FACULTAD DE INGENIERÍA UNAM
DIVISIÓN DE EDUCACIÓN CONTINUA

CURSOS INSTITUCIONALES

LINUX BÁSICO

Del 23 de Septiembre al 04 de Octubre de 2002

APUNTES GENERALES

CI-372

Instructor: Ing. Rodolfo González Maldonado

CONSEJERÍA JURÍDICA
SEPTIEMBRE / OCTUBRE DEL 2002

INTRODUCCION

El presente material, sirve como soporte y complemento al curso presencial de Linux
Los temas cubiertos en el curso se detallan en este material, permitiendo asi, que el material sea utilizado tanto como material dentro del curso, como de consulta y referencia posterior.

El material cubre el uso y navegación del Linux, así como la administración del servidor Linux en su aspecto básico

OBJETIVO

El participante, aprenderá y conocerá la navegación a través del sistema operativo y administración del Linux

¿Por qué utilizar una Shell?

Las interfaces gráficas para Linux han evolucionado mucho en los últimos años. Se puede trabajar utilizando la interfaz gráfica sin necesidad de una ventana de shell.
Así que, ¿por qué aprender a utilizar la shell? por que se puede ahorrar tiempo. En el tiempo que se tarda en abrir el gestor de ficheros en GNOME o KDE, busca el directorio, por lo que crea o modifica ficheros, utilizando la shell, se obtiene el mismo resultado con pocos comandos.
En esta sección le mostraremos como moverse en el interior de una shell, como crear o modificar los ficheros, ejecutar algunos simples shell de administración y muchas otras cosas Todas estas tareas pueden ser ejecutadas también desde el interior de una ventana Xterm, sin tener que abandonar la interfaz gráfica que eventualmente está utilizando.

Protección en la fase de Inicio

Cuando esté conectado como root, deberá dedicar unos minutos a crear un *disquete de arranque* o incluso copiar el disquete de inicio.
Existen diversas razones por las que crear un disquete de arranque: le puede servir para reiniciar el sistema en caso de que el sistema falle, para comprobar una nueva versión del kernel que acaba de descargar, o incluso puede servirle de ayuda el crear particiones en su ordenador para mas de un sistema operativo.
Tiene la posibilidad de crear un disquete de arranque durante la instalación de su sistema Red Hat Linux. Si todavía no lo ha creado, este es el momento de hacerlo.
Abra una ventana de shell y asegurese que está conectado como root. Si en el indicador de la shell ve algo parecido a [billy@localhost billy]$, teclee:

```
[billy@localhost billy]$su -
Password: password-di-root
[root@localhost billy]#
```

Un comando "super"

El comando **su** significa usurario sustituto, y le permite conectarse temporalmente como otro usuario Cuando teclee **su** y pulse **Enter** se convertirá en root o superusuario mientras este en el interior de su shell de login ¿Cual es la diferencia? Existen algunos comandos que pueden ser ejecutados solo si se esta conectado como root, por lo que tecleando **su -** le será posible ejecutar aquellos comandos sin tener que salir y volver a entrar en el sistema continuamente

Notas:

MANUAL DE LINUX

INDICE

En breve buscaremos la versión del kernel de Linux en su sistema; asi que utilizaremos el comando **mkbootdisk** para crear el disquete de arranque del kernel.
Introduzca un disquete en la disquetera.

La disquetera

En Linux el lector aparece como /dev/fd0.

Si ha utilizado con anterioridad el disquete,¡ recuerde que los datos que hay en el disquete se perderán!

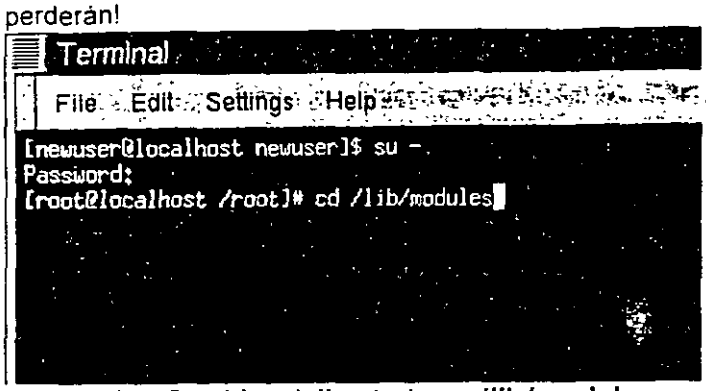

Figura 12-1. Cambiar el directorio en /lib/modules

En el indicador teclee.

cd /lib/modules

(Mostrado en Figura 12-1.)
Ahora, teclee:

ls

El comando **ls** le mostrará el contenido del directorio que nos ocupa. (Para más información sobre los comandos **cd** y **ls** Capitulo 13. Por ahora teclee los comandos tal y como los ve .
Aqui, podrá encontrar la versión del kernel de su sistema Red Hat Linux El kernel es el corazon de todo sistema Linux La versión de su kernel será similar a.

2.2.*x-yy*

(habrá diferentes números despues de 2.2, tales como 2.2.14-*xx*).
Ahora que ya ha encontrado la versión de su kernel puede decir al comando **mkbootdisk** qué kernel copiar en su disquete. (Si no le dice al comando **mkbootdisk** donde copiar el kernel, por defecto lo copiará en el disquete /dev/fd0.)
Ahora teclee

mkbootdisk --dispositivo /dev/fd0 2.2.*x-yy*

Notas:

Después pulse **Enter**.

Hacer limpieza

Si en su pantalla aparecen mensajes de diferentes tipos como por ejemplo command not found, puede volver a empezar con una pantalla de limpieza tecleando el comando **clear** en el prompt.

Hecho.

Resumen

Como root, desde una ventana del terminal, **cd /lib/modules** ; elija el número del kernel de entre aquellos que aparecen en el directorio; teclee el comando **mkbootdisk --dispositivo /dev/fd0 kernel.number**.

Para limpiar la pantalla teclee **clear**.

Notas:

Un buen "Manual" es fácil de encontrar

Como exploradores de su nuevo sistema, seguramente tendrá preguntas sobre los comandos y servicios del sistema. Uno de los métodos más fáciles para comprender el uso de muchos comandos es la utilización de las páginas **man**.
La palabra **man** significa "manual". Son una serie de "páginas" on-line que puede consultar para conocer mejor muchos comandos. De manera resumida, las páginas man ofrecen una sintesis de las caracteristicas de los comandos, las opciones disponibles y la sintaxis a utilizar.
Si se considera un "novato" de Linux le podrá parecer que las páginas man son menos útiles, que a un usuario experto. Pero las páginas man le ayudan a aprender las propiedades de los comandos de su sistema. Llegados a este punto, puede aprender mucho sobre su sistema familiarizandose con las páginas man. Ahora seguramente querrá saber como utilizarlas.
Existen diversos métodos para visualizar las páginas man graficamente:

- desde el explorador de la guia de GNOME (vea la sección de nombre *Cómo encontrar ayuda* en Capítulo 2)
- desde el explorador de la guia de KDE (vea la sección de nombre *Cómo encontrar ayuda* en Capítulo 2)
- a través de una aplicación llamada xman desde el prompt de la shell.

Notas:

A menudo necesitará acceder a una página man desde el prompt de la shell. Entonces teclee
man *nombrecomando*

Para desplazarse hacia delante en el interior de un documento pulse **Barra Espaciadora**; Para moverse hacia detrás pulse la tecla **B**. Para salir de la página man pulse la tecla **Q**.

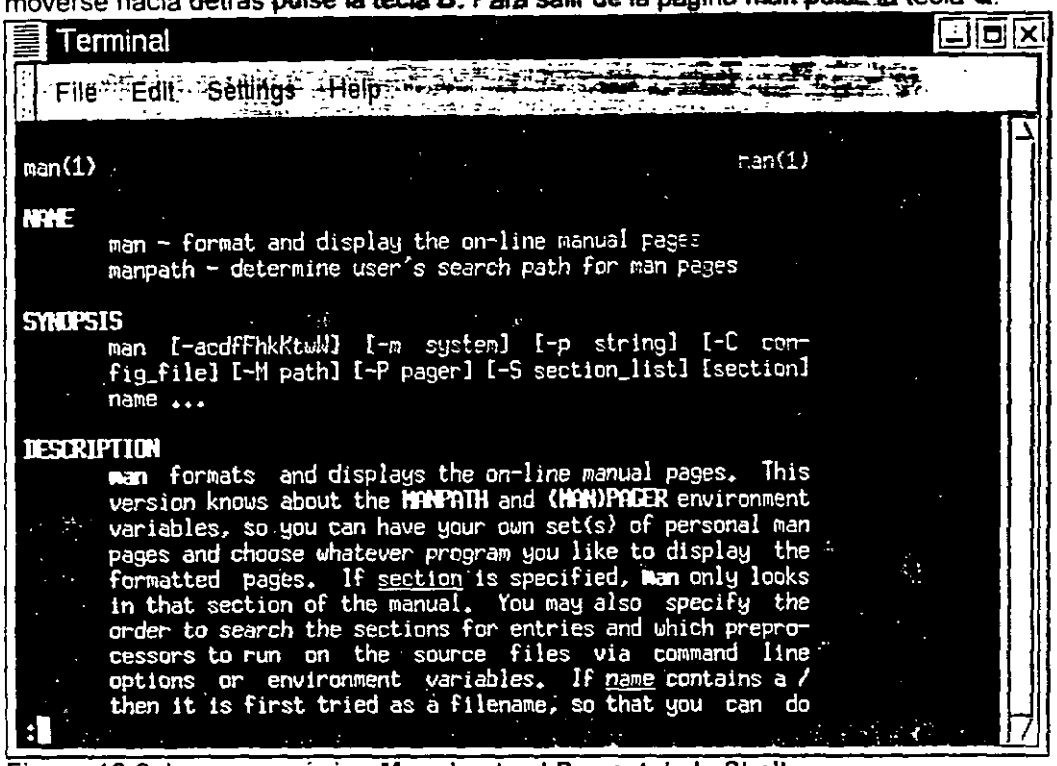

Figura 12-2. Leer una página Man desde el Prompt de la Shell

El comando man, como todo sistema de ayuda tiene su página man. En el prompt del comando teclee:

man man

para visualizar las páginas manules de man (como se muestra en la figura Figura 12-2)
Si desea imprimirlas:
algunas veces leer las paginas man en la pantalla no es suficiente Puede ser que desee tener una copia imprimida en papel. Atención por que la impresión de las paginas man podria generar impresiones con símbolos extraños (garbage), los cuales no son traducidos en el paso del video a la impresora
Por lo tanto antes de imprimir es necesario "formatear" la página con el comando **col** Si desea mayor información sobre el comando **col** teclee **man col**.
Por ejemplo, para imprimir la pagina man con el comando **man** teclee

man man | col -b | lpr

Notas:

Situarse gracias a pwd

Pronto o tarde (probablemente muy pronto), cuando empiece a mirar el contenido de un fichero, se preguntará "¿Dónde me encuentro?" y por cierto no estará haciendo una pregunta filosófica.
El DOS puede contestarle mostrándole el recorrido en el prompt como:
C:\GAMES\Game\ID1>

Sin embargo Bash, la shell de su sistema Linux muestra por defecto sólo el directorio actual.

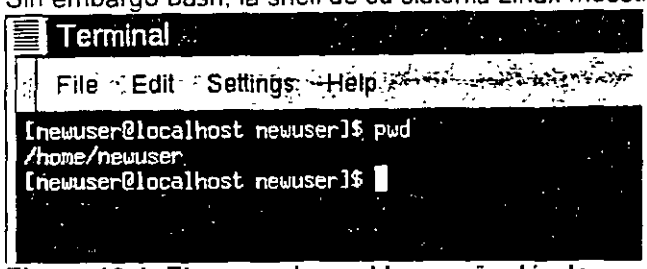

Figura 13-1. El comando pwd le enseña dónde se encuentra
Intente abrir una ventana de Xterm. Verá el siguiente prompt: prompt:
[newuser@localhost newuser]$

Ahora teclee:
pwd

Será visualizado algo parecido a:
/home/newuser

El comando **pwd** significa *print working directory* Cuando teclea **pwd**, le está preguntando a su sistema Linux: "¿Dónde me encuentro?" Su sistema le contestará "mostrándole" el directorio en el que se encuentra en la pantalla -- conocido también como *standard output*.
Parece fácil ¿verdad? Y es así; utilizará el comando **pwd** muchas veces para hechar un vistazo alrededor suyo. (incluso los expertos de Linux dependen de este pequeño programa)

Moverse en el sistema: cd

Cada vez que quiere moverse en el directorio, la único que tiene que teclear es:
cd

Siga intentándolo desde una ventana Xterm
No ha pasado mucho ¿verdad? Esto es porque no ha dicho al sistema donde quiere ir.
Cuando entra en el supermarcado o visita a su familia en el campo, tiene que saber como llegar de un punto a otro, tiene que conocer el camino correcto
Como en cada caso de la vida, el recorrido -- o pathname -- es fundamental para elegir la dirección a tomar. En el caso de su sistema Linux (así como en el mundo DOS/Windows), necesitará saber el camino para desplazarse de un directorio a otro.
Inténtelo otra vez. Abra una ventana Xterm. Compruebe su posición con el comando **pwd**. Tecleando este comando en la ventana aparecera:

Notas:

[newuser@localhost newuser]$ **pwd**
/home/newuser
[newuser@localhost newuser]$

Ahora que sabe donde se encuentra, teclee el comando siguiente:
O casi
Intente teclear.
cd home

¿Qué ha pasado? Sabe que hay un directorio nombrado home, y ha tecleado el recorrido. Entonces, ¿por qué visualiza no such file or directory ?
Significa que su recorrido está incompleto.
Intentelo ahora tecleando:
cd /home

Ahora ha cambiado los directorios de forma adecuada y se ha desplazado al interior de su directorio de login en el sub-directorio llamado home.

