Linux

Linux es un sistema operativo libre. Existen varias versiones de linux: OpenSuse, Ubuntu, FreeBSB, Fedora, Red Hat, etc. Una vez instalado el sistema linux se crea una estación de trabajo.

Tux: Nombre de la mascota oficial de Linux.

Sistema de archivos de Linux

Una vez instalado un Sistema Operativo de linux se crean carpetas de trabajo comunes para el sistema operativo y el usuario.

Carpetas de trabajo de Linux	
/bin	Contiene comandos comunes tales como ls, sort, date, y chmod.
/boot	Contiene el sistema booteable y la configuración básica del sistema instalado.
/dev	Contiene archivos representando puntos de acceso a dispositivos en el sistema.
/etc	Contiene la configuración de los archivos administrativos.
/home	Contiene las carpetas de trabajo de los usuarios.
/media	Proporciona una localización estándar para automontar los dispositivos.
/lib	Contiene librerías compartidas necesitadas por las aplicaciones en /bin y /sbin.

Sistema de archivos de Linux

Carpetas de trabajo de Linux	
/mnt	Un punto en común para todos los dispositivos montados antes de ser enviados a /media
/misc	Carpeta de trabajo usada para automontar archivos del sistema.
/opt	Estructura del directorio sugerida para almacenar programas.
/proc	Contiene información hacerca de las fuentes del sistema.
/root	Carpeta de trabajo del super usuario.
/sbin	Contiene comandos administrativos y procesos del “demonio”.
/tmp	Contiene archivos temporales del sistema.
/usr	Contiene manuales, juegos, gráficos, librerías, y archivos no necesitados por el sistema.
/var	Contiene carpetas de datos usadas por varias aplicaciones.

Comandos Básicos de Linux

Comandos para crear y usar archivos	
Comando	***Resultado***
cd	Cambia a otro directorio.
pwd	Imprime el nombre de la carpeta de trabajo en donde te ubicas.
mkdir	Crea un directorio.
chmod	Cambia el permiso de un archivo o carpeta.
ls	Despliega el contenido de un directorio.
Desde la terminal podemos desplegar los manuales de ayuda con el comando **man**	
man	Despliega el manual del comando: man mkdir para salir presionar **q**

Comandos Básicos de Linux

Comandos para mover, copiar y eliminar archivos	
Comando	***Resultado***
mv	Comando para mover un archivo de un punto a otro.
cp	Comando para copiar un archivo de un punto a otro.
rename	Comando para renombrar un archivo.
rmdir	Comando para borrar una carpeta de trabajo vacía.
rm	Comando para borrar un archivo.

Comandos Básicos de Linux

Comandos y aplicaciones usadas en el sistema	
ps	Despliega los procesos en el sistema.
kill	Elimina el proceso desplegado por el comando ps.
jobs	Despliega cuales comandos estan corriendo en segundo plano.
&	Se coloca este símbolo para enviar un trabajo o abrir una aplicación en segundo plano.
emacs	Editor gráfico.
nano	Editor de archivos de texto.
useradd	Elimina un usuario del sistema de archivos.
passwd	Cambia la contraseña de un usuario.
ssh	Comando para acceso remoto a otra computadora o servidor. **Ejemplo:** xhost 148.222.44.1 ssh -XAY usuario@148.222.44.1

Comandos Básicos de Linux

Comandos y aplicaciones usadas en el sistema	
df	Despliega el espacio disponible en tu sistema.
du	Despliega el espacio ocupado por un usuario en particular.
find	Encuentra archivos usando una variedad de criterios.
vi	Editor de archivos.
date	Despliega la fecha en tu sistema.
who	Despliega quien eres y los usuarios conectados al sistema.
locate	Localiza archivos y carpetas de trabajo.

La mascota del sistema operativo Linux es un pingüino llamado Tux.

Hay diferentes versiones sobre el origen del término. La más aceptada es la que afirma que viene del término inglés "**tux**edo", que quiere decir esmoquin, y es lo primero que se le viene a la cabeza a mucha gente cuando ve a un pingüino.

