Guía de administración de Debian GNU/Linux

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

Versión 0.1 (25 enero 2001)

Resumen

Este documento es una guía de introducción a la administración de Debian GNU/Linux para principiantes, correspondiente a la versión 2.2 (potato) 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 usuarios principiantes y a la resolución de los aspectos más frecuentes relacionados con la administración del sistema. Con esta guía, un usuario debe ser capaz 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 e 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. *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"

# 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=Win95
# 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.
```

La siguiente línea:

```
append="mem=256M"
```

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).

```
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`.

```
other=/dev/hda1
label = Win95
alias = 3
```

Describe la situación de otros sistemas operativos. En este caso se trata de MS–Windows95(TM) situado en la primera partición del primer disco IDE.

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 la funcionalidad del 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 ow
```

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 la 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.

Capítulo 3

Sistema de ficheros

En este capítulo se describe el sistema de ficheros y 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` continen 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 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.

El 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)`

3.4 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.

3.5 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.

3.5.1 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.

3.5.2 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.

3.5.3 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 *passwd*. 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*
```

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 incluidos 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 actúa 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) 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 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
```

- 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 communicator gimp gnome-xbill
```

instalará los paquetes `communicator`, `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 communicator gimp gnome-xbill
```

desinstala los paquetes anteriores, y

```
# apt-get --purge remove communicator 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 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 proceso no es difícil pero se considera administración avanzada. Se invita al lector a que explore esta vía.

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 de fichero `.deb`.

Capítulo 6

Españolización de Linux

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.

Básicamente, podemos decirle al sistema que queremos 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`. 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.

Finalmente, las páginas de manual en castellano se encuentran en el paquete `manpages-es` y otra documentación sobre Linux (como HowTo's, FAQ, etc.) se encuentra en el paquete `doc-linux-es`. Si queremos instalar estos paquetes y otros relacionados podemos instalar simplemente el paquete `task-spanish`:

```
# apt-get install task-spanish
```

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 modificar ligeramente el script `/etc/init.d/ntpdate` para incluir una o más direcciones IP de servidores de hora. La parte a modificar de este fichero quedaría algo así:

```
...
# echo "You must customize /etc/init.d/ntpdate before ntpdate can be run."
# exit 0

case "$1" in
start)
  echo -n "Running ntpdate to synchronize clock"
  /usr/sbin/ntpdate -b -s 192.168.1.1 100.100.100.100
...
```

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` 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 el programa `configure-ntp`.

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.

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.

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, 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

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

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 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`

Capítulo 9

Configuración de X–Window

En este capítulo daremos unas guías para la instalación del entorno gráfico X–Window en Debian y algunas notas para configurar algunos aspectos del entorno con posterioridad a la instalación.

9.1 Recomendaciones para la instalación de X–Window

Si tenemos espacio suficiente en disco, podemos instalar un completo sistema gráfico instalando los siguientes paquetes: `task-x-window-system`, `task-gnome-desktop`, `task-gnome-apps`, `sawmill`, `sawmill-gnome` y algún servidor X, seguramente `xserver-svga`. Para la instalación, deberemos tener a mano la siguiente información:

- Puerto y tipo de ratón. Los ratones modernos con conector circular son tipo PS/2 y se corresponden con el dispositivo `/dev/psaux`. Los ratones serie irán conectados a `/dev/ttyS0` o `/dev/ttyS1`, normalmente. Es conveniente hacer un enlace simbólico con nombre `mouse` al dispositivo real para su uso futuro. Por ejemplo, si el ratón es tipo PS/2 haríamos:

  ```
  # ln -s /dev/psaux /dev/mouse
  ```

- Modelo y marca de la tarjeta gráfica y, si es posible, cantidad de RAM instalada, etc. Hoy día, prácticamente todas las tarjetas son PCI o AGP. Se puede obtener información sobre ella listando los dispositivos PCI con el comando `lspci(8)` y apuntando la información del dispositivo VGA detectado.

- Rango de frecuencias horizontales y verticales soportadas por el monitor. Esta información viene en el apartado de especificaciones del manual del monitor. Si no se tiene esta información, se pueden elegir más tarde valores típicos, pero el rendimiento no será óptimo.

Existen diversos programas para configurar X–Window. Aquí describiremos `XF86Setup(1)`, incluido en el paquete `xf86setup`. Para ejecutarlo es necesario tener instalado el servidor VGA genérico (paquete `xserver-vga16`)

Cuando ejecutamos `XF86Setup` (como root) lo primero que encontramos es la pantalla de selección del ratón. Si hemos definido el enlace simbólico `/dev/mouse` mencionado más arriba y hemos tenido suerte, nuestro ratón funcionará. Probaremos los botones y si no funciona el central (o no existe) elegiremos la opción *Emulate3Buttons*. Si el ratón hace cosas "extrañas" es porque el protocolo elegido no es el correcto y probaremos a cambiarlo pulsando TAB y ENTER. Si el ratón no se mueve en absoluto, probaremos a seleccionar otros dispositivos, como `/dev/ttyS0` o `/dev/gpmdata`.

