.gz .
```

copia el fichero `prog.tar.gz` desde el diskette al directorio actual en UNIX.

```
$ mdel 'a:*.txt'
```

borra todos los ficheros con extensión `.txt` de la unidad `a:`. Observar el uso de las comillas simples para evitar que el shell interprete el comodín.

```
$ mcd prog
```

cambia el directorio actual en el diskette a `prog`.

```
$ mcopy *.c a:
```

copia todos los ficheros con extensión `.c` al directorio actual del diskette.

Finalmente, si quisiéramos configurar las mtools para, por ejemplo, añadir nuevas unidades además de la disketera (una unidad ZIP, una partición MS–DOS en el disco duro, etc.) tendríamos que editar el fichero de configuración `/etc/mtools.conf`. El mismo fichero trae ejemplos de como configurar otras unidades. Para más información consultar las páginas de manual o info de mtools.

3.4.2 Formatear diskettes

Las *mtools* incluyen un comando para dar formato rápido a diskettes, esto es, si el disco ya estaba formateado, borra todo su contenido y crea un nuevo sistema de ficheros vacío tipo FAT. Para formatear realmente un diskette (formateo físico) se emplea `superformat`. Por ejemplo:

```
$ superformat /dev/fd0
```

formatea un diskette en la primera unidad y le añade un sistema de ficheros MS–DOS usando `mformat`.

La primera vez que se ejecute `superformat` hará un proceso de calibración de la unidad de diskette e informará sobre el resultado obtenido por si se quiere almacenar para el futuro en el fichero `/etc/driveprm`. Si este fichero no existe aún, puede crearse con la información necesaria con este comando, que hay que ejecutar como root:

```
# echo "drive0: deviation=6240" /etc/driveprm
```

donde la expresión entre comillas debe sustituirse por la línea exacta generada por `superformat` como resultado de la calibración.

En el ejemplo anterior se ha usado `/dev/fd0` como el dispositivo que hace referencia a la primera unidad de diskette. En realidad hay otros dispositivos que hacen referencia a esta unidad y que especifican otras capacidades diferentes de la habitual de 1440KB. Una lista de estos dispositivos puede obtenerse con:

```
$ ls -l /dev/fd0*
```

El número final en el nombre del dispositivo hace referencia a la capacidad del formato que representa. Así, por ejemplo, para formatear un diskette a una capacidad de 1920KB (casi 2MB) haríamos:

```
$ superformat /dev/fd0u1920
```

Este formato especial será reconocido de forma automática tanto por las mtools como por el comando `mount`

A pesar de las indudables ventajas que tienen los formatos de capacidades mayores a la estándar, hay que tener en cuenta que estos formatos pueden no ser reconocidos por otros sistemas operativos o en máquinas de otras arquitecturas.

3.4.3 Manipulación de diskettes a bajo nivel

El acceso a los diskettes a través de dispositivos como `/dev/fd0` permite un acceso a bajo nivel. Esto es, al igual que ocurre con la mayoría de los dispositivos en `/dev`, podemos escribir y leer directamente de ellos (siempre que tengamos los permisos adecuados). Por ejemplo, si escribimos un bloque de 512 datos en `/dev/fd0`, estaremos escribiendo el primer sector físico del diskette, etc. Esto nos permite extraer literalmente el contenido de diskettes, sector a sector, o grabar en ellos imágenes literales almacenadas en ficheros, como discos de arranque de Linux. A continuación veremos algunas aplicaciones de esto. Aunque aquí se muestran ejemplos para diskettes, la misma idea se aplica a otros dispositivos tipo disco, cinta, etc.

Una pregunta frecuente en grupos de usuarios novatos de Linux es *¿hay un comando diskcopy como en MS–DOS?*. La respuesta es *NO*, o mejor dicho, *no hace falta*. Para crear una copia literal de un diskette, primero se vuelca el contenido en un fichero (creamos la imagen) y luego éste se vuelca en otro diskette. Por ejemplo:

```
$ cat /dev/fd0 > /tmp/imagen
```

vuelca la imagen del diskette en el fichero `/tmp/imagen`. Luego insertamos el diskette de destino y hacemos:

```
$ cat /tmp/imagen > /dev/fd0
```

y ya está hecha la copia. Si no queremos hacer más copias podemos borrar el fichero `/tmp/imagen`.

Una alternativa a usar el comando `cat`, y que puede ser más eficiente, es usar el programa `dd`. `dd` es un programa genérico para transferir datos entre ficheros o dispositivos con la posibilidad de hacer ciertas traducciones de formato (ver `dd(1)`). En nuestro caso, usaríamos:

```
$ dd if=/dev/fd0 of=/tmp/imagen bs=512
```

para crear la imagen, y

```
$ dd if=/tmp/imagen of=/dev/fd0 bs=512
```

para grabarla, donde `if` e `of` indican los ficheros de entrada y salida respectivamente y la opción `bs=512` indica transferir en bloques de 512 bytes, que es el tamaño del sector y puede optimizar la transferencia.

Si la imagen ya está creada y simplemente queremos grabarla en un diskette, como es el caso de los discos de arranque de Debian que vienen en el CD de la distribución, basta con ejecutar el comando de escritura. Por ejemplo, para crear un disco de rescate a partir de la imagen que viene en el CD de Debian, primero montamos el CD número 1 en `/cdrom`:

```
$ mount /cdrom/
```

luego entramos en el directorio con las imágenes:

```
$ cd /cdrom/dists/stable/main/disks-i386/current/images-1.44
```

y finalmente introducimos un diskette y copiamos la imagen (bien con `cat` o con `dd`:

```
$ cat rescue > /dev/fd0
```

o bien

```
$ dd if=rescue of=/dev/fd0 bs=512
```

Capítulo 4

Gestión de usuarios y grupos

Aquí se presentan los aspectos básicos para la gestión de usuarios y grupos de usuarios en el sistema.

4.1 Añadir usuarios y grupos

Para añadir usuarios y grupos al sistema se emplean los comandos `adduser` y `addgroup`. La operación de estos comandos se configura en el fichero `/etc/adduser.conf`. Veamos algunos ejemplos:

```
# adduser pepe
```

añade el usuario *pepe* al sistema. El sistema pedirá alguna información adicional sobre el usuario y un *password* o clave. Por defecto, se crea un grupo con el nombre del usuario y éste será el grupo por defecto. Este comportamiento se configura en `/etc/adduser.conf`.

```
# adduser --ingroup users pepe
```

añade el usuario *pepe* al sistema estableciendo *users* como su grupo principal:

```
# adduser pepe cdrom
```

añade el usuario pepe (previamente creado) al grupo *cdrom*.

Cuando el número de usuarios es numeroso y heterogéneo, puede ser necesario añadir nuevos grupos. Esto se hace con el comando `addgroup`. Por ejemplo:

```
# addgroup alumnos
```

añade al sistema un grupo llamado *alumnos*.

Alternativamente a los comandos anteriores, se pueden añadir usuarios y grupos empleando `useradd` y `groupadd`. Estos comandos leen información de configuración del fichero `/etc/login.defs`.

4.2 Eliminar usuarios y grupos

Para eliminar usuarios y grupos se emplean `userdel` y `groupdel` respectivamente. Por ejemplo:

```
# userdel pepe
```

elimina el usuario pepe. Si además se indica la opción `-r`, también se borrará el directorio personal del usuario con todo su contenido.

```
# groupdel alumnos
```

elimina el grupo *alumnos*.

4.3 Modificar usuarios y grupos

Para modificar las características de los usuarios y grupos se emplean los comandos `usermod` y `groupmod`. Algunos ejemplos:

```
# usermod -d /home/profes/pepe -m
```

cambia el directorio de inicio del usuario `pepe` para que sea `/home/profes/pepe`. La opción `-m` hace que mueva el contenido del antiguo directorio al nuevo emplazamiento.

```
# usermod -g profes pepe
```

cambia el grupo inicial del usuario `pepe` para que sea `profes`.

```
# usermod -l joseg pepe
```

cambia el nombre del usuario `pepe`. El nuevo nombre es `joseg`.