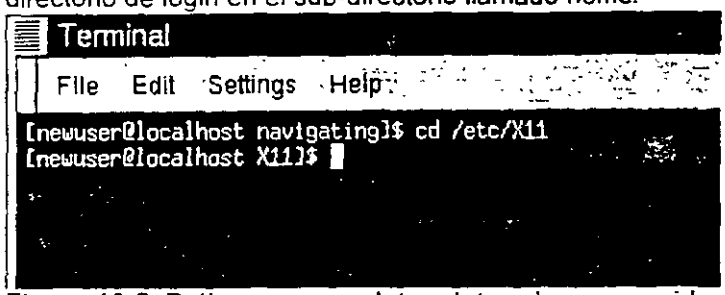

Figura 13-2. Pathname completos determinan recorridos absolutos
La diferencia está en el carácter slash (barra /)
Intentemos pensar en la razón por la que un solo carácter crea toda esta diferencia.
Cuando se ha enterado de que estaba en /home/newuser, estaba mirando el recorrido absoluto -- o el path entero del directorio de root. Esto quiere decir que en el directorio newuser se encuentra dos directorios "bajo" la raíz, que representa el primer nivel en la jerarquía de los directorios.
Por esto cuando teclea
cd /home

está diciendo al ordenador "ve al directorio root (raiz), luego entra en el directorio llamado home, que se encuentra en un nivel más bajo de root." Tiene que especificar un recorrido absoluto para ir al directorio home

Notas:

Ahora, si teclea:
cd /

conseguirá un prompt parecido a esto:
[newuser@localhost /]$

El carácter / le indica que se encuentra en el directorio root. Cuando está en root, no puede subir más en la jerarquía de su sistema (el mismo concepto vale para DOS/Windows).
Para volver atrás en su directorio de login desde el directorio de root, utilizando el recorrido absoluto, teclee:
cd /home/newuser

Está en casa.
Usar el recorrido absoluto sólo es una manera de desplazarse entre los varios directorios. Un método alternativo de moverse de un directorio a otro es el de usar un recorrido relativo (como en Figura 13-3).
Volvamos en el directorio de root:
cd /

Ahora vuelva *atrás* en su directorio de login usando un recorrido relativo:
cd home/newuser

¿Ha notado que falta el carácter /? Esto es posible porque el directorio de root es la *base* del directorio home, lo cual quiere decir que el directorio home es paso atrás con respecto al directorio root. Hasta que home sea el padre del directorio newuser, estos dos directorios estarán partidos por el carácter /.
Si se encuentra en su directorio de login, puede desplazarse hacia el directorio home, tecleando
cd ..

El recorrido relativo describe el directorio en el que desea desplazarse en términos relativos con respecto a su directorio actual.

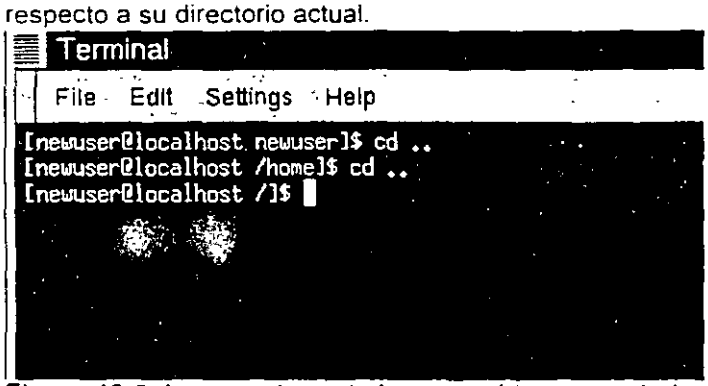

Figura 13-3. Los nombres de los recorridos son relativos a la posición actual
Cuando ha tecleado **cd ..**, ha pedido al sistema "súbete un directorio". El primer directorio superior, desde su directorio de login, es home.

Notas:

¿Qué son los *directorio padre*?

Cuando se habla de un directorio que contiene otros, puede referirse al primero llamándolo *parent directory* (directorio padre). En nuestro caso, home es el directorio superior de newuser.

Utilizando dos puntos (..) cuando utiliza el comando **cd**, le permite desplazarse en el directorio padre. Inténtelo ahora insertando un solo punto. Teclee:

cd .

¿Qué ha pasado? No mucho. Esto es porque usando un solo punto (.) está comunicando al sistema que quiere desplazarse en el directorio actual.
La diferencia entre recorrido absoluto y recorrido relativo a veces puede ser muy difícil de entender. Volviendo al ejemplo del centro comercial, si quiere seguir las direcciones utilizando recorridos absolutos, hará algo parecido a:
"Coger las llaves Montar en el coche. Arrancar el coche. Ponerse en camino Conducir hasta la esquina..."
. y siguiendo así hasta que se encuentra finalmente dentro de su tienda de zapatos preferida en el centro comercial.
Cuando utiliza un recorrido relativo, está diciendo al sistema algo como.
"La tienda se halla a un par de kilometros de aquí en el centro comercial "
Es un ejemplo un poco exagerado, sin embargo da la idea de como funciona un sistema de directorios: en el momento en que sabe a dónde ir, puede utilizar un recorrido relativo para moverse

¿El recorrido es absoluto o relativo?

Un recorrido es absoluto si el primer carácter es /, distinto, se trata de un recorrido relativo

Ahora se encuentra en el directorio home, el directorio padre de su directorio de login. Teclee

cd ..

y se encontrará en el directorio root.
Utilizando un recorrido relativo, se desplazará en el directorio de login, tecleando:

cd home/newuser

No parece muy distinto del recorrido absoluto ¿No cree? Ha visto que no aparece ninguna barra antes de home En otras palabras es como si dijera: "baja un directorio, en home, luego ve a newuser, en el directorio home."

Vuelva rápidamente a su directorio de login

Cada vez que desea volver rapidamente a su directorio de login, teclee **cd** y pulse **Enter** y, no dependiendo del sitio en que se encuentra en su sistema, acabará en su directorio de login.

Esta no quiere ser nada más que una demostración.
Desde su directorio de login, teclee

cd ../../etc/X11

Notas:

Ahora se encuentra en el directorio X11, que contiene los directorios y los ficheros de configuración del sistema X Window.

Nota

Puede siempre teclear **pwd** para saber donde se encuentra. Y puede volver a su directorio de login con el comando **cd**.

Eche un vistazo al último comando que ha insertado **cd**. Lo que está diciendo a su sistema es, "sube al directorio superior, luego al directorio inmediatamente superior (que corresponde al directorio root), luego vaya al directorio etc y entonces pase al directorio X11."
Utilizando un recorrido absoluto, puede llegar rápidamente al directorio X11. Teclee:

cd /etc/X11

¡Hecho!

Saber donde está

Compruebe cuál es el directorio actual en el que se encuentra antes de moverse con un recorrido relativo hacia otro directorio o fichero. No tiene que preocuparse de su posición en el sistema de archivos, para desplazarse en otro directorio utilizando un recorrido absoluto.

Ahora que ha hecho un poco de práctica, mire que pasa cuando se desplaza en el directorio root

cd /root

Oops... no ha entrado en el sistema como root (superusuario), por ello no tiene los permisos necesarios para acceder a este directorio.
Impedir el acceso al directorio root o a otras cuentas usuario (o *directorios de login*) es una de las soluciones que su sistema Linux le ofrece para evitar errores casuales o acciones hechas por gente que no tiene el derecho a acceder a su sistema. Encontrará más informacion sobre las propiedades y los permisos más adelante en este capitulo.
¿Quiere de verdad acceder al sistema como superusuario? Entonces le hace falta el comando **su**
Teclee este conjunto de comandos:

```
[newuser@localhost newuser]$ su
Password:your root password
[root@localhost newuser]#cd /root
[root@localhost /root]#
```

Notas:

Introducida la clave de root, verá los cambios en el prompt que le mostrara su nueva condición de root: la cuenta de superusuario tiene delante del prompt la palabra root y al final el carácter "#" (como se muestra en la figura Figura 13-4).

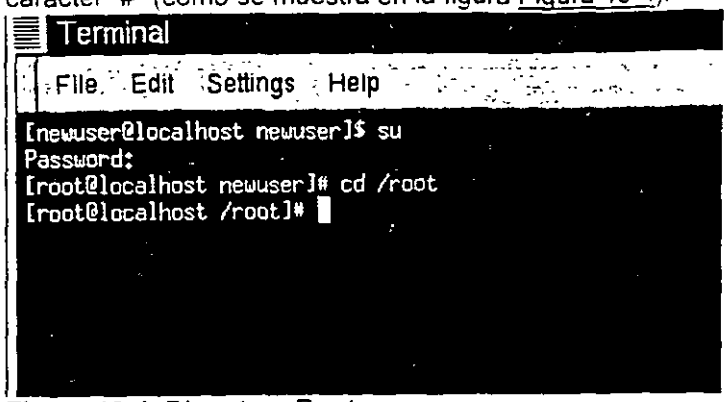

Figura 13-4. Diventare Root

Ahora puede desplazarse en el directorio de login de root, porque tiene los privilegios requeridos Para desconectarse, teclee simplemente **exit** en el prompt.

[root@localhost /root]# **exit**
exit
[newuser@localhost newuser]$

Resumen

Para cambiar los directorios utilizando un recorrido absoluto, teclee **cd** ***/directory/directory***, para cambiar directorio utilizando un recorrido relativo, teclee **cd** ***directory*** para ir al directorio que se encuentra más abajo del actual, **cd** ***directory/directory*** para ir dos directorios más abajo, etc.; Para desplazarse desde cualquier punto del sistema de archivos a su directorio de login, teclee **cd** , para ir al directorio anterior con respecto a donde se encuentra, teclee **cd** .. Utilice . para hacer referencia a su directorio actual

Notas:

Mirar con ls

Ahora que sabe como moverse, seria oportuno ver lo que hay en los directorios.
Sin embargo, compruebe antes si hay algo que comprobar en su directorio de login.
Puede empezar creando un fichero vacio. Para ello puede utilizar una utilidad llamada **touch** desde el prompt de la shell. Pruébelo tecleando:

touch foo.bar

Ahora, en su directorio de login tiene un fichero llamado foo.bar. Lo verá dentro de un par de minutos.
Creamos un nuevo directorio, usando el comando **mkdir**.
En el prompt, teclee:

mkdir tigger

Así ha creado el directorio llamado tigger en su directorio de login. Desde el directorio root, el recorrido absoluto a su nuevo directorio será /home/*yourlogin* /tigger y su directorio tigger. (Puede conseguir más informacion sobre la creación y la eliminación de un fichero en Capítulo 14)
Ahora está listo.
En el mundo DOS, el comando **dir** muestra el contenido de un directorio.
Lo mismo vale para Linux -- con algunas excepciones.
En Linux, **dir** no le enseña completamente el contenido de los directorios y no tiene la potencialidad y la flexibilidad del comando – **ls**.
En su directorio de login, por ejemplo, teclee:

dir

Y ahora en la misma ventana de Xterm teclee:

ls

Serán visualizadas las mismas informaciones (vea Figura 13-5). Verá el nuevo fichero foo.bar y el nuevo directorio tigger.

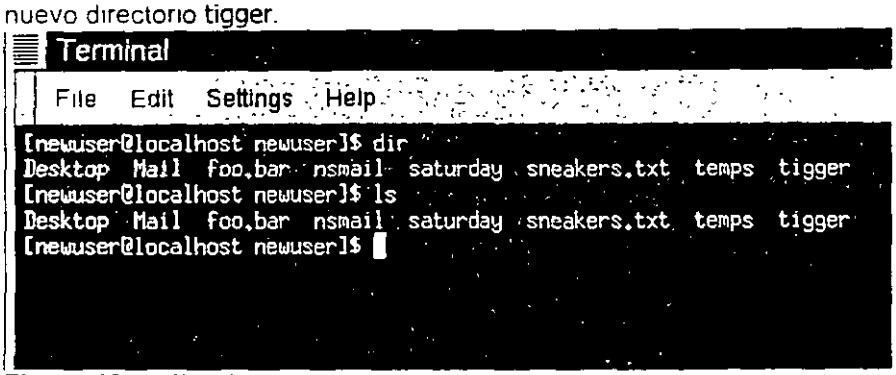

Figura 13-5. dir y ls se parecen

Sin embargo las semejanzas acaban aqui. **dir** no muestra todo el contenido de su directorio e incluso con el comando **ls**, no son visualizados todos los ficheros de su directorio. Para ver todo el contenido de su directorio tendrá que añadir una o dos opciones.

Notas:

Por ejemplo, en la misma ventana que ha utilizado para introducir los comandos dir y ls, teclee:

ls -a

Mire la diferencia, añadiendo la opción **-a**, está pidiendo una lista de *todos* los ficheros del directorio (vea Figura 13-6).
De hecho, hay muchas opciones disponibles para el comando **ls**.

Lea la página man ls

Si quiere ver todas las opciones del comando **ls**, puede leer la página man tecleando **man ls** en el prompt de la shell. Si quiere imprimir esta página, teclee **man ls | col -b | lpr** en el prompt.

¿Porqué todas estas opciones? Porque pueden ayudarle para visualizar las informaciones según sus necesidades. Por ejemplo, puede especificar como visualizar los ficheros, ver los *permisos* etc.