Aunque hay quien dice que podría venir también de **T**orvalds **U**ni**x**.

La mascota fue elegida por el propio Torvalds inspirándose en una foto que encontró en internet.

Tux es el protagonista de muchos de los juegos hechos para Linux como "Tux Racer", "Tux on the Run", "Super Tuxedo T. Penguin: A Quest for Herring", "Chromium B.S.U." o "Pingus".

RESUMEN

- Un **sistema operativo** es un programa que permite al usuario interactuar con el ordenador y sus componentes hardware y que facilita la realización de tareas básicas.

- Trabajar mediante comandos, en una ventana de terminal, permite realizar tareas de forma similar en cualquier versión de Linux o Unix.

- **Linux** es un sistema operativo que se caracteriza por ser libre y, en la mayoría de los casos también gratuito. Está hecho por voluntarios. Es multiusuario, multitarea y multiplataforma. Es muy estable y aprovecha bien los recursos de que dispone la máquina. La mayoría de los programas disponibles para Linux son también libres.

- La principal diferencia entre Linux y **Unix** radica en que Linux es libre y multiplataforma mientras que Unix suele ser comercial y muy orientado al hardware. **Windows** también es un sistema operativo comercial y las aplicaciones para este SO también suelen ser comerciales.

- Se puede usar Linux sin tener que instalar nada en el ordenador mediante alguno de estos métodos: live-CD, lápiz de memoria, telnet.

- Una **distribución** consta del sistema operativo propiamente dicho más el programa de instalación y una selección de aplicaciones. Algunas de las distribuciones más importantes son Ubuntu, openSUSE, Mint, Fedora, Debian y Mandriva.

- La primera versión de Linux fue creada por Linus Torvalds en 1991 con el fin de mejorar MINIX, un sistema operativo tipo UNIX utilizado en la universidad.

- La mascota de Linux es un pingüino al que se ha bautizado con el nombre de **Tux**.

EJERCICIOS

En ocasiones, la respuesta a los ejercicios no se puede completar únicamente con el material teórico que se proporciona en este capítulo y el alumno debe, por tanto, buscar en otras fuentes complementarias como Internet.
Las soluciones a los ejercicios se encuentran al final del libro.

Los ejercicios están clasificados según su nivel de dificultad:

	Fácil. El concepto viene explicado en el capítulo.
	Dificultad media. Es necesario relacionar conceptos y/o buscar información en
	Internet. Difícil. Hace falta una investigación concienzuda

1.  Un sistema operativo es a) un programa que permite al usuario realizar tareas específicas, b) un procesador de textos, c) un programa que permite al usuario interactuar con el ordenador y sus componentes o d) ninguna de las respuestas anteriores es correcta.

2.  Una distribución de Linux es a) el núcleo del SO, junto con un programa de instalación y una selección de aplicaciones, b) el núcleo del SO, junto con un entorno gráfico y una selección de aplicaciones o c) las dos respuestas anteriores son correctas.

3.  Cita a) alguna distribución de Linux buena en el apartado gráfico y en juegos 3D y b) alguna distribución de Linux con servidores preinstalados.

4.  a) ¿Cuáles son las diez distribuciones de Linux más populares del último mes? b) y del último año?

5.  a) ¿Quién creó la primera versión de Linux?, b) ¿con qué objetivo?

6.  Para ejecutar comandos de Linux en una máquina remota mediante telnet hace falta instalar antes Linux en la máquina local. a) Verdadero b) Falso.

7.  a) ¿La primera versión de Linux se creó para un tipo de máquina concreta o podía funcionar con cualquier hardware? b) Sucede lo mismo hoy día.

8.  La mascota de Linux se llama a) Tuxedo, b) Tux, c) Pingu o d) todas las anteriores son correctas.

9.  ¿Quién fue el artista encargado de dibujar la primera versión de la mascota de Linux?

10.  Citar tres distribuciones de Linux ligeras (con pocas necesidades en cuanto a recursos hardware) que se puedan instalar en un lápiz de memoria.