```
# groupmod -n profesores profes
```

cambia el nombre del grupo `profes` a `profesores`.

4.4 Ficheros relacionados con la gestión de usuarios y grupos

Algunos ficheros relacionados con las cuentas de usuario son:

- `/etc/passwd`: contiene información sobre cada usuario: ID, grupo principal, descripción, directorio de inicio, shell, etc. También contiene el *password* encriptado, salvo que se usen *shadow passwords*.
- `/etc/shadow`: contiene los *passwords* encriptados de los usuarios cuando se emplean *shadow passwords*.
- `/etc/group`: contiene los miembros de cada grupo, excepto para el grupo principal, que aparece en `/etc/passwd`.
- `/etc/skel`: directorio que contiene el contenido del directorio de los nuevos usuarios.

4.5 Algunos grupos especiales

En el sistema existen algunos grupos especiales que sirven para controlar el acceso de los usuarios a distintos dispositivos. El control se consigue mediante los permisos adecuados a ficheros de dispositivo situados en `/dev`. Algunos de estos grupos son:

- **cdrom**: dispositivos de CD–ROM. El dispositivo concreto afectado depende de donde estén conectadas las unidades de CD–ROM. Por ejemplo, `/dev/hdc`.
- **floppy**: unidades de diskette, por ejemplo, `/dev/fd0`
- **dialout**: puertos serie. Afecta, por ejemplo, a los modems externos conectados al sistema. Por ejemplo, `/dev/ttyS1`
- **audio**: controla el acceso a dispositivos relacionados con la tarjeta de sonido. Por ejemplo, `/dev/dsp`, `/dev/mixer` y `/dev/sndstat`.

Para dar acceso a un usuario a uno de estos servicios, basta con añadirlo al grupo adecuado. Por ejemplo, para dar acceso al usuario pepe a la disketera haríamos:

```
# adduser pepe floppy
```

Alternativamente, para sistemas pequeños suele ser mejor “desproteger” los dispositivos adecuados para que todos los usuarios puedan usarlos, evitando tener que recordar añadir usuarios a los grupos adecuados. Por ejemplo, para dar acceso de lectura al CD–ROM (suponiendo que esté en `/dev/hdc`) y de lectura/escritura a la disketera a todos los usuarios, haríamos:

```
# chmod a+r /dev/hdc
# chmod a+rw /dev/fd0*
```

4.6 Límites a los usuarios

En los sitemas UNIX/LINUX existe la posibilidad de limitar recursos a los usuarios o grupos, por ejemplo, el máximo numero de logins que puede realizar simultaneamente un usuario, el maximo tiempo de CPU, el maximo numero de procesos etc.

Estos límites se controlan en LINUX a traves del fichero `/etc/security/limits.conf`. Concretamente, en este fichero se controlan los limites sobre los procesos de un usuario:

```
# /etc/security/limits.conf
#
#Each line describes a limit for a user in the form:
#
#domain        type  item  value
#
#Where:
#domain can be:
#        - an user name
#        - a group name, with @group syntax
#        - the wildcard *, for default entry
#
#type can have the two values:
#        - "soft" for enforcing the soft limits
#        - "hard" for enforcing hard limits
#
#item can be one of the following:
#        - core - limits the core file size (KB)
#        - data - max data size (KB)
#        - fsize - maximum filesize (KB)
#        - memlock - max locked-in-memory address space (KB)
#        - nofile - max number of open files
#        - rss - max resident set size (KB)
#        - stack - max stack size (KB)
#        - cpu - max CPU time (MIN)
#        - nproc - max number of processes
#        - as - address space limit
#        - maxlogins - max number of logins for this user
#        - priority - the priority to run user process with
#
# dominio       tipo  item            value
#

#*               soft    core            0
#*               hard    rss             10000
#@student        hard    nproc           20
#@faculty        soft    nproc           20
```

```
#@faculty        hard    nproc           50
#ftp             hard    nproc           0
#@student        -       maxlogins       4

# End of file
```

Tambien es posible limitar los tiempos de acceso a los usuarios. Una de las formas de hacerlo en Debian 2.2 potato es con el servicio *timeoutd*. Este servicio se instala a traves de la distribucion y, una vez instalado aparece un fichero de configuracion `/etc/timeouts`.

En este fichero de configuracion las lineas en blanco o que cominezan por # no son interpretadas. El resto de las lineas debe tener alguna de las dos siguientes sintaxis:

```
TIMES:TTYS:USERS:GROUPS:MAXIDLE:MAXSESS:MAXDAY:WARN
```

o bien

```
TIMES:TTYS:USERS:GROUPS:LOGINSTATUS
```

En la pagina de manual `timeouts(8)` se explica el significado de cada uno de los campos en estas lineas.

4.6.1 Ejemplos de limites horarios

- El ususario curso no puede hacer login durante el fin de semana:

  ```
  SaSu:*:curso:*:NOLOGIN
  ```

- Solo el ususario root puede acceder desde las consolas tty1–tty6:

  ```
  Al:tty1,tty2,tty3,tty4,tty5,tty6:root:*:LOGIN
  Al:tty1,tty2,tty3,tty4,tty5,tty6:*:*:NOLOGIN
  ```

- Solo el usuario root puede acceder entre las 15:00 y las 16:00h de cada dia:

  ```
  Al1500-1600:*:root:*:LOGIN
  Al1500-1600:*:*:*:NOLOGIN
  ```

Una vez que se ha preparado el fichero `/etc/timeouts` es necesario reiniciar el servidor timeoutd:

```
# /etc/init.d/timeoutd restart
Stopped /usr/sbin/timeoutd (pid 2412).
Starting /usr/sbin/timeoutd...
```

Es importante destacar que este proceso no actua durante el proceso de login lo que da lugar a que, aunque un usuario tenga prohibido el acceso a una maquina en un momento determinado, inicialmente puede entrar y solo, una vez que se ejecute el proceso timeoutd, sera expulsado del sistema.

Para conseguir que durante el proceso de login se revisen las condiciones de timeouts se debe incluir las siguientes lineas en el fichero `/etc/profile`:

```
# Comprueba restricciones de timeoutd (ver timeoutd, timeouts(5))
/usr/sbin/timeoutd ‘whoami‘ ‘basename \‘tty\‘‘ || exit
```

Con esta linea incluso aunque el servicio *timeoutd* este parado si en el fichero `/etc/timeouts`se prohibe e acceso a un usuario este no podra entrar en el sistema.

4.7 Limites de quotas

El sistema de quotas provee un mecanismo de control y uso del espacio de disco duro disponible en un sistema. Se pueden establecer limites en la cantidad de espacio y el numero de ficheros de que puede disponer un usuario o grupo.

En las quotas hay cuatro numeros para cada limite: la cantidad actual ocupada; el limite *soft* (quota propiamente dicha); el limite *hard* (espaqcio sobre quota); y el tiempo que resta antes de eliminar el exceso entre *soft* y *hard*. Mientras que el limite *soft* puede ser superado temporalmente, el limite *hard* nunca puede rebasarse.

4.7.1 Administrando el sistema de quotas

Para implementar el sistema de quotas es necesario instalar algun paquete de control de dicho sistema. En Debian hay un paquete denominado `quota` que instala todo lo necesario para implementar todo el sistema. Una vez instalado tenemos que realizar una serie de pasos para activar el mecanismo de quotas. Estos pasos son:

- Configuracion de kernel
- Eleccion del sistema de ficheros sobre el que se aplican las quotas
- Habilitar las quotas
- Especificar quotas para usuarios o grupos
- Desabilitar quotas para usuarios o grupos

Configuracion del kernel

Antes de instalar el sistema de quotas debe disponerse de un kernel con la opcion de *quota–system* habilitada. Esto se consigue en el proceso de compilacion de un nuevo kernel respondiendo *yes*a la pregunta de *Disk QUOTA support* Los kernels precompilados que se distribuyen con Debian (paquetes `kernel-image`.ya tienen esta opción habilitada.

Eleccion del sistema de ficheros

Una vez dispuesto el kernel, hay que seleccionar que sistema de ficheros necesitan tener aplicadas las quotas. Lo normal es que solo el sistema donde estan las cuentas de usuarios tengan quotas, aunque es recomendable que tenga quotas todo sistema de ficheros donde los usuarios puedan escribir.

Para habilitar las quotas en un sistema de ficheros hay que editar el fichero `/etc/fstab` y incluir las opciones usrquota y grpquota:

```
# /etc/fstab: static file system information.
#
# file system  mount point  type  options                          dump pas
/dev/hda5  /        ext2    defaults,errors=remount-ro,usrquota,grpquota  0
/dev/hda6  none     swap    sw                                             0
proc       /proc    proc    defaults                                       0
/dev/fd0   /floppy  auto    defaults,user,noauto                           0
/dev/cdrom /cdrom   iso9660 defaults,ro,user,noauto                        0
```

Habilitando las quotas

Para instalar los ficheros de quotas se debe ejecutar el comando:

```
# quotacheck -avug
Scanning /dev/hda5 [/] done
Checked 4943 directories and 57624 files
Using quotafile /quota.user
Updating in-core user quotas
Using quotafile /quota.group
Updating in-core group quotas
```

La primera vez que se ejecuta este comando sirve para crear los ficheros de quotas: `quota.user` y `quota.group`.

Especificando la quota a un usuario o grupo

Para editar la quota de un usuario o grupo se usa el programa `edquota` con la opcion `-u`para editar las quotas de usuarios y con la opcion `-g`para editar las opciones de grupo.

Solo hay que editar los numeros que estan detras de soft y hard. El periodo de gracia que hay entre el limite soft y el hard puede cambiarse con:

```
# edquota -t
```

La mayoria de las veces los usuarios tienen la misma quota. Una forma rapida de editar la quota de todos los usuarios es colocarse en el directorio donde tienen sus directorios raiz cada usuario. Editar la quota de uno de estos usuarios con los valores apropiados y, posteriormente, ejecutar:

```
# edquota -p usuarioprototipo *
```

Para verificar las quotas que tiene un usuario se utiliza el comando:

```
$ quota -v
```

El superusuario puede ver las quotas de todos los usuarios con el comando:

```
# repquota filesystem
```

Deshabilitando quotas para un usuario o grupo

Para deshabilitar las quotas de un usuario o grupo solo hay que editarlas quotas y poner los limites a 0. Asi un usuario puede usar tantos bloques e inodos como quiera.

Capítulo 5

Manipulación de paquetes Debian

Debian tiene un formato propio de paquetes que se reconocen por la extensión `.deb`. Este formato es equivalente al *rpm* usado por otras distribuciones como RedHat, pero el formato de Debian es más sofisticado y tiene muchas más posibilidades. A este formato y a las herramientas que lo manipulan debe Debian su gran estabilidad y facilidad de instalación y actualización de paquetes.

En Debian existen básicamente dos tipos de herramientas para manipular paquetes: unas de bajo nivel que manipulan paquetes individuales y otras de más alto nivel que manipulan paquetes incluídos en una base de datos previamente creada. Estas últimas son las más convenientes, ya que resuelven, a menudo de forma automática, las posibles dependencias y conflictos entre paquetes, haciendo más fácil su instalación y gestión. En esta sección describiremos cómo emplear estas herramientas para realizar las tareas más comunes relacionadas con los paquetes.

5.1 Manipulación de paquetes a alto nivel

Existen diferentes aplicaciones para manipular paquetes a alto nivel como `dselect`, `console-apt` y `gnome-apt`. En la actualidad, todas ellas se basan en el programa `apt` y actúan como interfaz de usuario para el mismo (`dselect` puede usar otros métodos aparte de `apt`, pero estos han quedado obsoletos). Tanto `dselect` como `console-apt` funcionan en modo texto, mientras que `gnome-apt` funciona en X–Window. Los dos últimos se encuentran en fase de desarrollo pero se pueden usar sin demasiados problemas. Aquí sólo daremos algunas notas sobre el uso de `dselect`, ya que en el primer CD de instalación (y en la página web de Debian) se proporciona un manual para principiantes en castellano. En cuanto a los otros programas, esperamos que el lector no tenga dificultad en instalar los paquetes correspondientes y encontrar la documentación.

5.1.1 Dselect

Los puntos más importantes a recordar sobre `dselect` son:

- El método de acceso está por defecto en modo APT y no debe cambiarse.

- Es necesario ejecutar la opción Actualizar/Update cada vez que cambie el contenido del fichero `/etc/apt/sources.list` o se ejecute el comando `apt-cdrom` (ver 'APT' en esta página) incluso si se ha ejecutado el comando

```
# apt-get update
```

- La opción Selección/Select permite ver la lista de paquetes instalados y disponibles. Para salir de la pantalla de información inicial pulsar la barra espaciadora.
- Se pueden buscar paquetes por nombre en la pantalla de selección usando la tecla "/"y se puede repetir la búsqueda usando "\". Esta búsqueda se realiza sólo en los nombres y no en la descripción de los paquetes, lo cual es una importante deficiencia de `dselect`.
- Algunas teclas útiles cuando se presentan conflictos y nos entre el pánico son:
  - `U`: vuelve a las selecciones hechas por `dselect`
  - `R`: cancela las selecciones hechas por nosotros en la pantalla actual
  - `X`: descarta todos los cambios en la sesión actual y sale de `dselect`

5.1.2 APT

Configuración de fuentes para `apt`

Lo primero que hay que hacer para usar `apt` y los programas que dependen de él es indicarle dónde debe buscar paquetes Debian para crear su base de datos. Las fuentes de paquetes se indican en el fichero `/etc/apt/sources.list` y son básicamente de dos tipos: URL's (*Localizador Universal de Fuentes*) y CD–ROM's. Los primeros incluyen direcciones *http, ftp* y directorios locales, mientras que los segundos se refieren a unidades de CD extraibles, como los CD's de instalación de Debian.

Las fuentes tipo URL se indican en `/etc/apt/sources.list` con líneas de este tipo:

```
     deb ftp://ceu.fi.udc.es/debian stable main contrib non-free
     deb http://non-us.debian.org/debian-non-US stable/non-US main con-
trib non-free
     deb http://security.debian.org stable/updates main contrib non-free
```

En cada línea se indica la localización del directorio principal de Debian, la versión de la distribución (stable, unstable, etc.) y las secciones que se quieren tomar.

Las fuentes tipo CD–ROM no se crean manualmente, sino que son introducidas en `/etc/apt/sources.list` ejecutando el comando

```
# apt-cdrom add
```

Entonces el sistema pedirá que introduzcamos un CD–ROM en la unidad, lo analizará en busca de paquetes Debian y actualizará tanto la información en el fichero de fuentes como otra información interna para recordar en el futuro en que CD–ROM se encuentra cada paquete

Se puede encontrar información más detallada sobre el formato del fichero de fuentes y `apt-cdrom` en `sources.list(5)` y `apt-cdrom(8)`. En cualquier caso, después de cambiar el contenido del fichero de fuentes se debe actualizar la base de datos de APT ejecutando:

```
# apt-get update
```

Si se emplea `dselect`> también se debe elegir la opción de Actualizar/Update la próxima vez que se ejecute.

Instalación y desinstalación de paquetes

La instalación y desinstalación de paquetes con APT es extraordinariamente sencilla si se conoce el nombre del paquete a instalar. Basta con ejecutar el comando `apt-get` con el argumento `install` y la lista de paquetes a instalar. APT instalará los paquetes seleccionados y cualquier otro del que éstos dependan, resolviendo todas las dependencias. Para la instalación considerará todas las fuentes listadas en `/etc/apt/sources.list` y obtendrá las versiones más actualizadas de los paquetes, realizando las conexiones necesarias y solicitando los CD's que se necesiten. En caso de encontrar un paquete en varias fuentes, utilizará aquella que se especifique en primer lugar en el fichero de fuentes. Por ejemplo,

```
# apt-get install mozilla gimp gnome-xbill
```

instalará los paquetes `mozilla`, `gimp` y `gnome-xbill` más todos aquellos que éstos necesiten.

Para desinstalar paquetes se emplea la opción `remove` de `apt-get`. Por defecto se conservan los ficheros de configuración del paquete que se desinstala, pero se puede indicar que se borren con la opción `--purge`. Por ejemplo,

```
# apt-get remove mozilla gimp gnome-xbill
```

desinstala los paquetes anteriores, y

```
# apt-get --purge remove mozilla gimp gnome-xbill
```

borraría además los ficheros de configuración.

Actualizar la distribución

El sistema APT incluye potentes algoritmos que permiten actualizar una gran cantidad de paquetes simultáneamente e incluso renovar la distribución completamente. Por ejemplo, si en el servidor web o ftp han actualizado los paquetes de nuestra versión, podemos obtener las listas actualizadas con:

```
# apt-get update
```

y luego actualizar a las nuevas versiones de todos los paquetes que hayan sido actualizados con:

```
# apt-get upgrade
```

Si la actualización supone un cambio a una versión completamente nueva de la distribución (porque aparezca una nueva versión mayor o porque queramos cambiar a una de las versiones de desarrollo) necesitamos usar otra opción que es capaz de manejar la actualización simultánea de todos los paquetes de la distribución:

```
# apt-get dist-upgrade
```

Otras opciones de `apt-get`

Algunas opciones útiles de `apt-get` son:

- `clean` y `autoclean`: borran del disco local ficheros `.deb` descargados pero que no se han llegado a instalar.
- `-f (--fix-broken)`: trata de resolver conflictos de dependencias en la instalación de paquetes.
- `-s (--simulate)`: muestra las operaciones que haría pero no hace cambios en realidad.

Obtención de información sobre paquetes

La utilidad `apt-cache(8)` permite obtener información diversa sobre los paquetes. Aquí se muestran las opciones más útiles:

```
# apt-cache stats
```

muestra estadísticas generales sobre el sistema de paquetes: paquetes instalados, disponibles, etc.

```
# apt-cache show paquete
```

muestra información general sobre `paquete`, incluyendo una descripción.