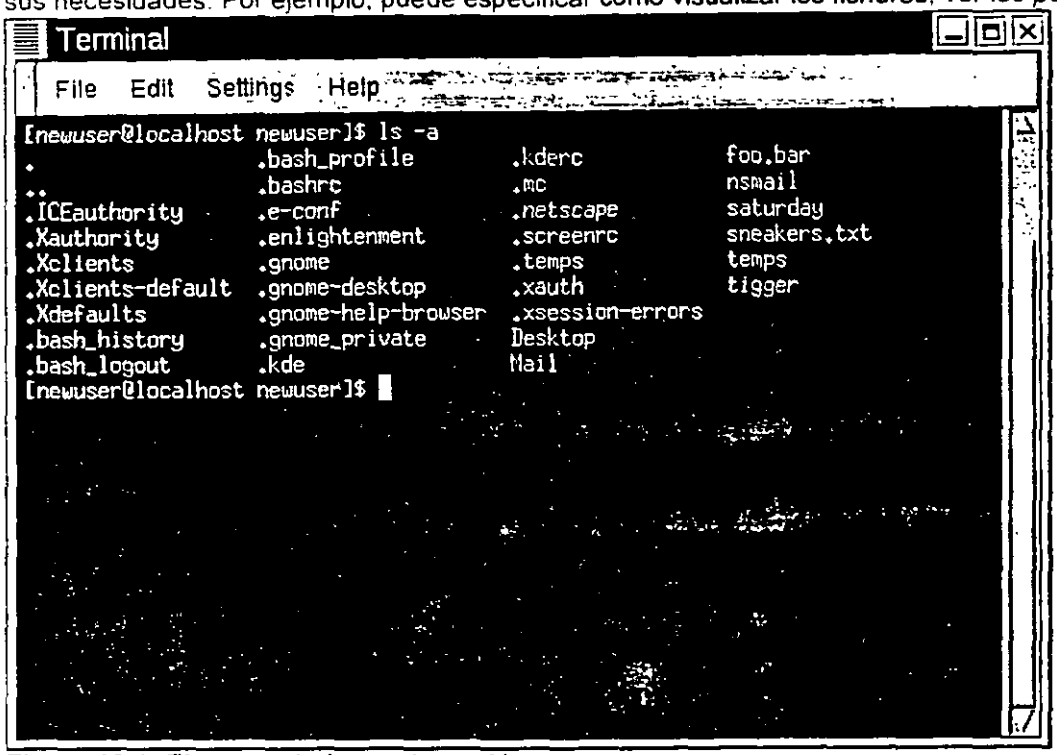

Figura 13-6. El comando ls con la opción -a

Tecleando **ls -a**, habra probablemente notado que unos ficheros empiezan por un punto. Estos han sido nombrados *ficheros ocultos* o *ficheros apuntados*.
Muchos ficheros de configuración son ficheros ocultos que definen las preferencias en los programas, en el administrador de ventanas, en la shell y mucho más. Estos ficheros están "ocultos" para impedir cualquier error accidental por parte del usuario. Además si está buscando algun fichero en el directorio, en la mayoria de los casos no está buscando estos ficheros de

Notas:

configuración y por ello el hecho de mantenerlos escondidos le ayudará a evitar crear confusión en la pantalla.
Cada vez que el nombre de un fichero empieza por un punto(.), se trata por cierto de un fichero oculto, y por lo tanto, **ls** no lo visualizará.
La visualización de todos los ficheros puede proporcionarle muchos detalles, sin embargo es posible visualizar otros, simplemente añadiendo otras opciones.
Si queremos ver el tamaño de un fichero o de un directorio, cuando ha sido creado, etc., podemos añadir la opción "largo" (**-l**) a nuestro comando **ls -a**.
Intentémoslo. Teclee:

ls -al

Han sido visualizados más detalles. Puede incluso ver la fecha de creación del fichero, su tamaño, las propiedades, los permisos y más.
No tiene que preocuparse necesariamente del directorio donde se encuentran los ficheros para visualizarlos.
Veamos el contenido del directorio /etc tecleando:

ls -al /etc

Aquí conseguirá mucha información sobre el contenido del directorio /etc.
Si quiere utilizar los colores de su lista, es suficiente con añadir la opción **--color**.

ls -al --color /etc

Poder visualizar los colores con la opción **--color** resulta útil en el caso en que quiera identificar fácilmente los tipos de ficheros. Por ejemplo los directorios se visualizarán en azul, los ficheros-programas en verde, etc...
Si quiere esta configuración, aquí tiene como puede visualizar los colores todas las veces que pide la lista de sus ficheros. En resumen, añadiremos una línea al fichero .bashrc en su directorio de login
El fichero .bashrc es usado por su shell cuando entra en el sistema (por ejemplo, los ficheros .bashrc se muestran en Figura 13-7).
Ahora antes de seguir...

Notas:

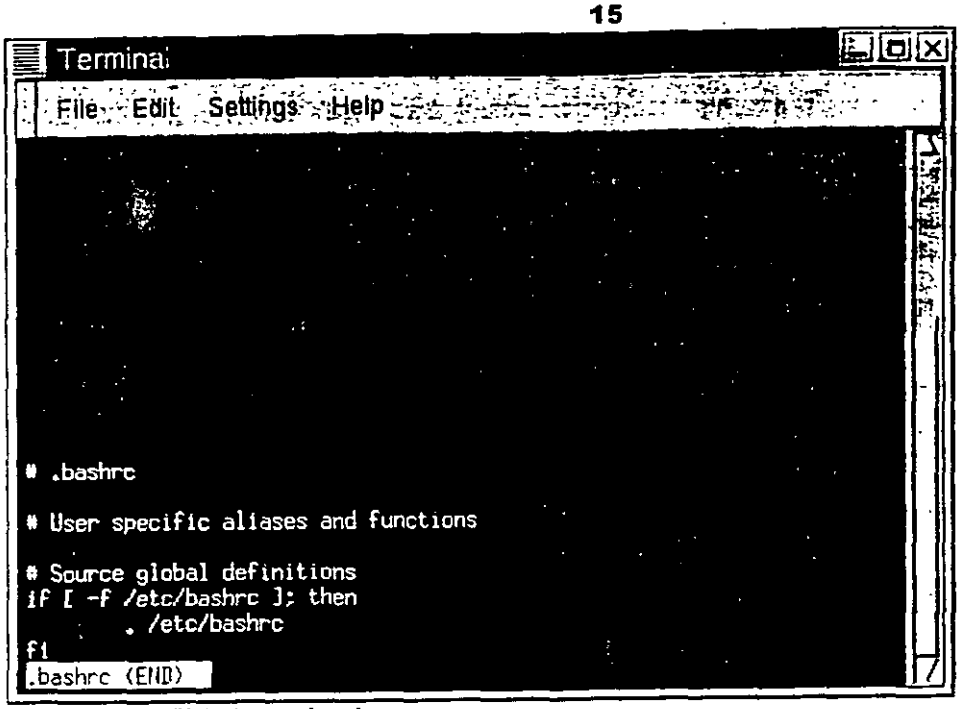

Figura 13-7. El fichero .bashrc

Recuerde que cada cambio hecho por un fichero de configuración puede provocar varios problemas si ha sido cometido algún error en la sintaxis y no posee una copia de seguridad del fichero.

Para hacer una copia de seguridad, debe estar seguro de encontrarse en su directorio de login, y en una ventana de Xterm teclee:

cd

para ir a su directorio de login. Luego copie el fichero .bashrc, en el mismo directorio, pero con otro nombre parecido a .bashrc2

cp .bashrc .bashrc2

cuando teclea este comando, está diciendo al sistema "haz una copia del fichero .bashrc y nómbralo .bashrc2 "

Ahora tiene una copia de seguridad del fichero .bashrc no modificado en su directorio de login Si comete algún error, podrá volver a copiar su fichero .bashrc tecleando:

cp .bashrc2 .bashrc

en el prompt de la shell

Si tiene que insertar este comando, estará diciendo al sistema "haz una copia del fichero .bashrc2 y renómbralo .bashrc ." El comando copia reescribirá el fichero original .bashrc -- y se metera la copia del fichero original (no modificado) .bashrc con el nombre de .bashrc2.

Notas:

Ahora que estamos listos, abriremos el fichero .bashrc con Pico, un *editor de texto* simplificado (Un editor de textos es una unidad que permite crear, modificar y grabar un fichero) En una ventana Xterm, teclee:

pico .bashrc

Verá algo parecido a lo que se muestra a continuación:

```
# .bashrc

# User specific aliases and functions

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc

fi
```

Es un fichero bastante corto. Los simbolos **(#)** son *comentarios*. Cualquier parte del texto que está después de estos simbolos, es ignorada por la shell, sin embargo estas lineas resultan muy útiles para quien esté modificando o editando el fichero.

Lleve el cursor a la linea #User specific aliases and functions y teclee:

alias ls="ls -al --color"

Este tendría que ser el resultado:

```
# .bashrc

# User specific aliases and functions

alias ls="ls -al --color"

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc

fi
```

Mire Figura 13-8 como ejemplo de Pico

Notas:

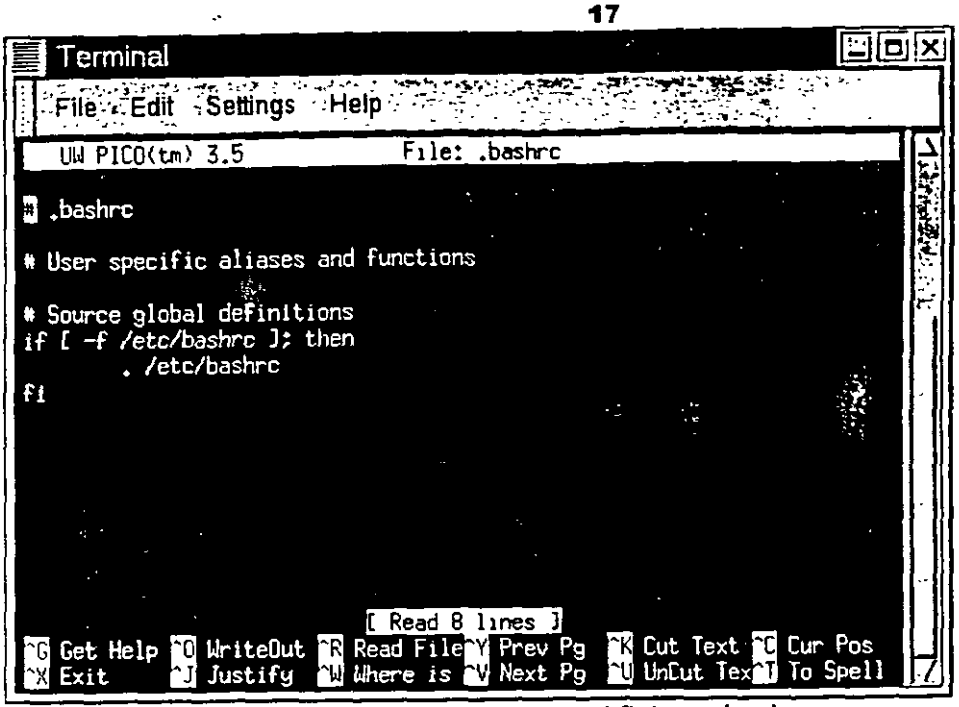

Figura 13-8. Añada un alias al comando ls en el fichero .bashrc

Controle con atención lo que ha tecleado y si está satisfecho con los cambios, salga pulsando las teclas **Ctrl** y **X**. Verá en el fondo de la pantalla de su editor un mensaje parecido a

Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES)?

Pulse **Y** para confirmar. Ahora otro mensaje le aparecerá en la parte baja:

File Name to write: .bashrc

Pulsando **Enter** grabará los cambios en el fichero .bashrc

No provocará ningun cambio hasta que no cierre la ventana Xterm y abra otra nueva Una vez que haya hecho esto podrá visualizar sus cambios.

Aquí tiene una lista de opciones comunes del comando **ls** Recuerde que puede mirar la lista completa de las opciones en la página del man **ls** (**man ls**).

- **-a** -- all. Muestra todos los ficheros en el directorio, incluidos los ficheros ocultos (.filename) Los puntos dobles .. y el punto sólo . al principio de su lista hace referencia al directorio superior y al directorio actual

- **-l** -- long. Muestra los detalles sobre los contenidos, incluidos los permisos (los modos), el propietario, el grupo, el tamaño, la fecha de creación, los enlaces con otros ficheros en el sistema y las especificaciones que nos refieren a qué ficheros hacen referencia

- **-F** -- file type. Añade un símbolo al final de cada lista. Estos símbolos incluyen / para indicar un directorio, @ para indicar un enlace simbólico hacia otro fichero, * para indicar un fichero ejecutable.

Notas:

- **-r** – reverse. Muestra el contenido de un directorio desde el principio hasta el final.
- **-R** – recursive. La opción iterativa muestra el contenido de todos los directorios (bajo la actual) de forma iterativa.
- **-S** – size. Ordena los ficheros basándose en su tamaño.

Más adelante en este capítulo, cuando se hable de las *pipe* y del *I/O redirection*, descubrirá que hay otras maneras de ver el contenido de un directorio.

Resumen

Para ver el contenido de un directorio, teclee **ls** en el prompt de la shell; tecleando **ls -a** aparecerá el contenido del directorio actual; tecleando **ls -a --color** se visualizaran los ficheros y los directorios con un color distinto.

El sistema de archivos

Cada sistema operativo tiene un método para ordenar los ficheros y los directorios de forma que puede tener en consideración las partes añadidas, las partes modificadas y otras variaciones.
En Linux, cada fichero se memoriza en los directorios o en los sub-directorios con un nombre unívoco.
Puede imaginar el sistema de archivos como la estructura de un árbol, en la que distintos directorios se parten. Estos directorios pueden contener – o ser superiores de – otros directorios que pueden contener ficheros y otros directorios en su interior.
No hay árboles sin raíces y esto vale también para los sistemas de archivos de Linux. No importa cuanto estén lejos las ramas, todo esta pegado a la raíz que se suele representar con un solo slash (/).
Podríamos confundirnos al leer todas estas referencias a root - la cuenta de root, el directorio de login de la cuenta de root, el directorio de root (raiz) /. Vamos a ver. la login de root, que pertenece al administrador del sistema, es muy importante para que todo el sistema funcione como un unico conjunto como la raiz del sistema (/).

¿Qué es *FHS*?

Existen varias distribuciones de Linux, y su sistema Red Hat Linux es compatible con ellas. Esto se hace posible gracias al *Filesystem Hierarchy Standard* (conocido también como FHS). Estas líneas guía tienen la tarea de recordarle dónde se guardan los ficheros y los programas de sistema de todos los sistemas Linux.

Para más información sobre FHS, vea el capítulo sobre la administración del sistema contenido en la Guía de Referencia de Red Hat Linux Puede además visitar el sitio web FHS a la dirección http://www.pathname.com/fhs

Puesto que se haya conectado al sistema como usuario normal -- para no cometer errores peligrosos en el sistema -- podemos echar un vistazo alrededor.
El primer paso en esta vuelta es el directorio de root que le permite tener una visión general.
En el prompt de la shell teclee:

Notas:

cd /

Verá un prompt parecido a esto:
[newuser@localhost /]$

Ahora veremos los directorios que parten de root tecleando:
ls

No se ve mucho ¿por qué? Esta solo es la punta del iceberg. Estos son los directorios principales en los cuales se reparten los demás directorios....