En este caso,

```
cd Music
```

sería equivalente a

```
cd /home/luisjose/Music
```

ya que se suma la ruta actual (`/home/luisjose`) a la ruta relativa indicada (`Music`)

Las rutas, tanto las absolutas como las relativas se pueden utilizar en la mayoría de comandos. No son algo específico que se utilice sólo con `cd`.

Podemos, por ejemplo, utilizar rutas como argumentos del comando `ls`.

```
luisjose@ubuntu-desktop:~/Music$ ls /boot/grub/
default         installed-version  minix_stage1_5     xfs_stage1_5
device.map      jfs_stage1_5       reiserfs_stage1_5
e2fs_stage1_5   menu.lst           stage1
fat_stage1_5    menu.lst~          stage2
```

Dos puntos (..) hacen referencia al directorio que hay justo a un nivel superior.

```
luisjose@ubuntu-desktop:~/Music$ ls ..
Desktop  Documents  Examples  Music  Pictures  Public  Templates  Videos
```

`ls ..` muestra el contenido del directorio `/home/luisjose` que es el directorio que hay justo a un nivel superior de `/home/luisjose/Music`

```
luisjose@ubuntu-desktop:~/Music$ cd ..
luisjose@ubuntu-desktop:~$ pwd
/home/luisjose
```

`cd ..` sube un nivel en la estructura de directorios

2.3.4 mkdir

Se pueden crear directorios con el comando `mkdir`. Por ejemplo, para crear una estructura de carpetas donde un estudiante guardará información sobre sus asignaturas según el siguiente esquema:

<table>
<tr><td rowspan="11">/home/luisjose</td><td>/Documentos</td><td></td><td></td><td></td></tr>
<tr><td>/Escritorio</td><td></td><td></td><td></td></tr>
<tr><td>/Imágenes</td><td></td><td></td><td></td></tr>
<tr><td>/Música</td><td></td><td></td><td></td></tr>
<tr><td rowspan="4">/matematicas</td><td rowspan="4">/curso_01</td><td>/algebra</td><td>/examenes_antiguos
/apuntes</td></tr>
<tr><td>/analisis</td><td></td></tr>
<tr><td>/fisica</td><td>/libros_de_ejercicios
/videos</td></tr>
<tr><td>/informatica</td><td>/compiladores_pascal</td></tr>
<tr><td>/Video</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
</table>

tendría que hacer lo siguiente:

```
~$ mkdir matematicas
~$ cd matematicas/
~/matematicas$ mkdir curso_01
~/matematicas$ cd curso_01/
~/matematicas/curso_01$ mkdir algebra analisis fisica
informatica ~/matematicas/curso_01$ ls
algebra  analisis  fisica  informatica
~/matematicas/curso_01$ cd algebra/
~/matematicas/curso_01/algebra$ mkdir examenes_antiguos apuntes
~/matematicas/curso_01/algebra$ cd ..
~/matematicas/curso_01$ cd fisica
~/matematicas/curso_01/fisica$ mkdir libros_de_ejercicios
~/matematicas/curso_01/fisica$ mkdir videos
~/matematicas/curso_01/fisica$ cd ..
~/matematicas/curso_01$ cd informatica/
~/matematicas/curso_01/informatica$ mkdir compiladores_pascal
```

Nótese que ya no se muestra en el ejemplo el prompt completo, con el nombre de usuario y el nombre de la máquina. Se seguirá en el libro esta norma a partir de ahora.

2.4 VISUALIZACIÓN DE FICHEROS (cat, more, less, head, tail)

Los comandos `cat`, `more` y `less` sirven para mostrar el contenido de ficheros de texto. La diferencia radica en cómo se muestra el contenido. A todos estos comandos hay que pasarles como argumento el fichero que se quiere mostrar. Se puede indicar una ruta, en caso de que el fichero que se quiere mostrar no esté en el directorio actual.