```
# apt-cache search regexp
```

busca paquetes con la expresión regular `regexp` en su descripción, opción de la que carece `dselect`. Muy útil para buscar un paquete sobre un tema determinado. Por ejemplo:

```
$ apt-cache search chess
xshogi - An X Window System Japanese Chess (Shogi) Board.
xboard - An X Window System Chess Board.
gnome-chess - GNOME Chess
...
```

5.2 Reconfiguración de paquetes instalados

Cada vez que un paquete es instalado, el sistema pasa automáticamente a la fase de configuración del mismo. En muchas ocasiones la configuración se realiza sin la intervención del usuario, pero otras veces ésta es necesaria para que el usuario eliga ciertas preferencias sobre la operación del programa, como en el caso de los paquetes `exim` o `ssh`. Si en el futuro queremos modificar las opciones de instalación de uno de los paqutes instalados, pueden darse tres situaciones:

1. No hay programa de reconfiguración: en este caso se deben editar los ficheros apropiados y modificar allí las opciones deseadas. Es el caso, por ejemplo, de la configuración de los parámetros de la red.
2. El paquete posee un programa de configuración propio: en este caso basta con ejecutar dicho programa. Por ejemplo, `eximconf` configura la instalación inicial del paquete `exim`.
3. El paquete emplea el sistema *debconf* para su configuración inicial. El sistema *debconf* es la interfaz genérica de Debian para configurar paquetes durante su intalación inicial. Se tiende a que todos los paquetes de la distribución empleen este sistema. Si el paquete emplea este sistema, se puede reconfigurar con el el comando `dpkg-reconfigure`, que presentará por defecto los parámetros de configuración empleados la última vez que se configuró el paquete. Por ejemplo, para reconfigurar el paquete `sshd`:

   ```
   # dpkg-reconfigure sshd
   ```

   El propio sistema *debconf* puede configurarse con este método. Entre otras facilidades, *debconf* proporciona una variedad de interfases de configuración y la posibilidad de configurar los paquetes en bloque antes de su descarga e instalación.

En cualquier caso, se recuerda que esta configuración se refiere a las opciones de instalación inicial de los paquetes, y no a programas generales de configuración y administración del software. Estos programas, si existen, dependen de cada paquete en concreto y tendremos que consultar la documentación del paquete para averiguar si existen y cómo se usan.

5.3 Manipulación de paquetes a bajo nivel

La manipulación directa de ficheros de paquetes se realiza con el comando `dpkg(8)`. Gracias a la existencia de programas como `dselect` y `apt-get`, sólo es necesario usar `dpkg` cuando nos encontramos con paquetes Debian "sueltos", esto es, que no forman parte de un archivo estructurado en la red o de un CD–ROM con el formato adecuando para `apt-cdrom`.

El comando

```
# dpkg --install paquete.deb
```

instalará el fichero de paquete indicado. En caso de que se produzcan problemas de dependencias (falta un paquete del que depende), el paquete no será configurado, pero será desempaquetado. En este caso deberemos instalar también los paquetes necesarios para resolver las dependencias (con `dselect` o `apt-get` si es posible o bien con `dpkg` y finalmente ejecutar

```
# dpkg --configure --pending
```

para configurar todos los paquetes pendientes de configuración.

Es necesario advertir que se debe ser cuidadoso con los paquetes instalados directamente con `dpkg`. Por ejemplo, si tenemos instalada una versión "stable" de la distribución y tomamos paquetes de la versión "unstable", casi con toda seguridad tendremos que instalar versiones nuevas de las que depende este paquete. El número de dependencias implicadas puede llegar a ser tan alto que implique a la práctica totalidad de los paquetes instalados o a paquetes "vitales" para el sistema cuya instalación "manual" es, cuando menos, delicada si no se sabe bien lo que se hace. Una opción más adecuada para tomar paquetes de la distribución inestable es descargar los fuentes de los mismos y compilarlos en nuestro sistema. Esto se realiza con la opción `source` de `apt-get` y con el comando `dpkg-buildpackage`. El procedimiento se explica en el apartado siguiente.

Para desinstalar paquetes con `dpkg` se emplea

```
# dpkg --remove paquete
```

o bien

```
# dpkg --purge paquete
```

si se quiere además borrar los ficheros de configuración. En ambos casos se indica el nombre del paquete, no del fichero `.deb`.

5.4 Obtención y compilación de paquetes fuente Debian

También es posible descargar el código fuente de los paquetes Debian. Para ello debemos tener configurado el fichero `sources.list` con las líneas `deb-src` adecuadas. Por ejemplo:

```
deb-src ftp://ftp.uk.debian.org/debian unstable main contrib non-
free
```

Para descargar paquetes fuente se emplea la opción `source` de `apt`, siendo conveniente disponer de un directorio específico para realizar las descargas, como `/usr/local/deb`. Por ejemplo:

```
# cd /usr/local/deb-src
# apt-get source gtoaster
```

La principal utilidad que tiene el descargar paquetes fuente para el usuario convencional, es la posibilidad de compilar en un sistema “stable"paquetes de la rama ünstable”. Como se comentó en el apartado anterior, la instalación directa de paquetes binarios de la rama “unstable"en la "stable” provoca múltiples conflictos de dependencias que prácticamente pueden obligar a la actualización completa de la instalación a “unstable”, pero esto no suele ser deseable en sistemas en producción. Muchas de estas dependencias se resuelven si obtenemos el paquete de la rama “unstable” en formato fuente y lo compilamos dentro de la “stable” usando las librerías presentes en las mismas. No obstante, es posible que programas de la rama “unstable” necesiten versiones actualizadas de las librerías u otros programas. Este hecho es más probable cuanto más espacio temporal separe las ediciones estable e inestable de la distribución. En este caso es necesario obtener los paquetes fuente de las librerías o programas a actualizar y realizar el mismo proceso con éstos.

A continuación se enumeran los pasos a seguir para obtener y compilar un paquete fuente. Se supone que trabajamos con una versión estable y que queremos obtener y compilar paquetes de la rama inestable.

1. Configuramos `/etc/apt/sources.list` con las direcciones donde encontrar paquetes fuente:

   ```
   deb-src ftp://ftp.uk.debian.org/debian unstable main contrib non-
   free
   deb-src ftp://ftp.uk.debian.org/debian/non-US unstable/non-US main con-
   trib non-free
   ```

2. Cambiamos al directorio adecuado y descargamos el código fuente del paquete a compilar:

   ```
   # cd /usr/local/deb-src
   # apt-get source paquete.deb
   ```

   Este proceso descarga tres ficheros por cada paquete:

   - `paquete.orig.tar.gz`: código fuente original del programa, antes de ser modificado por el equipo de Debian.

- `paquete.diff.gz`: modificaciones realizadas por el equipo de Debian al código original.
- `paquete.dsc`: información administrativa sobre el paquete.

Además, se habrá creado un directorio con el nombre del paquete donde se ha descomprimido el código y se han aplicado las modificaciones realizadas por Debian.

3. Para compilar cualquier programa se necesita tener instaladas librerías de desarrollo. Las librerías necesarias para compilar el paquete se mencionan el el fichero `paquete.dsc` o en el fichero `paquete/debian/control` en la línea encabezada con *Build–Depends:*

   Si las versiones de estos paquetes están disponibles en la versión estable de la distribución, basta con instalarlos directamente con `apt-get install ...` En caso de que no dispongamos de alguna de las librerías o fuera necesaria una versión posterior, tendríamos que obtener el paquete fuente de esa librería y repetir el mismo proceso que estamos describiendo.

4. Compilamos el código fuente. Para ello entramos en el directorio del paquete y ejecutamos el comando `dpkg-buildpackage`:

   ```
   # cd paquete
   # dpkg-buildpackage
   ```

   Aquí debemos estar atentos a los posibles errores que en la mayoría de los casos se deben a la falta de algún programa de desarrollo (compilador, etc.) o de alguna de las librerías mencionadas anteriormente. Tras resolver el problema se vuelve a ejecutar el comando hasta que el proceso finalice correctamente.

   El paquete binario generado tiene extensión `.deb` y se escribe en el directorio superior, junto al fichero `.dsc`.

5. Instalar el paquete generado. Ésto se hace como vimos en el apartado anterior, con el comando `dpkg --install`, instalando así mismo aquellos paquetes de los que dependa y que imposibiliten su configuración. En este punto es posible que la dependencia de alguna versión más reciente de otro programa no obligue a compilar ese programa desde el código fuente. Al igual que antes, esto es más probable cuando el tiempo que separa las distribuciones estabe e inestable es mayor.

Capítulo 6

Españolización de Linux

6.1 Configuracion del teclado para el modo texto (consola)

Cuando se realiza el proceso de instalacion, entre otras cosas, se configura el teclado para la consola. No obstante, una vez instalado es posible volver a reconfigurarlo. La forma más sencilla de hacer esto es reconfigurando el paquete `console-data` mediante el comando:

```
# dpkg-reconfigure console-data
```

que nos mostrará un asistente para la configuración de forma sencilla del mapa del teclado mediante menús. Por ejemplo, para establecer la configuración para teclados españoles podemos seguir las opciones *Select keymap from arch list –> qwerty –> Spanish –> Standard –> Standard*.

A continuación se describe como realizar esta configuración de forma manual, para aquellos que deseen entender cómo funciona en detalle el proceso de carga del mapa del teclado en Debian.

El comando que permite la carga de un nuevo mapa del teclado es `loadkeys`, por ejemplo:

```
# loadkeys es.kmap.gz
```

Los mapas del teclado se encuentren en el directorio `/usr/share/keymaps/`, distribuídos en varios directorios y subdirectorios en función del tipo de ordenador y teclado al que se aplican. Si no se indica la ruta completa hasta el mapa del teclado deseado, el comando `loadkeys` buscará automáticamente en estos directorios el mapa solicitado, como en el ejemplo anterior. Por ejemplo, en un ordenador compatible PC (arquitectura i386) con teclado *qwerty*, que es el tipo más extendido, los mapas del teclado se encuentran en `/usr/share/keymaps/i386/qwerty`, donde encontramos mapas para múltiples configuraciones posibles de estos teclado, como por ejemplo:

- `es.kmap.gz`: mapa de teclado español.
- `fi.kmap.gz`: mapa de teclado findlandés.

- `us.kmap.gz`: mapa de teclado inglés americano.

`loadkeys` cargará un mapa de caracteres en la sesión en la que estamos trabajando pero no de forma permanente. Para cargarlo de forma permanente, esto es, que cada vez que arranquemos el sistema carge el mapa del teclado adecuado, hay que colocar el fichero `/etc/console/boottime.kmap.gz`, que es de donde es leido por el script de iniciación del teclado `/etc/init.d/keymap.sh`. Por ejemplo, para cambiar el mapa del teclado a la disposición de inglés americano de forma permanete, haciendo una copia de seguridad al mapa antiguo, podríamos usar los siguientes comandos:

```
# cp /etc/console/boottime.kmap.gz /etc/console/boottime.kmap.old.gz
# cp /usr/share/keymaps/i386/qwerty/us.kmap.gz /etc/console/boottime.kmap.g
# /etc/init.d/keymap.sh start
```

6.2 Localización

Muchas aplicaciones y parte de la documentación de Linux están traducidos a otros idiomas (entre ellos el castellano). Definiendo las variables de entorno apropiadas el sistema usará la información relativa al idioma y país del usuario, cuando esté disponible. Una descripción detallada del soporte para múltiples idiomas puede consultarse en `locale(7)`, aquí daremos unas sencillas instrucciones para tener un entorno en nuestro idioma.

En primer lugar, es necesario que el sistema *compile* o genere los datos de configuración propios de la localización o localizaciones que deseemos emplear en el sistema. Se pueden generar tantas localizaciones como se desee de modo que cada usuario pueda elegir la configuración en cuanto a nacionalidad, formato de números, moneda, etc. que más le interese. El paquete encargado de esto es `locales`, y podemos generar datos locales y seleccionar el idioma por defecto del sistema símplemente reconfigurando este paquete:

```
# dpkg-reconfigure locales
```

Aquí se nos mostrará un asistente que nos permitirá seleccionar la configuraciones a generar. Cada configuración tiene la forma:

```
xx_YY Codificación
```

donde `xx` representa el idioma, `YY` representa el pais y `Codificación` representa el mapa de caracteres asignado. Por ejemplo:

- `en_US ISO-8859-1` representa una localización para el idioma inglés con los convenios de los EE.UU. y una codificación ISO–8859–1 (Latin–1).
- `es_ES ISO-8859-1` representa una localización para el español, con los convenios de España y la misma codificación que el anterior.

- `es_ES@euro ISO-8859-15` es similar al anterior, pero incorpora las actualizaciones adecuadas para el uso del EURO como símbolo monetario, lo cual requiere una codificación diferente (Latin–9).

A modo de ejemplo, pueden generarse las tres localizaciones anteriores y en la pantalla siguiente elegir `es_ES` como localización por defecto. Aunque es posible emplear la localización para la "zona Euro", hay que tener en cuenta que el soporte para esta localización en el entorno gráfico X–Window es limitado en la actualidad.

Cada usuario puede decirle al sistema que quiere usar el castellano y los convenios de España para fechas, moneda, etc. definiendo la variable de entorno `LANG` al valor `es_ES`. Cada usuario puede hacer esto en sus ficheros personales de configuración (`.bash_profile` o `.bashrc`) incluyendo la línea:

```
export LANG=es_ES
```

o bien el administrador puede hacerlo para todos los usuarios incluyendo la línea:

```
LANG=es_ES
```

en el fichero `/etc/environment`, y eliminando (o comentando) cualquier otra definición de la variable `LANG`. Esto es precisamente lo que hace el asistente al reconfigurar el paquete `locales` y seleccionar `es_ES` como localización por defecto.

Hecho esto, y una vez que hemos abierto una nueva sesión para leer el nuevo valor, debemos tener traducidos la mayoría de los mensajes de error del sistema. Por ejemplo:

```
$ ls mifichero
ls: mifichero: No existe el fichero o el directorio
```

También se mostrarán en castellano los mensajes y componentes traducidos del entorno GNOME y las páginas de manual que estén traducidas, siempre que las tengamos instaladas.

Si se emplea algún gestor del display avanzado como `gdm(8)` o `kdm(8)`, suele ser posible elegir el idioma preferido mediante un sistema de menús antes de entrar en la sesión gráfica, por lo que el usuario no necesita editar manualmente sus ficheros de configuración.

Las páginas de manual en castellano se encuentran en el paquete `manpages-es` y otra documentación sobre Linux traducida al español (como HowTo's, FAQ, etc.) se encuentra en el paquete `doc-linux-es`. El paquete `user-es` contiene el comando `castellanizar`, que puede resultar útil para configurar de forma sencilla otros aspectos del sistema para su uso en español. Puede instalarse un conjunto completo de paquetes útiles para hispanoparlantes eligiendo el grupo *Spanish environment* dentro del programa `tasksel`.

Para una descripción más detallada y general de la configuración de Linux para hispanoparlantes puede consultarse el *Spanish HowTo*, que puede encontrarse, por ejemplo, en el paquete `doc-linux-html` o en la web en `http://www.insflug.org/`.

Capítulo 7

Configuración de la hora

En Linux hay que distinguir entre dos relojes: el reloj del sistema (hora del sistema) y el reloj hardware. El reloj del sistema es un reloj software que es actualizado por unas rutinas del kernel. Este reloj fija la hora del sistema. El reloj hardware es el que incorpora la propia electrónica del sistema y que es alimentado por una pila. Es el encargado de mantener la hora cuando el ordenador está apagado. Típicamente, en Linux, el reloj hardware se consulta una vez al arrancar el sistema para fijar la hora del sistema y a partir de aquí funcionan de forma independiente. Al parar el sistema se suele almacenar la hora en el reloj hardware para conservarla. En la práctica, el usuario (y el administrador) pueden olvidarse del reloj hardware ya que este será leído y actualizado en el arranque y la parada de forma automática.

Por otra parte, mantener un valor correcto de la hora del sistema es algo importante y llega a ser fundamental en un servidor conectado permanentemente. Muchas tareas y programas dependen de la exactitud de la hora del sistema para que se realicen correctamente. Dicho esto, veamos las distintas tareas que tienen que ver con la lectura y actualización de la hora del sistema.

7.1 Ver la hora actual del sistema

El comando básico para consultar (y también para establecer) la hora es `date(1)`. Ejecutado sin argumentos muestra la hora actual:

```
$ date
mar ene 23 17:31:17 CET 2001
```

7.2 Cambiar la hora

Para establecer una nueva hora del sistema se emplea el comando `date` con un argumento que expresa el valor de la hora. Sólo el superusuario puede establecer la hora del sistema. El formato es:

```
# date MMDDHHMM[[CC]YY][.SS]
```

donde hay que especificar todos los dígitos, y significan:

- MM: número del mes actual.
- DD: día del mes.
- HH: hora.
- MM: minuto.
- CCYY: año. Es opcional y pueden indicarse solamente las dos últimas cifras del año.
- SS: segundo (opcional).

7.3 Escribir la hora en el reloj hardware

Esta operación se realizará automáticamente al parar el sistema, siempre que se pare correctamente usando el comando `shutdown`. No obstante, se puede ejecutar manualmente el script que salva la hora si se desea, esto es:

```
# /etc/init.d/hwclock.sh stop
```

y el script se encargará de ejecutar el comando `hwclock(8)` con los parámetros necesarios.

7.4 Establecer la hora desde el reloj hardware

De la misma forma, esta operación se realiza automáticamente al arrancar el sistema, pero se puede hacer de forma manual ejecutando:

```
# /etc/init.d/hwclock.sh start
```

De nuevo, el script ejecuta `hwclock(8)` pero ahora con opciones para leer el reloj hardware.

Tanto en la operación de escritura como de lectura interviene un parámetro importante que indica al sistema si queremos que el reloj hardware guarde la hora local de nuestro lugar de residencia o bien la hora universal. Lo más práctico es que el reloj hardware guarde la hora universal y el sistema calcule la hora local a partir de ésta. Una ventaja es que el sistema cambiará automáticamente de horario de verano a invierno y viceversa, conservando siempre la hora universal en el reloj hardware. El único inconveniente es que en la misma máquina se ejecute otro sistema operativo (Como MS–Windows™) que haga ajustes incompatibles del reloj hardware. En este caso, se recomienda que se ajuste el reloj hardware para que almacene la hora universal y, en caso de ver anomalías en la hora, probar la otra opción.

El uso de la hora universal para el reloj hardware se configura al instalar el sistema y se guarda en la variable `UTC` definida en el fichero `/etc/defaults/rcS`, que puede modificarse posteriormente. Este fichero contiene a su vez otras opciones para tareas ejecutadas en el arranque del sistema. Definiendo esta variable al valor "`yes`"mantendremos la hora universal, y definiendo el valor "`no`"mantendremos la hora local.

7.5 Establcer y mantener la hora desde un servidor de hora

Con el comando `ntpdate(1)` incluido en el paquete `ntpdate`, puede establecerse la hora, tomándola desde un servidor de hora NTP. Por ejemplo:

```
# ntpdate hora.rediris.es
```

Para que esta operación se realice automáticamente en el arranque del sistema hay que configurar el programa con una o varias direcciones de servidores NTP que serán leidas en el arranque. Esta configuración se realiza fácilmente usando el comando:

```
# dpkg-reconfigure ntpdate
```

e introduciendo los parámetros requeridos, que serán almacenados en el fichero `/etc/default/ntp-servers` para su uso durante el arranque del sistema.

Otra posibilidad para mantener la hora del sistema en ordenadores que están permanentemente conectados a Internet es instalar un servidor NTP. El paquete a instalar es `ntp-simple` y al instalarse nos pedirá la dirección IP de uno o varios servidores que conozcamos para sincronizarse con ellos. Esta configuración se puede alterar posteriormente editando el fichero `/etc/ntp.conf` o ejecutando:

```
# dpkg-reconfigure ntp-simple
```

Otra cuestión es como conseguir la IP de un servidor de hora. Las máquinas que actúan como *routers* o *gateways* suelen incorporar uno. Podemos averiguarlo ejecutando el comando `ntpdate` sobre ellas. Si esto no funciona tendremos que preguntar a algún gurú.

7.6 Cambiando la zona horaria

Durante el proceso de instalacion se configura la zona horaria en la que nos encontramos. Esta informacion es importante sobretodo si el reloj hardware guarda la hora universal ya que, en ese caso, es necesaria para establecer la hora local. Concretamente, con la informacion de zona horaria, la hora universal y la fecha en la que nos encontramos se determina cual es la hora local.

La zona horaria que tiene configurada el sistema se guarda en el fichero `/etc/timezone`:

```
Europe/Madrid
```

Para cambiarla el superusuario puede ejecutar:

```
# tzconfig
Your current time zone is set to Europe/Madrid
Do you want to change that? [n]:
Your time zone will not be changed
```

Si se responde *y* a la pregunta de cambio se entra en un programa que permite selecionar la nueva zona horaria. Este programa se encarga de cambiar la fecha y hora local segun la nueva zona horaria.

Capítulo 8

Configuración de la impresora

Tras la instalación de Debian encontramos configurada una impresora por defecto. El problema es que en esta configuración el sistema se limita a enviar a la impresora los datos que vienen de las aplicaciones sin tener en cuenta el modelo concreto de impresora que podamos tener. En los sistemas UNIX (Linux entre ellos) las aplicaciones suelen generar los datos a imprimir en formato *PostScript*, por lo que si tenemos una impresora que admita *PostScript* podremos imprimir sin problemas. El caso es que la mayoría de las impresoras pequeñas emplean otros lenguajes de más bajo nivel, por lo que es necesario instalar un filtro que convierta el PostScript en el formato de la impresora. Estos filtros suelen ir más allá y son capaces de convertir automáticamente otros formatos como GIF, JPG, PDF, etc.

Dicho esto, el sistema de impresión en Linux lo forman los siguientes componentes:

- Módulo del kernel para control del puerto paralelo.
- Servidor de impresión (*printer spooler*) y utilidades de control.
- Filtros para diferentes modelos de impresoras.

En este capítulo explicaremos como configurar cada uno de estos componentes. Por otra parte, la distribución Debian incluye desde la versión 3.0 (woody) una versión avanzada del sistema de impresión *CUPS (Common Unix Printing System)* que que incorpora un servidor de impresión avanzado, un sistema de filtros propio y un sencillo sistema de configuración mediante web. Esta es una buena alternativa a los servidores y filtros tradicionales y puede consultarse directamente en el apartado ‘Common Unix Printing System (CUPS)’ en la página 58.

8.1 Soporte en el kernel para puerto paralelo

Normalmente no tendremos que preocuparnos por esto, pues el soporte suele estar compilado en el kernel, como ocurre con los kernel estándar de Debian, pero cuesta poco comprobarlo. Para ello comprobamos los mensajes del kernel y vemos si hay alguno relacionado con el puerto paralelo. Por ejmplo:

```
$ dmesg|egrep 'lp|parport'
parport0: PC-style at 0x378 [SPP,PS2]
lp0: using parport0 (polling).
```

Vemos que este kernel tiene incluido el soporte para puerto paralelo. Si no lo tuviera, seguramente tendremos el modulo disponible y sera cargado automáticamente cuando se necesite. Podemos asegurarnos de que se carga ejecutando el programa `modconf`. El módulo a cargar se llama `lp` y se encuentra en la sección *misc*. Si el módulo no estuviera presente, tendríamos que instalar uno de los paquetes `kernel-image` que lo tuviera.

8.2 Servidor de impresión y utilidades de control. Sistema tradicional

El servidor de impresión instalado por Debian es `lpd(8)`. Alternativamente, podemos instalar uno compatible (`lprng(8)`) con funciones extra que permiten utilizar la herramienta gráfica de gestión de colas `printop(1)`. Para hacer la instalación basta con ejecutar

```
# apt-get install lprng printop
```

y suministrar al sistema, en su caso, los CD's necesarios. Durante la instalación de `lprng` se nos preguntará si queremos crear un fichero `/etc/printcap` nuevo o conserver el antigo. Podemos hacer cualquiera de las dos cosas, porque este fichero será sustituido en el apartado siguiente.

8.3 Filtros de impresión

El último componente que necesitamos para nuestro sistema de impresión es un filtro adecuado para nuestro modelo de impresora. En Debian se incluyen al menos dos paquetes de filtros: `magicfilter` y `apsfilter`. Ambos se basan en el programa `gs(1)` (GhostScript) que traduce el formato PostScript a múltiples formatos gráficos y de impresora, así como en otros conversores de formato. En este apartado describiremos la instalación y configuración de `magicfilter`.

Para instalar `magicfilter` hacemos:

```
# apt-get install magicfilter
```

Si no tenemos creado el fichero `/etc/printcap`, se ejecutará automáticamente la herramienta de configuración `magicfilterconfig(8)`. En caso contrario nos dará un mensaje y tendremos que eliminar manualmente este fichero y ejecutar el programa posteriormente. Por lo que pueda pasar, es recomendable salval el fichero de configuración anterior cambiándole simplemente el nombre:

```
# mv /etc/printcap /etc/printcap.old
```

Ahora podemos ejecutar `magicfilterconfig` manualmente para configurar nuestra impresora. Los pasos a seguir son:

1. Elegimos un nombre descriptivo para la impresora, por ejemplo “HP Deskjet 690C”
2. Luego se nos pedirá un nombre corto para dar nombre a la cola de impresión correspondiente a esta impresora. En este nombre no deben haber espacios, por ejemplo “dj690c”
3. Ahora se nos preguntará por el puerto paralelo al que está conectada la impresora. Si sólo tenemos un puerto paralelo en el ordenador, como suele ser el caso, este será casi con toda seguridad `/dev/lp0`
4. Ahora viene la parte más importante. Se nos mostrará una lista de posibles filtros y tenemos que elegir el adecuado para nuestra impresora. A menudo, elegir el filtro adecuado es un proceso de ensayo y error. Cabe además la posibilidad de que nuestra impresora no esté soportada por ninguno de los filtros. Esto es más que probable si se trata de impresoras tipo *Winprinter* o diseñadas específicamente para MS–Windows™. Algunos de los filtros más comunes son:
   - StylusColor–*: impresoras Epson StylusColor. Existe una gran variedad de impresoras de este tipo y no todas han de estar soportadas por los filtros.
   - bj*: impresoras Cannon de chorro de tinta.
   - dj500: impresoras HP Deskjet sólo blanco y negro.
   - dj500c: impresoras HP Deskjet en B/N y color pero que sólo admiten un cartucho instalado simultáneamente.
   - dj550c: impresoras HP Deskjet en B/N y color con posibilidad de dos cartuchos simultáneos.
   - ljet*: impresoras HP Laserjet.
   - ps*: impresoras PostScript.

   Si no encontramos un driver específico para nuestra impresora, es probable que otro para un modelo similar sirva, aunque a veces, modelos similares en número de serie son radicalmente distintos en hardware. Por ejemplo, los filtros para impresoras HP emplean el lenguaje PCL y es muy probable que funcionen con cualquier impresora HP que entienda este lenguaje. Ante la duda, podemos elegir uno que parezca adecuado y cambiarlo posteriormente.
5. Si no deseamos añadir otra cola de impresión, finalizamos indicando "`none`"como nombre de la siguiente cola. Se pueden añadir varias colas de impresión para una misma impresora para usar distintos filtros, por ejemplo `dj690c`, `dj690c-best` y `dj690c-low` para tener diferentes calidades de impresión.

8.4 Gestión de colas de impresión. Comando BSD.

Recordemos que para enviar ficheros a la impresora se emplea el comando `lpr(1)`, por ejemplo,