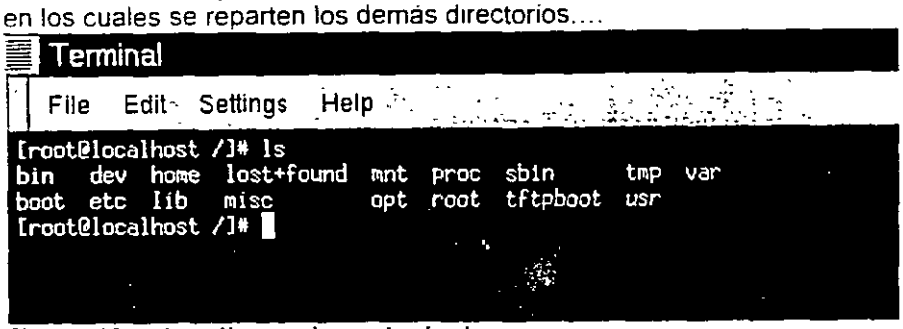

Figura 13-9. los directorios principales

Estos son algunos de los directorios que tomaremos en consideración:
etc lib sbin
usr var

Examinemos el directorio /etc.
[newuser@localhost /]$ **cd etc**
[newuser@localhost /etc]$ **ls**

Aquí además de distintos ficheros y directorios, encontraremos los *ficheros de configuración*, que sirven para que los programas funcionen en su sistema, etc.
Entre los directorios presentes, encontrará /etc/X11, que contiene los directorios y los ficheros de configuración para el sistema X Window.
En el directorio /etc/skel, encontrara el fichero de los usuarios *skeleton*, usado en la creación de nuevos usuarios Sirve para crear cuentas normales.
¿Qué quiere decir esto? Cuando ha entrado en el sistema como root, una de las primeras tareas ha sido la de crear una cuenta para usted mismo
Cuando su cuenta ha sido creada, se han copiado unos ficheros de /etc/skel y se han introducido en la nueva cuenta Estos ficheros le ayudarán a crear la configuración básica del usuario (skeleton... flesh.)
Veamos ahora el directorio /usr Desde donde se encuentra, /etc/skel, puede teclear
[newuser@localhost /skel]$ **cd /usr**

Notas:

[newuser@localhost /usr]$ **ls**

El directorio /usr, contiene algunos de los fichero más importantes para su sistema (vea Figura 13-10).

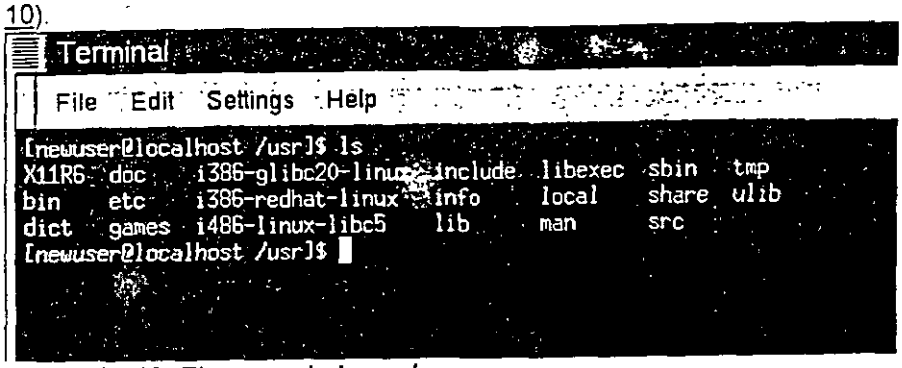

Figura 13-10. El comando ls en /usr

Dentro de /usr/man encontrará las páginas man; más documentación podra encontrarla en el directorio /usr/doc y en /usr/info.

dentro de /usr/X11R6, encontrará los ficheros relativos al sistema X Window, incluidos los ficheros de configuración y de documentación.

Notas:

Podríamos pensar en algo más culto oyendo la palabra "librerias", de hecho dentro de /usr/lib encontrará unos ficheros que componen las *librerias* del sistema. En este contexto, las librerias son ficheros que contienen instrucciones comunes que pueden ser compartidas por muchos programas.

Red Hat Linux utiliza RPM (la tecnología *R*PM *P*ackage *M*anager) para la instalación y la actualización. Utilizando RPM, es posible trabajar tanto en la shell, como con Gnome-RPM, y los dos son métodos prácticos para gestionar el software.

(Para más información sobre la utilización de Gnome-RPM, vea Capítulo 10, o lea el capítulo relativo en la Guía de Referencia de Red Hat Linux).

Por supuesto, cuando tenga mayor confianza con su sistema, podrá instalar programas también en el formato RPM. Para evitar cualquier conflicto con otros paquetes le aconsejamos instalar el software en formato RPM en el directorio /usr/local.

"Limpiar" la ventana

Después de cada comando **ls** en una ventana Xterm, están disponibles un monton de informaciones que rellenerán la pantalla. Podria cerrar su ventana y abrir otra, pero no es esta la forma mejor de hacerlo.

Teclee

clear

en el prompt de la shell El comando **clear** no tiene ningún mensaje. limpie la pantalla del terminal

Alguna vez, podria abrir de forma accidental un programa, un fichero de datos o algun otro fichero no de texto en una ventana terminal. Cuando cerrara el fichero podría pasar que lo que teclea no corresponda a la salida en la pantalla.

En estos casos, hay que insertar:

reset

para que la ventana vuelva a su configuarción básica

Resumen

Para limpiar la consola de una ventana Xterm, teclee el comando **clear**; para volver a la configuración inicial de una ventana Xterm, teclee **reset**

Utilice cat

Hay varias utilidades que pueden ayudarle para ordenar las listas, unirlas y a la vez mostrarle una pequeña parte de las potencialidades escondidas de su sistema Red Hat Linux.

La utilidad **cat**, abreviatura de "encadenadas", sirve para unir las cadenas.

Pero **cat** puede dar una demostración de dos conceptos importantes *entrada estandar* y *salida estándar*.

La entrada estándar y la salida estandar redireccionan el input y el output (a menudo abreviados con *I/O*) Si un programa lee de la entrada estandar, leerá por defecto las entradas del teclado. Si un programa escribe en la salida estandar, escribirá por defecto los datos en la pantalla

Lance **cat** para ver cómo funciona. En el prompt de la shell, teclee

Notas:

cat

El cursor se desplazará en una linea vacia. Ahora en esta línea teclee:
stop by sneaker store

y pulse la tecla **Enter**. Después aparecerá en su pantalla:
[newuser@localhost newuser]$ **cat**
stop by sneaker store
stop by sneaker store

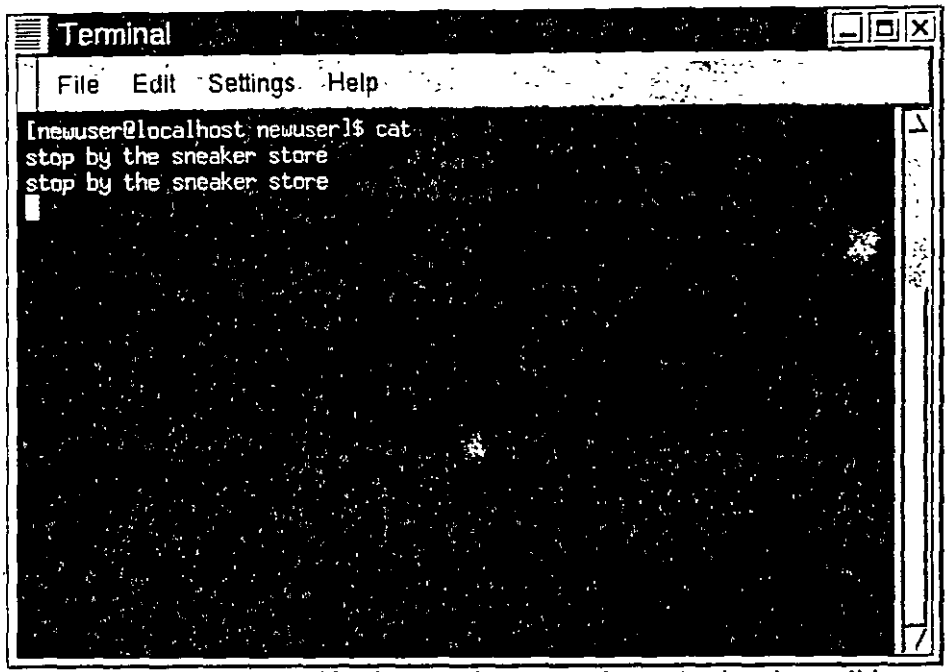

Figura 13-11. Demostración del uso de cat con las entrada y las salidas estándar. output
Para salir de **cat** mueva el cursor en la linea vacía pulsando la tecla **Enter** y luego apretando la tecla **Ctrl** y a la vez **D**.
Por cierto no se trata de un ejemplo brillante, sin embargo por medio de **cat** hemos tenido la posibilidad de mostrarle los conceptos de entradas y salidas estándar. Su entrada ha sido leída por el teclado y esta ha sido redireccionada hacia el monitor (salida estándar).

Resúmen

La entrada estándar es el texto que se inserta por el teclado. La salida estándar es el lugar donde se visualizan las informaciones, como su terminal (como se muestra en Figura 13-11).

Notas:

Redireccionar la salida y la entrada de los datos

Ahora que tenemos claro que se entiende por salida y entrada estándar, es la hora de profundizar en algunos conceptos.
redireccionar significa cambiar todo aquello que la shell considera como salida estándar y entrada estándar.
Hemos utilizado **cat** para proporcionarle un ejemplo de salida y entrada estándar. Ahora utilizamos **cat** para mostrarle cómo es posible redireccionar la salida estándar.
Para redireccionar la salida estándar, usaremos el símbolo >. Poniendo > después del comando **cat** (o después de cada programa que escriba en la salida estándar) se direcciona la salida en el fichero indicado después este símbolo.
Intentémoslo. En una ventana Xterm, teclee:

[newuser@localhost newuser]**$ cat > sneakers.txt**
buy some sneakers
then go to the coffee shop
then buy some coffee

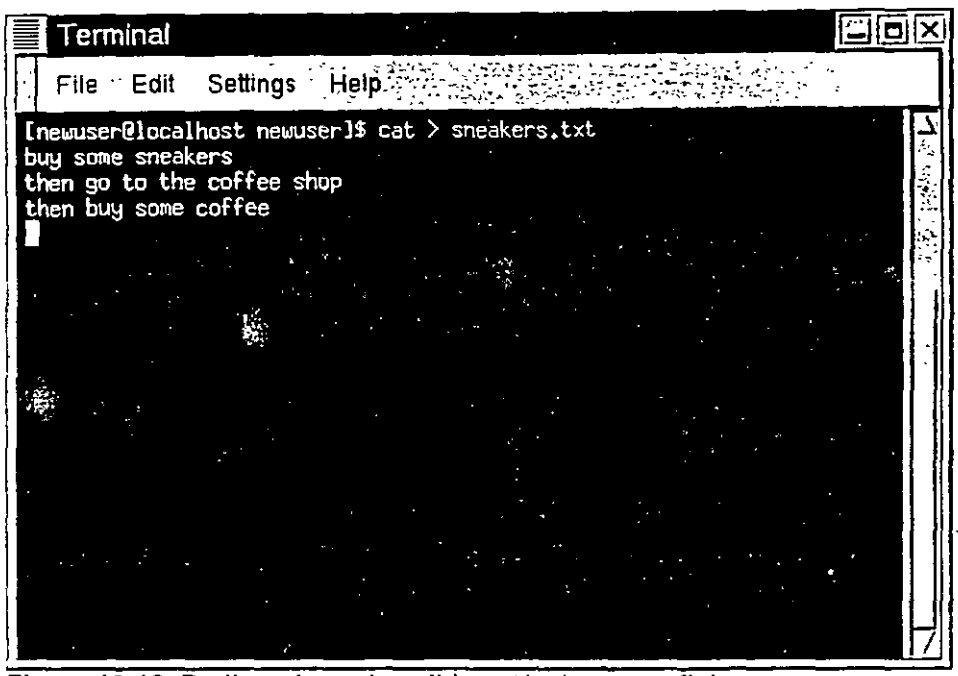

Figura 13-12. Redireccionar la salida estándar en un fichero

Ahora pulse la tecla **Enter** para ir a una línea vacía y utilice la tecla **Ctrl**-D para salir de **cat**.
¿Ha notado la diferencia (vea Figura 13-12)? No ha sido visualizado nada porque la salida estándar de cat ha sido redireccionado en el fichero llamado sneakers.txt

Notas:

Encontrará el fichero en su directorio de login (le aconsejamos usar **ls** si quiere visualizarlo). Puede utilizar **cat** para leer el fichero tecleando:

cat sneakers.txt

en el prompt.

Atención

Tenga cuidado cuando redirecciona la salida estándar en un fichero, porque ¡podría sobrescribir un fichero existente! Compruebe que el nombre del fichero que está creando no sea igual a uno que ya existe, si no quiere sobrescribirlo.

Redireccionemos la salida hacia otro fichero y llamémoslo home.txt.

[newuser@localhost newuser]$ **cat > home.txt**
bring the coffee home
take off shoes
put on sneakers
make some coffee
relax!

Ahora en una línea vacía, utilice las teclas **Ctrl-D** para salir de **cat**.
Podemos visualizar otra vez el fichero tecleando:

cat home.txt

en el prompt.
Utilice **cat** para unir home.txt con sneakers.txt y redireccionemos la salida de los dos ficheros en otro fichero nuevo que llamaremos saturday (encontrará un ejemplo de ello en Figura 13-13)

[newuser@localhost newuser]$ **cat sneakers.txt home.txt > saturday**

¡Hecho!

Notas:

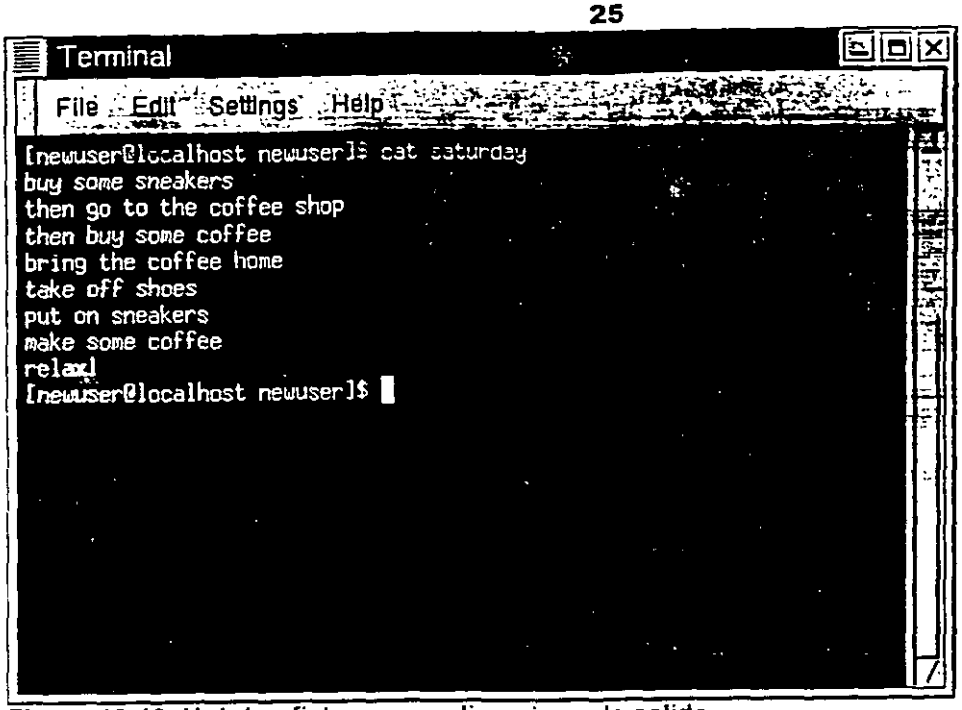

Figura 13-13. Unir los ficheros y redireccionar la salida

Para ver el resultado, teclee:

[newuser@localhost newuser]$ **cat saturday**

debería ver algo parecido a lo que sigue:

[newuser @localhost newuser]$ **cat saturday**
buy some sneakers
then go to the coffee shop
tnen buy some coffee
bring the coffee home
take off shoes
put on sneakers
make some coffee
relax!
[newuser @localhost newuser]$

Habra notado que **cat** ha añadido home.txt donde termina sneakers.txt.