Cuando el ratón no funciona correctamente es fácil que nos quedemos "colgados" sin poder salir del programa de configuración. En este caso podemos interrumpir el programa pulsando CTRL–ALT–BACKSPACE.

En la sección de teclado elegir un teclado de 104 teclas (a menos que sea un modelo antiguo) e idioma español (Spanish).

La siguiente pantalla es la de elección de tarjeta de vídeo. Si la nuestra no aparece, elegiremos una similar. Si el servidor X para esa tarjeta no está instalado, recibiremos unas señal auditiva y un mensaje nos dirá qué servidor corresponde a la tarjeta. Entonces saldremos del programa e instalaremos el servidor necesario. El servidor SVGA (paquete `xserver-svga`) tiene soporte para la mayoría de las tarjetas de vídeo. Otros servidores muy habituales son el MACH64 (paquete `xserver-mach64`) y el `xserver-rage128` usados con tarjetas ATI. Cada servidor X viene en un paquete Debian independiente cuyo nombre comienza por `xserver-`. En general no es necesario indicar nada en la sección de configuración detallada (Detailed Setup) a menos que el servidor no sea capaz de detectar correctamente algún dato como la memoria instalada en la tarjeta.

Si no se encuentra un servidor X adecuado, se puede consultar la documentación incluida en el directorio `/usr/doc/xserver-common`, en la página web de XFree86 (*www.xfree86.org*) o preguntar en algún grupo de noticias (es.comp.os.linux.hardware).

En la sección de monitor escribiremos los rangos de frecuencia correspondientes indicando el valor menor y mayor en cada caso separados por un guión, por ejemplo: 30–60, 50–130.

En la sección de selección de modos elegiremos las resoluciones que deseamos obtener haciendo que la máxima elegida sea la que usaremos normalmente. Se recomienda 1024x768 para monitores de 17"y 800x600 para monitores más pequeños. En cuanto a la profundidad de color, se recomienda elegir 16bpp incluso si nuestra tarjeta soporta más, ya que algunas aplicaciones y algunos servidores no funcionan bien a 24bpp (aunque siempre podemos probar...)

Finalmente pulsaremos "Done" (no es necesario cambiar nada en la sección "Other") y entonces el programa de configuración arrancará el servidor con los parámetros introducidos. Si el servidor no arranca tendremos que revisar la configuración, especialmente el tipo de tarjeta y las frecuencias del monitor. Si funciona correctamente nos dará la opción de ejecutar un programa para ajustar el tamaño de la imagen, pero esto no suele ser necesario en los monitores actuales con controles digitales, así que podemos aceptar la configuración y salvar el fichero.

9.2 Configuración para iniciar una sesión GNOME

Podemos configurar la sesión X para que sea gestionada por el administrador de sesiones del entorno GNOME. Caben dos posibilidades:

1. Instalar el “Gestor de Display de Gnome” (GDM) incluido en el paquete `gdm`: en este caso el sistema arrancará automáticamente el entorno gráfico cuando arranque y mostrará una ventana de entrada al sistema que nos permitirá elegir idioma y tipo de sesión.

2. Ejecutar `gnome-session(1)` (paquete `gnome-session`) al iniciar la sesión X desde la consola. Esto se consigue creando en la cuenta del usuario un fichero con nombre `.xsession` que contenga una línea con el comando `gnome-session`. Por ejemplo:

   ```
   $ echo "gnome-session" > .xsession
   ```

   De esta forma la sesión gnome se iniciará al entrar en el entorno gráfico con `startx(1)`

9.3 Otros aspectos de configuración de X–Window

La configuración del entorno X–Window se guarda en el directorio `/etc/X11`. Se pueden cambiar diversos aspectos del entorno editando los ficheros que allí se encuentran. Muchos de ellos poseen una página de manual que explica su función y posibilidades. De especial interés es el fichero `/etc/X11/Xserver`. Este fichero indica cual es el servidor X por defecto en el sistema en caso de que tengamos instalado más de uno. La primera línea contiene la ruta completa del ejecutable, mientras que la segunda indica qué usuarios están autorizados a ejecutar el servidor. La opción más común y la que figura por defecto es “Console”. Las distintas opciones y su significado se explican en el propio fichero.

Capítulo 10

Instalación de *drivers*

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
```

10.1 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 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 desde dselect algun 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 seleccionar el menu correcto (para tarjetas de sonido menó misc). Dentro de este menó seleccionar el módulo adecuado a la tarjeta (en el caso de la tarjeta de sonido SB128PCI:

```
es1371
```

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.