```
$ lpr carta.ps
```

imprime el fichero `carta.ps` en la impresora por defecto. Si hay varias colas instaladas, podemos seleccionar una concreta con la opción `-P`:

```
$ lpr -Pdj690c-low carta.ps
```

El comando `lpq(1)` permite ver la lista de trabajos en una cola y el comando `lprm(1)` permite eliminar un trabajo de la cola indicando el número de trabajo (*job*) proporcionado por `lpq`. Ambos comandos también emplean la opción `-P` para referirse a una cola en concreto. Para tareas de control complejas se emplea el comando interactivo `lpc(1)`, aunque éstas rara vez son necesarias salvo en grandes servidores de impresión. Opcionalmente, estas tareas se pueden realizar en modo gráfico con el programa `printop(1)`.

8.5 Refinar/modificar la configuración

Siempre podemos borrar el fichero `/etc/printcap` y ejecutar `magicfilterconfig` de nuevo para cambiar la configuración, pero es mucho más sencillo hacer los cambios a mano editando los ficheros correspondientes. Aquí damos algunas pistas sobre cómo hacer modificaciones en la configuración del sistema de impresión.

La configuración de las impresoras se guarda en el fichero `/etc/printcap` (ver `printcap(5)`). Editando este fichero resulta fácil modificar ciertos parámetros como:

- Nombre de la cola de impresión, indicado en la primera línea de configuración de cada impresora. Normalmente se definen varios nombres para cada cola, separados por "`|`".
- Filtro de impresión utilizado, que se indica con el parámetro "`if=`".
- También podemos editar el filtro de impresión (hacer antes una copia de seguridad del original) y añadir parámetros a la línea de comandos donde se ejecuta `gs` si es que nuestra impresora necesita algún parámetro concreto.

8.6 Common Unix Printing System (CUPS)

El sistema CUPS (`http://www.cups.org`) es una nueva concepción de la arquitectura de impresión para sistemas UNIX que tiene como objetivo homogeneizar los sistemas de impresión en los sistemas UNIX, facilitar la configuración y el mantenimiento del sistema de impresión y buscar la compatibilidad con sistemas de impresión de otros sistemas operativos. CUPS sustituye a servidores tradicionales como los incluidos en los paquetes `lpr` y `lprng` e incluye su propio sistema de filtros de impresión y mantiene la compatibilidad con los sistemas anteriores. CUPS emplea el protocolo de impresión estándar *IPP (Internet Printing Protocol)*, usado por otros sistemas operativos como MS–Windows(TM).

Cuando se instalan los paquetes de compatibilidad con los sistemas tradicionales de impresión de UNIX, CUPS puede controlarse usando comandos convencionales como `lpr`, `lpq`, `lprm` y `lpc`.

8.6.1 Instalación de CUPS

El sistema CUPS puede instalarse fácilmente, incluyendo filtros de impresión y programas complementarios con el siguiente comando:

```
# apt-get install cupsys cupsys-client cupsys-bsd cupsys-driver-gimpprint
```

Para poder imprimir en impresoras de red de servidores MS–Windows(TM) es necesario instalar también el paquete `smbclient`:

```
# apt-get install smbclient
```

8.6.2 Configuración/gestión de impresoras

La configuración y gestión de impresoras y colas de impresión se controla desde una interfaz web que proporciona el propio servidor CUPS. De hecho, el servidor CUPS actúa como un servidor web conectado al puerto 631. Para conectarnos a la interfaz de configuración de CUPS basta con con conectarnos con cualquier navegador web a la siguiente dirección: `http://localhost:631/`. La interfaz es muy intuitiva. Aquí describiremos algunas tareas comunes.

Para añadir una nueva impresora seguimos los siguientes pasos:

- Entramos en el apartado *Administration*. Aquí se nos pedirá un nombre de usuario, donde debemos escribir *root* y una clave, donde debemos escribir la clave del administrador de la máquina.
- Elegimos el botón *Add Printer*
- Escribimos un nombre para la impresora, por ejemplo *lpcolor*, una descripción de su ubicación, por ejemplo *Despacho de Manolo* y una descripción más detallada, por ejemplo *HP Deskjet 690C*, y pulsamos *Continue*
- A continuación elegimos el tipo de conexión de la impresora. Por ejemplo, para una impresora local conectada al puerto paralelo sería *Parallel Port #1*, para una impresora conectada a un servidor IPP sería *Internet Printing Protocol (ipp)* y para una impresora conectada a una máquina Windows sería *Window Printer via SAMBA*. Pulsamos *Continue*.
- Si se trataba de una impresora remota, se pedirá la dirección de la impresora. Se muestran varios ejemplos con los posibles formatos de direcciones. Por ejemplo, si se trata de una impresora llamada *lp1* conectada a una máquina Windows de nombre *asterix*, la dirección sería `smb://asterix/lp1`.
- A continuación se indica la marca de la impresora, que determinará el *driver* a utilizar. Por ejemplo, *HP*.
- Luego se muestra una lista de modelos correspondientes donde debemos elegir el de nuestra impresora o probar con uno parecido. Por ejemplo, *HP DeskJet 690 series* .... Al pulsar *Continue* la impresora quedará configurada.

A continuación resumimos la función del resto de secciones de la interfaz de configuración:

- *Classes*: permite definir conjuntos de impresoras que serán administradas por el sistema como si se tratase de una sola impresora. Muy útil en grandes servidores de impresión donde el servidor puede repartir los trabajos entre varias impresoras de la misma clase.

- *Help*: es un enlace a la documentación completa del sistema CUPS.
- *Jobs*: permite controlar los trabajos de impresión en curso.
- *Printers*: permite controloar las impresoras instaladas, así como modificar las opciones de instalación y configurar las opciones de impresión específicas de cada impresora.

Existen otros programas gráficos que permiten la configuración y control de las impresoras como `gtklp` y `gtklpq`, incluidos en le paquete `gtklp`.

8.6.3 Configuración del servidor CUPS

Los aspectos más avanzados del servidor CUPS como el control de accesos, exportar impresoras, etc. se definen en el fichero de configuración del servidor `/etc/cups/cupsd.conf`. Los detalles de esta configuración quedan fuera de los propósitos de esta guía, pero puede encontrarse información detallada en la propia documentación del sistema CUPS en `http://localhost:631/documentation.html`. Así mismo, muchos aspectos del servidor pueden configurarse con el programa gráfico `cupsd-conf`, incluido en el paquete `kdelibs3-cups`.

8.7 Información adicional sobre impresión

Otras fuentes de información complementaria son:

- Printing HOWTO y Printing–Usage HOWTO o sus traducciones: Configuración–Impresión–Como y Uso–Impresión–Como.
- Una descripción de algunos filtros de impresión de GhostScript se encuentra en el sistema en `/usr/doc/gs/devices.txt.gz`.
- Una lista de filtros de impresión de GhostScript y las impresoras que soportan se encuentra en la web en: `http://www.cs.wisc.edu/~ghost/doc/printer.htm`
- Página web de GhostScript: `http://www.cs.wisc.edu/~ghost`
- Página web de CUPS: `http://www.cups.org/`

Capítulo 9

Instalando nuevos modulos y compilando un nuevo *Kernel*

9.1 Instalando nuevos modulos

En muchos casos es necesario instalar los drivers adecuados para conseguir que este operativo todo el hardware de un ordenador. En Linux los drivers se denominan módulos. Estos son rutinas que se encuentran en el directorio:

```
/lib/modules/X.Y.Z/tipo_módulo/*.o
```

y están preparados para cargarse o descargarse dinámicamente en el kernel.

Durante el proceso de instalación se dispone de un conjunto de módulos básicos por si es necesario cargar alguno para completar correctamente la instalación del sistema. (Por ejemplo, para realizar una instalación a través de red –http o ftp– es necesario cargar el módulo correspondiente a la tarjeta de red del ordenador).

En Debian GNU/Linux la visualización, carga y descarga de un módulo en el kernel ejecutando como usuario root el comando:

```
# modconf
```

9.1.1 Ejemplo de instalación del módulo de la tarjeta de sonido SB128PCI

Vamos a desarrollar un ejemplo de instalación de un módulo correspondiente a una tarjeta de sonido muy común, la Sound Blaster 128 PCI (SB128PCI).

En el proceso de instalación de módulos lo primero que se debe realizar es determinar que tipo de hardware concreto se tiene para poder seleccionar el módulo adecuado a ese hardware.

En el caso de las tarjetas PCI existe una forma desde el sistema operativo de determinar cuantas y que tipo de tarjetas existen. Para ello hay que ejecutar como usuario root el comando:

```
# lspci
```

En el caso de la tarjeta SB128PCI se obtiene la información siguiente:

```
Ensoniq es1371
```

Con esta información se debe buscar y cargar el módulo correspondiente. En muchos casos puede ocurrir que este módulo no este disponible. Entonces será necesario poner disponibles nuevos módulos ya precompilados que están en la distribucción.

Para ello hay que instalar algún paquete del tipo:

```
kernel-image.X.Y.Z
```

Una vez seleccionado e instalado el paquete adecuado, están disponibles tanto una nueva imagen del kernel X.Y.Z en el directorio /boot denominado:

```
vmlinuz-X.Y.Z
```

como un conjunto amplio de módulos asociados a este kernel en el directorio /lib/modules/X.Y.Z/.

Para poder visualizar estos nuevos módulos es necesario arrancar el sistema con este nuevo kernel. De hecho, durante el proceso de instalación del paquete kernel–image se han creado los siguientes enlaces simbólicos:

```
/vmlinuz       ----> /boot/vmlinuz-X.Y.Z (nuevo kernel X.Y.Z)
/vmlinuz.old   ----> /boot/vmlinuz-H.G.J (antiguo kernel H.G.J)
```

y además, se ejecuta el comando lilo para poder arrancar el sistema con el nuevo kernel. (El fichero de configuración de lilo lilo.conf no es cambiado durante este proceso de tal forma que si esta bien configurado se podra arrancar el sistema Linux tanto con el nuevo kernel como con el antiguo.)

Una vez que están disponibles los nuevos módulos y se ha arrancado el sistema con el nuevo kernel se podrán visualizar con el comando:

```
# modconf
```

dentro de `modconf` aparecerán una serie de secciones y subsecciones. Los nombres y organización de éstas dependerá de la versión del kernel que se trate. Por ejemplo, para kernels de la serie 2.4 el módulo para esta tarjeta se encuentra en la sección *kernel/drivers/sound*. Dentro de este menú seleccionar el módulo adecuado a la tarjeta, en este caso:

```
es1371 - Ensoniq AudioPCI 97 (ES1371) based sound cards
```

En algunos casos es necesario dar un conjunto de parámetros para que el módulo funcione correctamente. Por lo general, las tarjetas PCI no necesitan de estos parámetros porque el módulo es capaz de detectarlos automáticamente.

9.2 Compilando un nuevo *kernel*

El kernel, basicamente, actua como mediador entre las aplicaciones y el hardware de la maquina. El motivo fundamental para cambiar o actualizar un kernel consiste en disponer de algun hardware nuevo que no tenga soporte en el kernel disponible y, por tanto, sea necesario compilar uno mas actualizado. Otros motivos alternativos serian: pueden ejecutar procesos mas rapidos que las versiones mas antiguas, corrigen errores de las versiones anteriores etc.

El proceso de compilar un kernel sigue una serie de pasos que son:

1. Obtener y desempaquetar los fuentes
2. Configurar el kernel
3. Compilar el kernel
4. Instalar el nuevo kernel

9.2.1 Obteniendo y desempaquetando los fuentes

Para obtener los fuentes de un kernel hay diferentes mecanismo o lugares para encontrarlos. Una opción es vía ftp anónimo en ftp.kernel.org. En Debian 2.2 otra opción es mediante los paquetes de la distribución.Lo más habitual es que vengan empaquetados en un único fichero de nombre `linux-x.y.z.tar.gz` o bien con sufijo bz2 lo cuál indica que han sido comprimidos con bzip2.

Una vez obtenidos los fuentes hay que desempaquetarlos. Para ello hay que colocar el fichero comprimido con los fuentes en el directorio `/usr/src` y, aquí, proceder a descomprimirlos bien con *gunzip* (si tiene sufijo gz) o con `<bunzip2>` (si tienen sufijo bz2):

```
host:/usr/src# bunzip2 kernel-source-2.4.5.tar.bz2
```

Una vez descomprimidos hay que deshacer el tar:

```
host:/usr/src# tar xvf kernel-source-2.4.5.tar
```

9.2.2 Configurando el kernel

Este es el paso más delicado en todo el proceso de compilación de un nuevo *kernel* ya que podría ocurrir que incluso no pudiera arrancarse el sistema si no se selecciona alguna opción fundamental para el mismo.

Una vez que el superusuario se sitúe en el directorio `/usr/src/linux/` donde se encuentra toda la estructura de directorios y ficheros que contienen los fuentes del *kernel*, se puede ejecutar el comando:

```
host:/usr/src/linux# make config
```

lanza un script de configuración en modo texto. Sin embargo, existen otros modos más adecuados y fáciles de usar. En concreto en modo texto el más ampliamente usado es:

```
host:/usr/src/linux# make menuconfig
```

Si esta operativo el sistema gráfico X–window, es más cómodo usar:

```
host:/usr/src/linux# make xconfig
```

Dentro de estos scripts hay que responder con si (y) o no (n) o con la opción m de módulo. La opción si significa que este módulo se compilará dentro del *kernel*, la opción no indica que no se compilará, mientras que la opción m significa que se compilará el módulo pero que no se incluirá dentro del *kernel*.

9.2.3 Compilando el *kernel*

Una vez finalizada la fase de configuración, cuando grabamos o almacenamos esta configuración el propio script nos pide que ejecutemos los siguientes comandos consecutivamente:

```
host:/usr/src/linux# make dep ; make clean
```

El primero asegura que todas las dependencias, como por ejemplo los include se encuentran en su sitio. El segundo borra todos los ficheros objeto que hubieran sido generados en una fase de compilación anterior.

Tras las dependencias y la limpieza hay que compilar el *kernel*. Para ello se ejecuta el comando:

```
host:/usr/src/linux# make bzImage
```

o bien:

```
host:/usr/src/linux# make bzdisk
```

El primero compilará el *kernel* y lo guardará en fichero llamado `bzimage` en el directorio `/usr/src/linux/arch/i386/boot/`. El segundo hace los mismo pero, además, guarda el kernel en un *floppy disk*. Esta opción permite que se pueda arrancar el sistema con el nuevo kernel sin tener que instalar el kernel en el sector de arranque del disco duro, de tal forma que si existiesen problemas de funcionamiento pudiera volver a arrancarse el sistema con el *kernel* anterior.

9.2.4 Instalando el nuevo *kernel*

Para poder arrancar con el nuevo *kernel* es necesario instalarlo en el sector de arranque adecuado con un programa gestor de arranque. El más conocido es `lilo`. En este proceso se tiene que copiar el kernel al fichero `/vmlinuz` (o bien hacer un link simbólico). Además, conviene pasar el antiguo kernel a */vmlinuz.old* para dejar activos tanto el antiguo como el nuevo, para poder arrancar desde cualquiera de los dos. Una vez copiados o convenientemente *linkados* se debe actualizar convenientemente el fichero `/etc/lilo.conf`:

```
...

image=/vmlinuz
        label=Linux
        read-only
#       restricted
        alias=deb

image=/vmlinuz.old
        label=LinuxOLD
        read-only
        optional
#       restricted
        alias=2

...
```

Una vez modificado este fichero de configuración del gestor de arranque `lilo`, se debe ejecutar el comando lilo para que se instale la nueva configuración en el sector de arranque:

```
# lilo
```

9.2.5 Compilando e instalando los nuevos módulos del *kernel*

Colocándonos de nuevo en el directorio `/usr/src/linux/` se debe ejecutar el comando:

```
host:/usr/src/linux# make modules
```

Con ello se compilan los módulos que se hubieran marcado con m en el proceso de configuración del *kernel*. Posteriormente se debe ejecutar:

```
host:/usr/src/linux# make modules_install
```

Con este comando se instalarán los módulos en el directorio `/lib/modules/x.y.z/`. A partir de ahora serán accesibles con el comando:

```
# modconf
```