- **Unir ficheros con cat**

  Crear y unir ficheros con **cat** puede ser una buena solución alternativa para un editor de textos como Pico.

- **Resumen**

Notas:

Usando el símbolo > puede enviar la salida de un fichero a un terminal. La utilidad **cat** puede ser utilizada para redireccionar la salida o para unirlo a otro fichero.

Adjuntar la salida estándar

Hay la posibilidad de redireccionar la salida añadiendo informaciones al final de un fichero existente. De forma muy parecida a la utilización del símbolo >>, puede utilizar la shell para añadir las informaciones al final de un fichero.
Cuando utiliza >, está *añadiendo* informaciones.
Le presentamos un ejemplo práctico para aclarar este concepto. En este ejemplo unamos dos ficheros creados anteriormente -- sneakers.txt y home.txt -- utilizando el símbolo para adjuntar la salida Queremos añadir las informaciones presentes en home.txt uniéndolas a las informaciones ya presentes en sneakers.txt. Es suficiente teclear:

cat home.txt >> sneakers.txt

Para visualizar el contenido del fichero, ejecute el comando:

cat sneakers.txt

De esta forma hemos añadido la salida del fichero home.txt.
Tecleando este comando hemos dicho al sistema "añada la salida del fichero home.txt al fichero sneakers.txt."
Añadiendo la salida directamente, hemos ahorrado uno o dos pasos (y un poco de espacio en el disco) utilizando unos ficheros que ya existían en memoria, en vez de crear uno nuevo.
Si controla los ficheros sneakers.txt y saturday, notará que son iguales. Para efectuar esta comparación teclee:

cat sneakers.txt; cat saturday

Los contenidos de los dos ficheros se visualizan - antes sneakers.txt, y luego saturday (como se muestra en Figura 13-14).

¡Cuidado!

Recuerde que cuando quiere añ la salida a un fichero, tiene que teclear los símbolos >> . En caso contrario ¡vuelva a crear el fichero original con lo que quería añadir!

Notas:

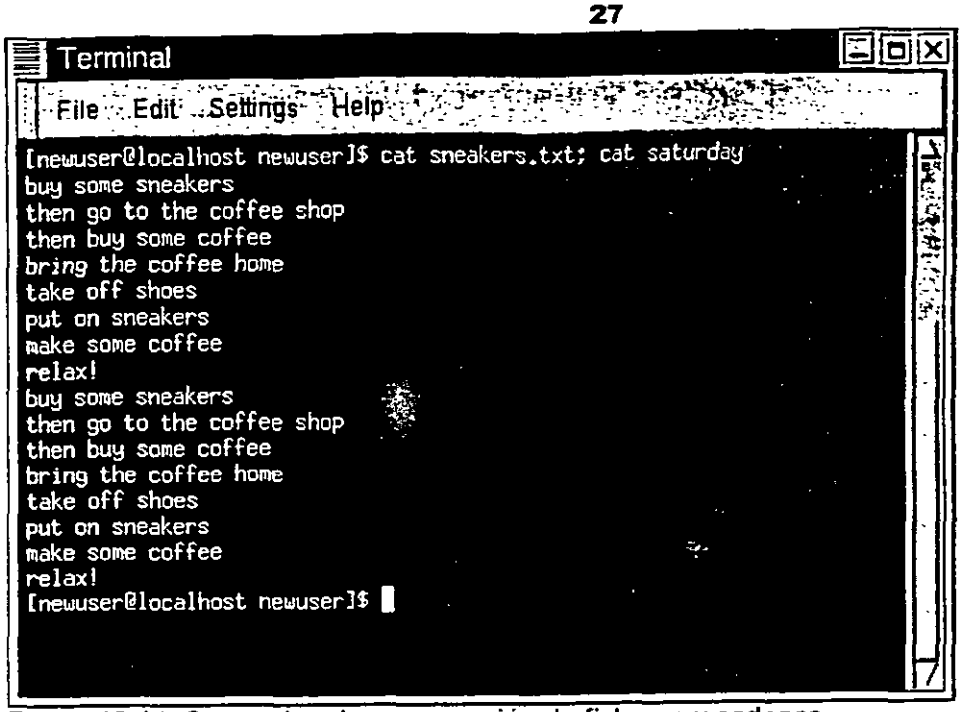

Figura 13-14. Comandos de comparación de ficheros y cadenas

(Si está interesado en conocer el uso del punto y coma en el último comando, le aconsejamos que lea el capítulo siguiente.)

Resumen

Para añ la salida a un fichero, utilice los simbolos de mayor (>>) Por ejemplo. **cat addthisfile >> tothisfile**.

Redireccionar la entrada estándar

No sólo puede redireccionar,la salida estándar, sino también la entrada estándar.
Vamos a ver como funciona:
Cuando utiliza el simbolo < para redireccionar la entrada estándar, está diciendo a la shell que un fichero debe ser utilizado como entrada para un comando.
Podemos utilizar un fichero anteriormente creado para explicar este concepto Teclee:

cat < sneakers.txt

Utilizando el simbolo menos (<) para separar el comando **cat** de un fichero, la salida de sneakers.txt es leida por **cat**.

Notas:

Pipe

¡No se preocupe! no vamos a hablarle de tuberias. En el mundo Linux, una *pipe* relaciona la salida estándar de un comando con la entrada estándar de otro comando.
Volvemos un poco atrás, al comando **ls**. Hay varias opciones disponibles con el comando **ls**, pero ¿qué pasa con la visualización del contenido de un directorio si es demasiado rápida para verla?
Vamos a ver el contenido del directorio /etc.

ls -al /etc

¿Cómo podemos visualizar tranquilamente la salida antes que desaparezca de la pantalla?
Es posible enviar la salida a la utilidad **less**. Conocido como *paginador*, **less**, (como **more**) le permite ver las informaciones en una página (o en una pantalla) a la vez.
Para enviar la salida estándar de un comando (o programa) a la entrada estándar de otro, se utiliza el carácter | (como se muestra en la figura Figura 13-15).

ls -al /etc | less

En esta manera verá el contenido del directorio partido en dos pantallas. Para acceder a la pantalla siguiente, pulse **Space**, para volver a la pantalla anterior, pulse **B**; para salir, pulse la tecla **Q**.

Como leer los mensajes de arranque (startup)

¿Quiere leer los mensajes de arranque más cuidadosamente? En el prompt de la shell, teclee **dmesg | less** Será capaz de leer todo el fichero una pantalla cada vez. Para seguir adelante, pulse **Space**; para salir, pulse **Q**.

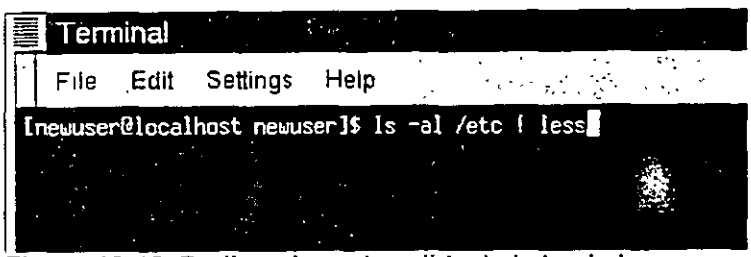

Figura 13-15. Redireccionar la salida de ls hacia less

De verdad hemos utilizado los comandos de redireccionamiento, antes de explicar que cosa son
En las páginas anteriores, hablando de las páginas man, hemos utilizado el comando pipe para imprimir en papel las páginas del comando ls:

man ls | col -b | lpr

Así está redireccionando la salida de **man ls** a un filtro llamado **col** con una opción **-b** para formatear el texto para la impresora, luego hemos redireccionado la salida a la impresora con el comando **lpr**.

Notas:

Resumen

El comando pipe permite redireccionar la salida de un comando hacia la entrada de otro comando. Por ejemplo: **ls -al /etc | more** envía la salida del comando **ls** al comando **more** para permitir su lectura.

Insertar más comandos a la vez

Linux le permite insertar más comandos en la misma línea. El único vínculo es el de separar los comandos por medio de un punto y coma (puede encontrar un ejemplo en la figura Figura 13-14) ¿Quiere controlar por cuánto tiempo ha estado conectado a la red con Netscape? Es suficiente con unir el comando fecha con el comando Netscape.

date; netscape; date

Recuerde que estos comandos distinguen mayúsculas de minúsculas, por ello el comando para lanzar Netscape tiene que ser escrito en minúsculas.
En la ventana Xterm teclee el comando siguiente:

[newuser@localhost newuser]$ **date; netscape; date**
Mon Feb 7 13:26:27 EST 2000

Verá la fecha siguiente cuando salga de Netscape. En su pantalla aparecerá lo que se muestra a continuación:

[newuser@localhost newuser]$ **date; netscape; date**
Mon Feb 7 13:26:27 EST 2000
Mon Feb 7 14:28:32 EST 2000
[newuser@localhost newuser]$

Al cierre de Netscape reaparecerá el prompt. La diferencia entre los dos resultados del comando **date** muestra por cuanto tiempo se ha utilizado la aplicación Netscape.

Propiedades y permisos

Al principio de este capítulo se ha presentado este ejemplo para la ejecución del comando **cd**.

[newuser@localhost newuser]$ **cd /root**
bash: /root: Permission denied
[newuser@localhost newuser]$

Esto es un ejemplo de las características de seguridad de Linux Linux, como UNIX, es un sistema multiusuario y los permisos para tener acceso a los ficheros presentan una solución para proteger la seguridad del sistema de cualquier daño.
Una manera de tener acceso es por medio del comando **su** de root, porque quien conoce la clave de root tiene el acceso completo al sistema

[newuser@localhost newuser]$ **su**
Password: ***your root password***

Notas:

```
[root@localhost newuser]# cd /root
[root@localhost /root]#
```

Entrar en el sistema como superusuario no es siempre la cosa mejor – por que es bastante sencillo hacer errores en importantes ficheros de configuración.
Todos los ficheros y los directorios pertenecen a la persona que los ha creado. Hemos creado el fichero sneakers.txt en nuestro directorio de login, por ello sneakers.txt nos "pertenece".
Esto quiere decir que podemos especificar quien puede leer o escribir un fichero Además en el caso de que un fichero sea ejecutable es posible especificar quien tiene el derecho a ejecutarlo.
Leer, escribir y ejecutar son tres parámetros muy importantes en permisos.
Cada usuario del sistema está incluido en un *grupo*, podemos también especificar que grupos de usuarios tienen acceso a nuestros ficheros.
Tomamos en consideración el fichero sneakers.txt con el comando **ls** utilizando la opción **-l** (long) (vea Figura 13-16).

```
[newuser@localhost newuser]$ ls -l sneakers.txt
-rw-rw-r--    1 newuser newuser     150 Mar 19 08:08 sneakers.txt
```

Con esta opción se visualizarán algunos detalles añadidos. Podemos ver quien puede leer (r) y escribir (w) el fichero, el propietario (newuser) y a qué grupo de usuarios pertenece (newuser).

su grupo por defecto

Recuerde que, por defecto, el grupo corresponde al nombre del usuario.

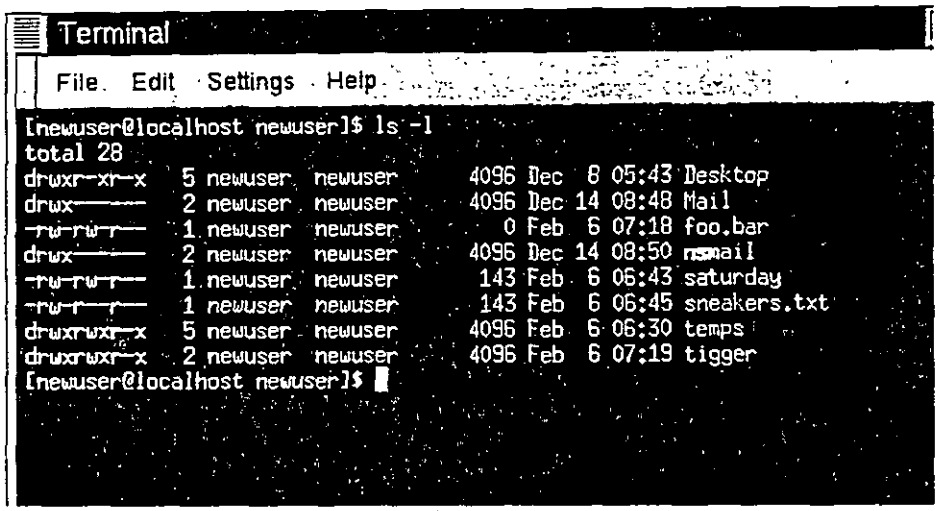

Figura 13-16. Permisos para sneakers.txt

Entre el nombre del grupo y el nombre del fichero hay informaciones relativas al tamaño del fichero, a la fecha y a la hora de creación
¿Qué sentido tienen las letras y los guiones que se encuentran en la parte izquierda? Es más sencillo explicar el sentido utilizando un ejemplo.

```
-rw-rw-r--
```

Notas:

Hay 10 columnas. La primera columna representa el tipo de fichero. Las restantes 9 son repartidas en grupos de 3 con 3 distintas clases de permisos.
Estos tres grupos se refieren respectivamente al propietario del fichero, al grupo y "al resto del mundo", o sea todos los demás usuarios y grupos además del propietario (newuser) y del grupo (newuser).
En detalle:

```
-  (rw-)  (rw-)  (r--)   1 newuser newuser
|    |      |     |
type owner  group others
```

El primer elemento, que especifica el tipo de fichero, puede tener uno entre los siguientes valores:

- d -- el fichero es un directorio
- - -- un fichero regular (excepto directorios y enlaces)
- l -- un enlace simbólico a un fichero

Después del primer caracter, en los tres grupos que siguen es posible especificar uno entre los siguientes valores:o

- r -- indica que el fichero es accesible para la lectura
- w --indica que el fichero es accesible para escribir
- x --indica que el fichero es un ejecutable (si es un programa)

Cuando aparece un guión en uno de estos campos, quiere decir que un permiso entre los que vimos no ha sido concedido.
Mire otra vez la primera columna del fichero sneakers.txt y lea sus permisos. (vea Figura 13-17)

```
[newuser@localhost newuser]$ ls -l sneakers.txt
-rw-rw-r--    1 newuser newuser     150 Mar 19 08:08 sneakers.txt
[newuser@localhost newuser]$
```

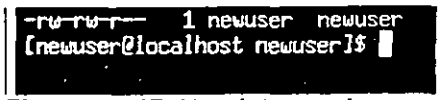

Figura 13-17. Un vistazo a los permisos

El propietario del fichero, newuser, tiene los permisos para escribir y leer el fichero; el fichero no es un programa, entonces newuser no tiene el permiso de ejecutarlo. El grupo, newuser, tiene los permisos para escribir y leer sneakers.txt. Para la notación relativa al permiso de ejecución , no hay aquí el permiso de ejecución para el grupo newuser.
Los últimos tres caracteres se refieren a los usuarios que no son newuser y tampoco pertenecen al grupo newuser Esos usuarios pueden leer el fichero, pero no pueden escribirlo o ejecutarlo.
Podemos utilizar el comando **chmod** para cambiar los permisos de los ficheros.
En el fichero sneakers.txt podemos cambiar los permisos por medio del comando **chmod**
En el fichero original están presentes los siguientes permisos:

```
-rw-rw-r--    1 newuser newuser     150 Mar 19 08:08 sneakers.txt
```

Notas:

Puesto que usted es el propietario del fichero – o ha entrado en el sistema como superusuario -- podemos cambiar los permisos utilizando todas las combinaciones.
Por el momento, el propietario (usted) y los usuarios del grupo (newuser) pueden leer y escribir el fichero.
Quien esté fuera de nuestro grupo, sólo puede leer el fichero (r--).

Los permisos son necesarios

Recuerde que los permisos de los ficheros son muy importantes para la seguridad. Cada vez permite a todo el mundo leer, escribir o ejecutar un fichero, puede ser que arriesga la seguridad del sistema. Como regla general, le aconsejamos evitar lo más posible los permisos de leer y escribir a los demás usuarios.

En nuestro ejemplo, suponga que quiere conceder a un grupo el permiso de escritura en un fichero, de forma que puedan leerlo, escribir en ello y grabarlo. Esto quiere decir que tendrá que cambiar los permisos en sección "otros".
Puesto que somos los proprietarios del fichero, no tenemos que utilizar el comando su para cambiar los permisos. Es bastante teclear:

ls -l sneakers.txt

wche visualiza las informaciones sobre el fichero:

-rw-rw-r-- 1 newuser newuser 150 Mar 19 08:08 sneakers.txt

Ahora inserte:

chmod o+w sneakers.txt

Para controlár los resultados, podemos nuevamente elencar los detalles relativos a los ficheros

-rw-rw-rw- 1 newuser newuser 150 Mar 19 08:08 sneakers.txt

Ahora, cualquiera puede leer o escribir el fichero (vea Figura 13-18)

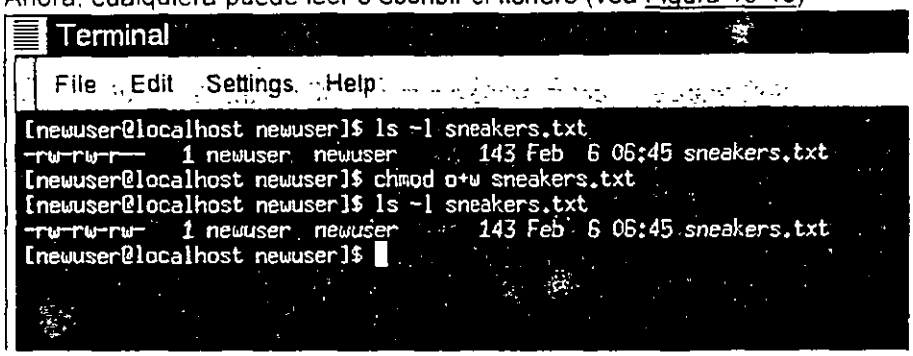

Figura 13-18. Cambiar los permisos para sneakers.txt

Cuando ha tecleado **o+w**, ha elegido "añadir" los permisos de escritura para el fichero sneakers.txt.
Si quiere eliminar todos los derechos de acceso para el fichero sneakers.txt puede utilizar el comando **chmod** para eliminar los permisos de escribir y leer en esta manera:

chmod go-rw sneakers.txt

Notas:

esto será el resultado:

```
-rw-------  1 newuser newuser  150 Mar 19 08:08 sneakers.txt
```

La opción **go-rw**, significa que "para el grupo y el resto del mundo, quiere quitar los permisos de lectura y escritura del fichero sneakers.txt."
Estos pasos le parecerán escritos en estilo estenográfico si quiere cambiar los permisos utilizando **chmod**, de hecho lo que tiene que saber son unos cuantos simbolos y letras que funcionan con el comando **chmod**.
Aquí tiene una lista de las opciones:

Identidad

u -- es el usuario proprietario del fichero

g -- es el grupo al que el usuario pertenece

o -- el resto del mundo (ni el proprietario, ni su grupo)

a -- todo el mundo (**u**, **g**, y **o**)

Permisos

r -- acceso de sola lectura

w -- acceso de sola escritura

x -- acceso de ejecución

Notas:

Acciones

+ – añadir los permisos

- – quitar los permisos

= – hacerlo como único permiso

¿Lo intentamos? Quite los permisos de sneakers.txt – para todos los usuarios.
chmod a-rw sneakers.txt

Ahora compruebe si es posible leer el fichero:
[newuser@localhost newuser]$ **cat sneakers.txt**
cat: sneakers.txt: Permission denied
[newuser@localhost newuser]$

Funciona; no tenemos acceso al fichero. Puesto que el fichero nos pertenece, podemos cambiar los permisos según nuestros deseos. (Vea Figura 13-19)

[newuser@localhost newuser]$ **chmod u+rw sneakers.txt**
[newuser@localhost newuser]$ **cat sneakers.txt**
buy some sneakers
then go to the coffee shop
then buy some coffee
bring the coffee home
take off shoes
put on sneakers
make some coffee
relax!
[newuser@localhost newuser]$

Notas:

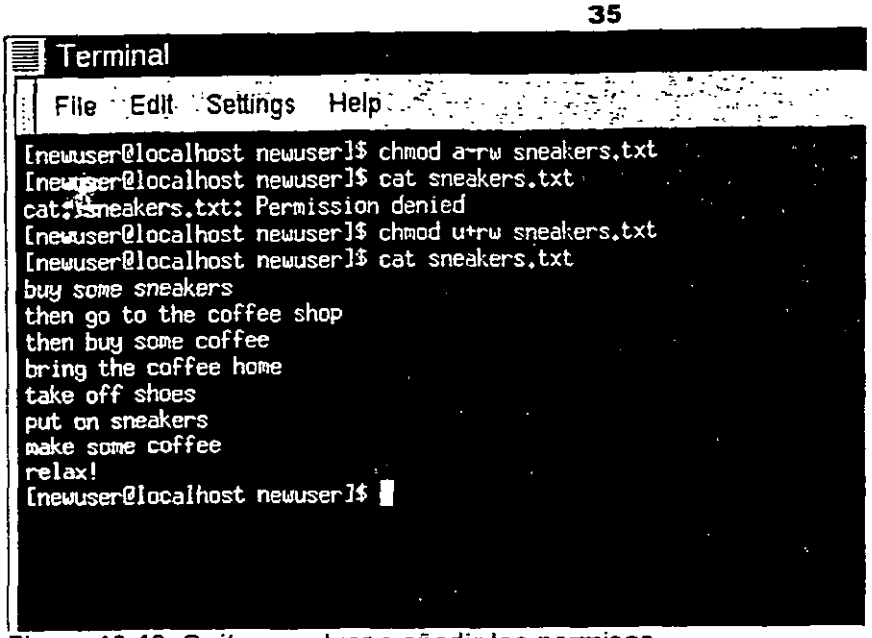

Figura 13-19. Quitar y volver a añadir los permisos

Aquí tiene unos ejemplos de parametros que pueden utilizarse con **chmod**.

- **g+w** -- Añadir el acceso de escritura para un grupo
- **o-rwx** -- quitar los permisos para los demás
- **u+x** -- permitir al propietario del fichero su ejecución
- **a+rw** -- permitir a cualquiera leer y escribir en el fichero
- **ug+r** -- permitir al propietario o al grupo leer el fichero file
- **g=rx** -- conceder al grupo la lectura y la ejecución (sin escribir)

Añadiendo la opción **-R**, podemos cambiar los permisos para el árbol entero del directorio donde nos encontramos

De todas formas hay una limitación, puesto que no podemos ejecutar un directorio como si fuera una aplicación Cuando añade o quita los permisos de ejecución para un directorio, de verdad estamos concediendo (o quitando) los permisos de búsqueda en el interior de este directorio

Para permitir a cualquiera tener el acceso de lectura y escritura al fichero tigger en su directorio de login, puede teclear

chmod -R a+rw tigger

Si no concede a otros el poder tener los permisos de ejecución para tigger, no sirve de mucho limitar el acceso en lectura y escritura, porque nadie puede entrar en el directorio - a menos que no conozca el exacto nombre del fichero

Por ejemplo, teclee

chmod a-x tigger

Notas:

para quitar todos los derechos de ejecución a todos los usuarios.
Aquí tiene lo que pasa cuando intenta acceder al directorio tigger:

```
[newuser@localhost newuser]$ cd tigger
bash: tigger: Permission denied
[newuser@localhost newuser]$
```

Configurar nuestros derechos de acceso y los de los grupos.

chmod ug+x tigger

Ahora, si controla su trabajo con el comando **ls -dl** verá que los demás no tienen acceso a tigger.

Jugar con los números en chmod

¿Se acuerda de cuando hablábamos del modo abreviado para utilizar el comando chmod? Aquí tiene otro camino para cambiar los permisos; en principio podrá parecerle algo más compleja - especialmente si la matemática no es su punto fuerte.
Volvamos a los permisos originales de sneakers.txt.

```
-rw-rw-r--   1 newuser newuser    150 Mar 19 08:08 sneakers.txt
```

Cada permiso puede ser representado por un valor numérico:

- r = 4
- w = 2
- x = 1
- - = 0

Si se suman estos valores, el resultado es utilizado para especificar los permisos..
Para el fichero sneakers.txt, aquí tiene un ejemplo de los permisos codificados en números

```
- (rw-)  (rw-) (r--)
   |      |     |
  4+2+0  4+2+0 4+0+0
```

El total para los usuarios es seis, el total para los grupos es seis y el total para para los demás es cuatro. Entonces conseguirá **664**.
Si quiere cambiar el fichero sneakers.txt también si los usuarios que pertenecen a nuestro mismo grupo no tienen el acceso para escribir, pero pueden leerlo (como se muestra en Figura 13-20), tiene que prohibir el acceso quitando un dos al conjunto de números
Los valores numéricos se volverán así seis, cuatro y cuatro -- o 644. .
Entonces podemos teclear

chmod 644 sneakers.txt

Para controlar los cambios, teclee el comando **ls -l sneakers.txt**

Notas:

-rw-r--r-- 1 newuser newuser 150 Mar 19 08:08 sneakers.txt

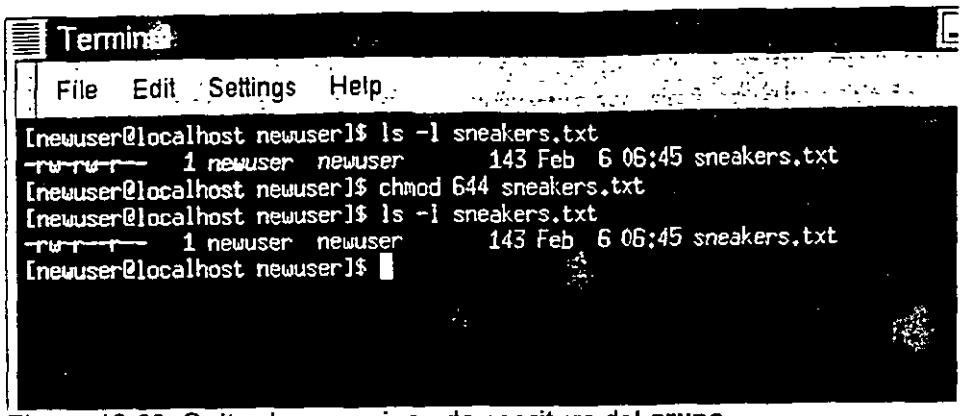

Figura 13-20. Quitar los permisos de escritura del grupo

Ningun usuario, además del propietario, puede escribir el fichero sneakers.txt Para reestablecer los derechos de escritura del grupo en el fichero, puede añadir el valor w (2) al segundo grupo de permisos.

chmod 664 sneakers.txt

Ponga cuidado con los permisos 666 y 777

Poner los permisos a 666 o 777 dará la posibilidad a todos de leer o escribir un fichero o un directorio. Esta configuración pueden interferir con los ficheros "sensibles a las mayusculas", por ello no es buena elección utilizarlos normalmente.

Aquí tiene una lista de algunos valores numéricos y los significados relativos.

- -rw------- (600) -- Sólo el usuario tiene el derecho de leer y escribir
- -rw-r--r-- (644) -- Sólo el usuario tiene los permisos de leer y escribir; el grupo y los demás sólo pueden leer
- -rwx------ (700) -- Sólo el usuario tiene los derechos de leer, escribir y ejecutar el fichero.
- -rwxr-xr-x (755) -- El usuario tiene los derechos de leer, escribir y ejecutar; el grupo y los demás sólo pueden leer y ejecutar
- -rwx--x--x (711) -- El usuario tiene los derechos de lectura, escritura y ejecución, el grupo y los demás sólo pueden ejecutar.
- -rw-rw-rw- (666) -- Cada uno puede leer y escribir en el fichero ¡No es una buena eleccion!
- -rwxrwxrwx (777) -- Cada uno puede leer, escribir y ejecutar ¡Otra mala elección!

Aquí tiene un conjunto de valores para los directorios

Notas:

- drwx------ (700) -- Sólo el usuario puede leer y escribir en este directorio.
- drwxr-xr-x (755) -- Cualquiera puede leer el directorio, pero su contenido sólo puede ser cambiado por el usuario.

Resumen

Puede cambiar los permisos con el comando **chmod** utilizando letras y números. Teclee **chmod *"permisos"* fichero** para cambiar los permisos de un fichero o de un directorio.

Ha dado el primer paso en el camino que le llevará a conocer a fondo su sistema Red Hat Linux -- de la navegación al cambio de los permisos. Ahora es el momento de profundizar en la gestión de las posibilidades que su sistema le ofrece.

Si no tiene mucha experiencia con Linux es posible que se sienta un poco desorientado a la hora de gestionar los ficheros y los directorios.

Relájese. Si tiene experiencia en el uso de otros sistemas operativos, "aprender Linux es un poco como aprender a conducir en otro pais". Muchas de las ideas son idénticas, pero otras en cambio son algo diferentes.

Trataremos muchas de estas "normas de circulación" en este capítulo.

Pero hay un componente en su nuevo sistema operativo sin el que no podria continuar la *shell* Nos hemos referido a la shell en muchas ocasiones -- como "el prompt de la shell" o "bash"

Ahora ha llegado el momento de tratar más a fondo la utilización de esta aplicación Pero antes, algunas nociones históricas.

Shell, la Historia

En los años '60, cuando Dennis Ritchie y Ken Thompson di AT&T estaban trabajando en UNIXTM, quisieron crear un instrumento a través del cual los usuarios pudiesen intercomunicarse con el sistema.
En aquellos tiempos los sistemas operativos utilizaban los *interpretes de comandos*, que aceptaban comandos de los usuarios para despues interpretarlos y que de ese modo pudieran ser utilizados por la máquina
Pero Ritchie y Thompson querian algo más, algo que pudiese ofrecer mejores caracteristicas que los interpretes de comandos que habia por aquel entonces.
Nace la *Bourne shell* (conocida simplemente como *sh*), creada por S.R. Bourne, que satisfacia los objetivos de los creadores UNIX
Después de la creación de la shell Bourne, se desarrollaron otros tipos de shell, como la *C shell* (*csh*) y la *Korn shell* (*ksh*).
Cuando la Free Software Foundation quiso una shell libre de royaltis los desarrolladores comenzaron a trabajar en un lenguaje dentro de la Bourne shell de las caracteristicas más comunes en la shell de aquel tiempo
El resultado fue la *Bourne Again Shell* -- o *bash*

Notas:

Hasta ahorá, probablemente, haya visto la palabra bash en mensajes de error causados por teclear de manera incorrecta un comando en el prompt de la shell (como en bash:uncomando: command not found).

En el Capitulo 13, cuando hemos hablado de redireccionar y pipe, estábamos mostrando también la potencia y la flexibilidad de la bash.

Notas:

Saber más de bash

Puede aprender más sobre la bash leyendo la página man de bash. En el prompt de la shell teclee **man bash** (o sino puede salvar el fichero como fichero de texto tecleando el comando **man bash | col -b > bash.txt**,que visualizará despues con un editor como pico o un paginador como **less**. También puede imprimir un fichero con **man bash | col -b | lpr** , pero ponga atención: es un fichero de grandes dimensiones. Si desea tener más información, O'Reilly & Associates ha publicado *Learning the bash Shell*, de Cameron Newham y Bill Rosenblatt.

Aunque su sistema tenga numerosas shells diferentes, bash es la shell por defecto de Red Hat Linux. Puede imaginar la bash como a una secretaria que tiene la costumbre de anotar lo que hace y de ejecutar los comandos de manera rapida. Esta "empleada" además tiene "apuntadores" que le dicen como le gusta a usted personalizar el modo de trabajar.
Los apuntadores gestionados desde la bash se llaman *variables de entorno*.
La shell utiliza un "ambiente" de la misma manera que nosotros utilizamos un entorno como la cocina. Trabajamos en la cocina, ponemos en su sitio las cacerolas, los tarros, los alimentos. Sabemos donde están los platos y cómo funciona cada aparato.
Lo mismo se puede decir con respecto a la bash y a su entorno Existen una un orden de base en la bash como la habria en cualquier cocina. Por ejemplo, esperamos encontrar cacerolas en una cocina de la misma manera que esperamos encontrar determinados comandos en la bash.
Esta es la idea de base de las variables de entorno.
Mientras la empleada tenga los apuntadores adecuados, ejecutará los comandos velozmente
Veamos ahora las variables de entorno. En el prompt de la shell escriba:

env

Existen algunos "accesos directos" a utilizar desde la bash, ¿verdad?
Cada uno de ellos ayuda a la bash a configurar el entorno para usted.
Entre las variables de entorno más importantes están el PATH (ruta) -- que determina lo que normalmente se llama *default path* (ruta por defecto).
La variable de entorno PATH para su cuenta **newuser** podria tener el siguiente aspecto.

PATH=/usr/local/bin:/usr/X11R6/bin:/usr/bin:/bin:/usr/X11R6/bin:/home/newuser/bin

PATH apunta a los lugares más frecuentes donde se hayan los programas

Un estándar para el PATH

¿ Recuerda la referencia que se hizó con anterioridad al Estándar en la Jerarquia del Sistema (vea la sección de nombre *El sistema de archivos* en Capítulo 13)? la variable PATH se configura según este estandar y los programas son instalados en el directorio de acuerdo con el Estándar en la Jerarquía del Sistema de Ficheros El resultado es que la definición de PATH habilitará la bash a buscar de forma automática practicamente cualquier programa, suponiendo que haya sido instalado de acuerdo con el Estándar de la Jerarquia de Sistema de Ficheros

Notas:

Localizar los Ficheros y los Directorios

En algunos momentos sabremos de la existencia de algún fichero o directorio pero no sabremos cómo encontrarlo. Buscar un fichero o un directorio puede ser más sencillo con el comando **locate**. Con **locate**, veremos cada fichero y directorio que se corresponde con el criterio de busqueda. Por ejemplo si queremos buscar todos los ficheros relativos al comando **finger**.

locate finger

El comando **locate** utiliza una base de datos para controlar la existencia de ficheros o directorios que corresponden a la cadena **finger**

Más información sobre locate

Para saber más sobre **locate**, lea la página man de **locate** (teclee **man locate** en el prompt de la shell).

Es un comando útil que funciona velozmente -- al menos cuando la base de datos está actualizada. Esta base de datos se actualiza automáticamente por la noche por **cron**. ¿Qué es **cron**? es un pequeño programa que se ocupa de varios cómputos -- como la actualización de la base de datos de **locate** -- durante intervalos programados de forma regular.

Más información sobre cron

cron es un *demonio* Los Demonios se ocupan de gestionar tareas en background Para leer la página man de **cron**, escriba **man cron** en el prompt de shell.

Qué sucede si

- Tenemos más de un sistema operativo en nuestra máquina, y pasamos de uno a otro obligándonos a salir y reiniciar el sistema Red Hat Linux;
- Cierre y apague la máquina al final del dia.

Esto quiere decir que **cron** raramente tendrá la posibilidad de actualizar la base de datos *slocate*, que es utilizado para catalogar la situación de los ficheros. Pero de todos modos podemos actualizar la base de datos manualmente.

Antes de nada, **su** como root (teclee **su** en el prompt, es decir teclee la contraseña de root)
Ahora, teclee en el prompt de la shell:

updatedb

Después de unos minutos, finalizará la actualización de la base de datos. slocate

Notas:

Historia de los comandos y Rellenado con el Tabulador

No hace falta mucho para que nos cansemos de teclear siempre el mismo comando o que mientras tecleamos un recorrido largo como argumento de un comando, por un pequeño error tengamos que volver a escribirlo todo de nuevo.
Los usuarios de Linux pueden encontrarse con el mismo problema. En Linux, ya que puede unir comandos, un pequeño error al teclearlos significaría haber escrito en vano esas lineas.
Pero aquí tiene una solución: llamada *command-line history*. Utilizando los cursores, podrá volver a los comandos tecleados con anterioridad – incluso los que tecleó mal.
Volvamos a ver sneakers.txt. La primera vez, sin embargo, en el prompt de la shell teclearemos:
cat sneakrs.txt

No ocurrirá nada, obviamente, ya que el fichero sneakrs.txt no existe. No hay ningún problema. Utilizaremos el cursor en alto para volver al comando, y por tanto utilizaremos el cursor de la izquierda para volver al punto en el que habiamos tecleado "e". Introduzca la letra e y pulse**Enter** de nuevo.
¡Voila! Ahora veremos el contenido de sneakers.txt.
Por defecto en el fichero history de la bash puede haber hasta 500 comandos.

Observar la variable de entorno

Tecleando el comando **env** en el prompt de la shell, podemos visualizar las variables de entorno que controlan la medida de la historia en linea de comando. La linea que dice, HISTFILESIZE=500 muestra el número de comandos que la bash conservará.

La history de la linea de comando en realidad está contenida en un fichero, que se llama .bash_history en el directorio de login. Podemos leerlo de varias maneras: utilizando pico, **cat**, **less**, **more**, y otros.
Prepárese: el fichero podría ser bastante largo.
Leámoslo con **more**:
more .bash_history

Para avanzar en la pantalla, pulse **Espacio**, para ir hacia atrás, pulse **B**, para salir, pulse **Q**

Localizar un comando utilizado con anterioridad

¿Quiere buscar un comando en el fichero history sin tener que pulsar una y otra vez los cursores o consultar el fichero history? Puede utilizar el comando **grep**, una potente utilidad para la busqueda. He aquí como buscar rápidamente un comando utilizado con anterioridad: por ejemplo queremos buscar el comando que se llama **cat sneak**- y algo más. Ha utilizado el comando y piensa que está en el fichero history. En el prompt de la shell, teclee

history | grep sneak

Notas:

Además del comando que acaba de teclear, verá el comando exacto, ya que **grep** busca en el fichero history cualquier nombre en el que aparezca la palabra "sneak" .Puede obtener más información sobre el comando **grep** más adelante en este mismo capitulo, cuando hablemos de las herramientas que pueden servir para leer ficheros.

Otro instrumento para evitar perder tiempo es el de *completar los comandos*. Si teclea parte de un fichero, un comando o una ruta y después pulsa la tecla **Tab**, la bash le mostrará o la parte del nombre del fichero/ruta que falta o emitirá un bip. Si escucha un bip, bastará con que pulse la tecla **Tab** para obtener una lista de ficheros/rutas que se corresponden con lo que está tecleando.
Por ejemplo, si se olvida del comando **updatedb**, pero recuerda parte del mismo, puede utilizar **su** para convertirse en root, en el prompt de la shell teclee **up**, y pulse la tecla **Tab** dos veces, entonces verá una lista de posibles comandos que empiezan con la sílaba "up", como **updatedb** y **uptime**. Añadiendo la letra "d" a **up** y pulsando otra vez la tecla **Tab**, el comando será completado.
De esta manera aunque la máquina sea apagada al final del día, no resulta difícil actualizar la base de datos **slocate**. Existen muchas posibilidades de que el comando sea salvado en el fichero history o bien puede utilizar la tecla **Tab** para completar el nombre del comando (siempre y cuando recuerde al menos cómo empieza el nombre del comando).

Identificar y Trabajar con los Ficheros Type

Si es un nuevo usuario de Linux, en breve empezará a ver ficheros con extensiones que le resultarán desconocidas Una extensión de un fichero es la última parte del nombre del fichero, despues del punto final (en el fichero sneakers. txt, "txt" es la extensión de este fichero)
He aquí una breve lista de extensiones y sus significados:

Fichero Comprimidos/Archivados

- .Z -- un fichero comprimido
- .tar -- un fichero en cinta (abreviado de *tape archive*)
- .gz -- fichero comprimido (gzip)
- .tgz -- un fichero archivado con tar y comprimido con gzip

Formatos de los Ficheros

- .txt -- un fichero de texto ASCII
- .html/.htm -- un fichero HTML
- .ps -- un fichero PostScript, formateado para imprimir
- .au -- un fichero audio
- .wav -- un fichero audio

Notas:

- .xpm -- un fichero imagen
- .jpg -- un fichero gráfico o una imagen, como una foto o un dibujo
- .gif -- un fichero gráfico o una imagen
- .png -- un fichero gráfico o una imagen

Fichero de Sistema

- .rpm -- un fichero paquete del Gestor de Paquetes de Red Hat
- .conf -- un fichero de configuración
- .a -- un fichero archivo
- .lock -- un fichero "lock"; determina si un programa está en uso

Ficheros de Programación y Script

- .h -- un fichero header para el lenguaje de programación C y C++
- .c -- un fichero de código fuente para el lenguaje de programación C
- .cpp -- un fichero de código fuente para el lenguaje de programación C++
- .o -- un fichero objeto
- .pl -- un script Perl
- .tcl -- uno script TCL
- .so -- una librería

Pero las extensiones de los ficheros no siempre son utilizadas. Por tanto, ¿que ocurre cuando un fichero no tiene extensión, o el fichero no es del tipo que indica la extensión?
He aquí cuando el comando **file** puede ser útil.
En Capitulo 13, hemos creado un fichero llamado saturday -- sin extensión. Utilizando el comando **file**, podemos saber de que fichero se trata, tecleando:

file saturday

y veremos que se trata de un fichero de texto. Cualquier fichero que sea reconocido como fichero de texto debería ser legible a través de **cat**, **more**, o **less**

Leer la página man

Para saber más sobre el comando **file**, lea la página man de **file** tecleando **man file**

Veamos como leer los ficheros.

Notas:

Existen muchas maneras de leer ficheros en Linux. En Capitulo 13, por ejemplo, hemos hablado de los paginadores **more** y **less** – se llaman paginadores porque pueden pasar las "páginas" de los documentos de pantalla en pantalla. Hemos aprendido cómo podemos no solo ver sino también manipular los ficheros con el comando **cat**
Pero todavía hay más opciones para cuando queramos ver ficheros README, páginas man o documentos que hemos creado
Tiene a su disposición un cierto número de instrumentos para leer ficheros, entre ellos, el editor de texto como pico, emacs, y vim, los paginadores **more** y **less**, y los visualizadores **head**, **tail**, **cat**, y **grep**.
Veamos algunas de las caracteristicas de estos instrumentos.

El Comando less

En Capitulo 13, le hemos dado a conocer el paginador **less**. Less es el paginador más utilizado para mostrar las páginas man.
Para consultar las páginas man de **less** tecleemos

man less

Para avanzar en la pantalla, pulse **Espacio**; para ir hacia atrás pulse **B**, y para salir, pulse **Q**
Existen otras potentes caracteristicas de**less**, como su habilidad de deslizar el texto horizontalmente y de especificar el número de líneas a deslizar.

El Comando more

Aunque pueda parecer extraño, **more** ofrece menos caracteristicas que **less** (en realidad **less** está inspirado en **more**)
Echemos un vistazo a la página man de **more**, pero esta vez abriremos la página utilizando **more** -- utilizando la pipe para redirigir el output de **man** a **more**.

man more | more

Al principio puede parecer que no sean tan diferentes, pero el comando **more** acepta menos comandos que el comando **less**. Probablemente la diferencia sustancial sea la de que no puede desplazar un documento hacia atrás -- aunque si para desplazarse hacia delante ambos utilicen **Espacio** y para salir **Q**.

El comando head

Puede utilizar el comando **head** si solo quiere ver el principio de un fichero El comando es

head *<nomedelfile>*

Head puede ser útil, pero como se limita a visualizar las primeras lineas, no sabrá el tamaño **real** del fichero. Por defecto, puede leer solo las primeras 10 líneas de un fichero, si bien puede especificar el número de líneas que quiere ver, tecleando

head -20 *<nomedelfile>*

Lea la página man del comando **head** (**man head**) para obtener mas información Probablemente **less** o **more** le parezcan más útiles, ya que pueden deslizar el fichero en su totalidad

El Comando tail

Notas:

El opuesto de **head** (obviamente), es **tail**. Con **(tail)**, puede ver las 10 últimas lineas de un fichero.

El Comando cat

El comando **cat**, abreviación de concatenación, le mostrará el contenido total de un fichero en la pantalla. Utilizar **cat** puede ser de utilidad si el fichero es relativamente corto, como cuando creamos **sneakers.txt**. Pero si el fichero es bastante largo, no podrá ver todo en pantalla.

El Comando grep

El comando **grep** es bantante útil para buscar cadenas de caracteres especificas en un fichero. Supongamos que queremos buscar cualquier referencia a la palabra "topina" en el fichero sneakers.txt, que creamos en el directorio de login. Teclee:

grep topina sneakers.txt

de esta manera podemos ver cada una de las líneas en las que aparece la palabra "topina".

Recuerde las mayúsculas y las minúsculas

A menos que se especifique, las busquedas de **grep** son *sensibles a las mayúsculas y minúsculas.* Esto quiere decir que buscar *Coffee* no es lo mismo que buscar *coffee*. Entre las opciones de **grep** encontramos **-i**, que le permite ejecutar una busqueda sin tener en cuenta las letras mayúsculas/minúsculas de un fichero. Consulte la página man de **grep** para más información sobre este comando.

Direccionar I/O y la Pipe

No olvide la utilización del pipe y la redirección del output cuando quiera salvar y/o imprimir información para leerla en una ocasión posterior

Puede, por ejemplo, utilizar, **grep** para buscar determinados contenidos en un fichero, y por tanto salvalos e imprimirlos.

Para imprimir información referente a la palabra "coffee" en **sneakers.txt**, por ejemplo, teclee

grep coffee sneakers.txt | lpr

Este comando se comporta de forma parecida al comando **ls -al /etc | more**, que es posible que ya haya utilizado en Capitulo 13 para mostrar el contenido del directorio /etc y por tanto enviar los resultados a través del comando **more** para poder visualizarlos en la pantalla.

Es más seguro utilizar >>

Recuerde las diferencias entre utilizar > y >>: > sobrescribe un fichero, mientras que >> añade información a un fichero, si este ya existe En general, a no ser que este seguro de lo que va hacer, es más seguro utiliza >>, ya que no perderá información importante (y también se puede volver a modificar si no quiere que estén juntos).

Wildcard y Expresiones Regulares

¿Qué sucede si se olvida del nombre de un fichero que quiere buscar? No puede decirle al ordenador: "Búscame el fichero llamado 'sneak' o 'sneak-nosequé'"

Notas:

Bueno, si puede, en cierto modo. Utilizando los *wildcard* o las *expresiones regulares*. puede ejecutar acciones en uno o más ficheros sin saber el nombre completo del fichero Teclee solo aquello que sabe, y sustituya el resto con un wildcard.

Más información sobre los wildcard y las expresiones reguralares

Para más información sobre los wildcards y las expresiones regulares, puede consultar la página man de la bash (**man bash**). Recuerde que puede salvar el fichero en un fichero de texto, tecleando **man bash | col -b > bash.txt**. Por tanto, puede abrirlo y visualizarlo con **less** o **pico** (**pico bash.txt**). Si quiere imprimirlo,¡ojo!, es muy largo.

Sabemos que el fichero se llama "sneak-nosequé.txt," por tanto,teclee:

ls sneak*.txt

y ahora el nombre del fichero:
sneakers.txt

Probablemente utilizará el asterisco(*) frecuentemente durante una busqueda El asterisco le permite buscar cualquier cosa que se corresponda con el nombre que está buscando Por lo tanto tecleando:

ls *.txt

o:

ls sn*

también encontrará sneakers.txt -- pero cuanto más especifique con wildcards, más ficheros de texto habrá, y por tanto serán mostrados más ficheros que se correspondan con el nombre que está buscando.
Le servirá de ayuda por tanto, reducir al máximo la busqueda.
Una manera de reducir la busqueda puede ser utilizar el signo punto de interrogacion (?). Al igual que el asterisco, utilizar ? puede ayudarle en la busqueda
En este caso, sin embargo,? es útil para buscar caracteres individuales -- de este modo si está buscando sneaker?.txt, encontrará sneakers.txt como resultado -- y/o sneakerz.txt, si un fichero de este tipo existiese.
Cuando un asterisco, por ejemplo, puede formar parte de un nombre, como podría ser el caso si el nombre de fichero sneakers.txt fuese sneak*.txt, pueden servirle de ayuda las expresiones regulares.
Las expresiones regulares utilizan el asterisco y son muy fáciles de usar.
Utilizando el caracter (\), puede especificar que no desea buscar *cualquier cosa* utilizando el asterisco, en cambio está buscando un fichero con un asterisco en el nombre
Si el fichero se llama sneak*.txt, entonces, teclee

sneak*.txt

He aquí una breve lista de wildcard y de expresiones regulares:

- * -- Corresponde a todos los caracteres
- ? -- Correspone a un carácter en una cadena (como sneaker?.txt)

Notas:

- * – Corresponde al carácter *
- \? – Corresponde al carácter ?
- \) – corresponde al carácter)

También puede utilizar las wildcard para algo más que para buscar: pueden ser útiles cuando quiera eliminar o renombrara ficheros. Y las expresiones regulares pueden servirle de ayuda para renombrar ficheros con caracteres como * y ? en su nombre.

Copiar, Eliminar y Renombrar Ficheros y Directorios

Hasta ahora ha aprendido algo con referencia al sistema de ficheros; y ha aprendido como crear ficheros y directorios.
Pero el hecho de saber como crear ficheros y directorios no quiere decir que no pueda cambiar las modificaciones aportadas. ¿Cómo se pueden eliminar y/o renombrar ficheros y directorios?
Comencemos por el comando copy.

Copiar Ficheros

Como tantas otras características de Linux, tendrá una variedad de opciones entre las que elegir cuando quiere manipular los ficheros y los directorios. Puede utilizar también los wildcard durante la copia, el traslado o el borrado de los ficheros y los directorios.
Generalmente, el comando de copia no es mucho más difícil:

cp *<origen> <destino>*

de esta manera, para copiar el fichero sneakers.txt en el directorio tigger de su directorio de login, teclee

cp sneakers.txt tigger

Observe que puede utilizar algunas rutas relativas para copiar ficheros. Puede utilizar tantos rutas relativas como absolutas con el comando **cp**. El directorio de login es el directorio precedente al directorio tigger; esto quiere decir que tigger está más abajo de nuestro directorio.
Lea la página man de **cp** (**man cp**) para obtener una lista completa de opciones disponibles con cp

Notas:

Entre las opciones que puede utilizar con **cp** están:

- **-i** -- interactivo Le pide que confirme si el fichero sobrescribirá el fichero de destino. Esta es una opción útil, ya que puede ayudarle a prevenir errores.

- **-r** -- recursivo. En vez de copiar todos los ficheros y directorios, copia la totalidad del subárbol del directorio, todos los subdirectorios y los ficheros comprimidos, en otro sitio.

- **-f** -- fuerza. Copia sin preguntarle que confirme si el fichero debe ser sobreescrito, A no ser que éste seguro de lo que está haciendo, probablemente sea mejor que no utilice esta opción de momento.

- **-v** - verbose Le mostrará información sobre el progreso de la copia del fichero.

Utilizando solo **cp**. no verá mucho cuando el comando sea ejecutado. Utilizando una opción como **-i**, puede hacer el proceso más útil, ya que si quiere copiar un fichero donde ya tiene un fichero con ese nombre, se le preguntará primero que confirme si lo quiere sobrescribir -- esto quiere decir que será reemplazado -- del fichero ya existente.

No "fuerce" demasiado

Recuerde que entre las opciones esta **-f** (fuerza), que sobreescribe los ficheros sin pedir confirmación. Asegurese, cuando utilice la opción fuerza, de querer *realmente* sobrescribir un fichero

Ahora que tenemos el fichero sneakers.txt en el directorio tigger, probemos a utilizar **cp -i** para copiarlo de nuevo en el mismo sitio.

[newuser@localhost newuser]$ **cp -i sneakers.txt tigger**
cp: overwrite 'tigger/sneakers.txt'?

Para sobrescribir el fichero ya existente, pulse **Y** y despues **Enter**. ¿No desea sobrescribir el fichero? Pulse**N** y despues **Enter**.

Eliminar Ficheros

Para eliminar ficheros,utilice el comando **mv** (**man mv**), que es parecido al comando **cp**, excepto que con **mv** el fichero es fisicamente trasladado de un sitio a otro, en vez de ser duplicado como con **cp**.

Opzioni comuni disponibili con **mv** includono

- **-i** -- interactivo Le pedirá que confirme si el fichero que está desplazando sobrescribirá un fichero con el mismo nombre en el directorio de destino. Esta es una buena opción, visto que la opción **-i** in **cp**, le dará la posibilidad de reemplazar un fichero existente

- **-f** -- fuerza. No tiene en cuenta la modalidad interactiva y traslada los ficheros sin pedir confirmación A no ser que este realmente convencido de lo que está haciendo, esta opción no es muy segura, por lo tanto sea prudente a la hora de utilizarla.

- **-v** -- verbose Muestra una lista de los ficheros que son trasladados

Si quiere trasladar un fichero de su directorio home a otro directorio, teclee.

mv sneakers.txt tigger

Notas:

o bien, **mv sneakers.txt /home/newuser /home/newuser/tigger** utilizando rutas absolutas

Renombrar Ficheros

En realidad este argumento lo hemos tratado un poco, ya que cuando copia o traslada un fichero puede también renombrarlo.
Para copiar el fichero sneakers.txt de nuestro directorio de login a un subdirectorio tigger, teclee:
cp sneakers.txt tigger

Para copiar y renombrar este fichero de sneakers.txt a piglet.txt, teclee:
cp sneakers.txt tigger/piglet.txt

Para *trasladar* y renombrar el fichero, basta con sustituir **mv** a **cp** en el ejemplo de arriba
Si se mueve con **cd** en tigger y usa **ls**, verá el fichero **piglet.txt**.
si solo quiere renombrar el fichero en su sitio, utilice el comando **mv** en el directorio actual:
mv sneakers.txt piglet.txt

Borrar Ficheros y Directorios

Hemos hablado de como crear ficheros con el comando **touch** y utilizando la redireccion en Capitulo 13. Hemos creado el directorio tigger utilizando **mkdir**.
Pero no hemos hablado de cómo borrar los ficheros y los directorios.
Borrar ficheros y directorios con el comando **rm** (**man rm**) es un proceso intuitivo.
Probemos a crear un nuevo fichero piglet.txt, y borrarlo del directorio tigger con el comando **rm**
rm piglet.txt

¿Qué ocurre si realmente no queremos borrarlo? ¡Demasiado tarde! De nuevo, la opción **-i** (interactiva) vuelve a ser útil, ya que proporciona una segunda oportunidad si de verdad queremos borrar el fichero.
[newuser@localhost newuser]$ **rm -i piglet.txt**
rm: remove 'piglet.txt'?

También puede borrarlo utilizando el wildcard *, pero esté atento, porque podria facilmente borrar ficheros sin querer.
Para eliminar un fichero, escriba:
rm pig*

También puede eliminar más de un fichero con un solo comando, como con:
rm piglet.txt sneakers.txt

Notas:

Opciones para la eliminar ficheros –y directorios –:

- **-i** – interactivo Se le pide que confirme el borrado.
- **-f** -- fuerza. Borra sin esperar ningun tipo de confirmación interactiva. Esto no es una buena idea a menos que sepa exactamente qué es lo que está haciendo.
- **-v** -- verbose. Muestra una lista de los ficheros que van a ser borrados.
- **-r** -- recursivo. Cuando elimina ficheros serán eliminados también todos los ficheros y los subdirectorios del directorio especificado. Serán eliminados también los directorios vacios

Para eliminar un directorio con **rm**, debe especificar la opción **-r**.
Por ejemplo, si desea eliminar recursivamente el directorio **tigger** escriba:
rm -r tigger

Y si desea combinar las opciones, como forzar un borrado recursivo teclee
rm -rf tigger

¡Atención!

El comando **rm** es un comando muy potente, y podría borrar ¡todo el sistema!. Si utiliza el usuario root y teclea simplemente el comando **rm -rf /** ejecutará la autodestrucción-- como una serpiente que se come la cola, el comando eliminará de manera recursiva cada una de las cosas de su sistema.

Una alternativa más segura es utilizar **rm** para eliminar los directorios el comando **rmdir** Con este comando, no se le permitirá utilizar el borrado recursivo, de este modo un directorio que contenga ficheros no será borrado
Lea la página man de **rmdir** tecleando **man rmdir** para obtener más información acerca de este comando.

Notas:

BIBLIOGRAFIA

LINUX
Guia práctica para usuarios
César Martín Pérez
ANAYA

Manual avanzado de LINUX
Raúl Montero Rivero
ANAYA