El comando cat muestra por pantalla el contenido de un fichero y, cuando termina, el usuario está otra vez de vuelta en la línea de comandos.
Por ejemplo,

```
~$ cat /var/log/dmesg
```

muestra el contenido del fichero `dmesg` que está dentro del directorio `/var/log`. Si el lector ha probado a hacerlo él mismo, se habrá dado cuenta de que es imposible ver todo el contenido de este fichero, porque ha pasado por pantalla muy rápido. Por eso `cat` se suele utilizar para visualizar el contenido de archivos pequeños.

El comando `more` hace lo mismo que `cat`, a diferencia de que muestra el fichero pantalla a pantalla, es decir, llena de texto la pantalla y se espera a que el usuario pulse la tecla <espacio> para pasar a la siguiente:

```
~$ more /var/log/dmesg
```

El comando `less` es el más versátil de los tres, ya que permite moverse hacia delante y hacia atrás dentro del fichero, utilizando los cursores o las teclas de “AvPág” y “RePág”:

```
:~$ less /var/log/dmesg
```

En cualquier momento se puede interrumpir la visualización y volver al símbolo del sistema pulsando la letra “q”.

Los comandos `head` y `tail` permiten mostrar de forma parcial el contenido de un fichero. Como su nombre indica, head muestra las primeras líneas del fichero (la cabecera) y tail muestra las últimas líneas (la cola).

Veamos algunos ejemplos:

```
~$ head /boot/grub/menu.lst
# menu.lst - See: grub(8), info grub, update-grub(8)
#            grub-install(8), grub-floppy(8),
#            grub-md5-crypt, /usr/share/doc/grub
#            and /usr/share/doc/grub-doc/.

## default num
# Set the default entry to the entry number NUM. Numbering starts from 0, and
# the entry number 0 is the default if the command is not used.
#
# You can specify 'saved' instead of a number. In this case, the default entry

~$ tail /boot/grub/menu.lst
root          (hd0,0)
kernel        /boot/vmlinuz-2.6.24-19-generic   root=UUID=409e68a1-6123-476f-abf7-
042854b68f3c ro single
initrd        /boot/initrd.img-2.6.24-19-generic

title         Ubuntu 8.04.2, memtest86+
root          (hd0,0)
kernel        /boot/memtest86+.bin

quiet

### END DEBIAN AUTOMAGIC KERNELS LIST
```

Por defecto, tanto `head` como `tail` muestran 10 líneas, pero eso se puede cambiar con la opción `-n`.

```
~$ tail -n4 /boot/grub/menu.lst
kernel        /boot/memtest86+.bin
quiet

### END DEBIAN AUTOMAGIC KERNELS LIST
```

En este caso se han mostrado solamente 4 líneas.

2.5 EDICIÓN DE FICHEROS (touch, vi, ee, mcedit)

El comando `touch` permite crear un fichero vacío. Con cualquier editor de texto se puede crear un fichero vacío pero con `touch` es especialmente cómodo y rápido.

```
~$ ls
Desktop  Documents  Examples  Music  Pictures  Public  Templates  Videos
~$ touch prueba.txt
~$ ls
Desktop    Examples  Pictures    Public     Videos
Documents  Music     prueba.txt  Templates
~$ cat prueba.txt
~$
```

Se puede ver en el ejemplo cómo se ha creado el archivo `prueba.txt` pero al visualizar su contenido con `cat`, no aparece nada en pantalla, por tanto está vacío.

El programa `ee` es un editor muy rudimentario pero al mismo tiempo efectivo. Podemos editar el archivo anterior y escribir alguna frase:

```
~$ee prueba.txt
```

Presionando la tecla ESC, el usuario puede salir al menú principal y guardar el fichero.
Podemos comprobar ahora cuál es el contenido del fichero:

```
~$ cat prueba.txt
Hola, aquí estoy aprendiendo Linux.
```

Otro editor muy simple es `nano`. Se deja al lector curioso probar su funcionamiento y compararlo con `ee`. En el hipotético caso de no estar instalado alguno de estos editores, su instalación es muy sencilla, basta con teclear sudo `apt-get install` seguido del nombre del programa que queremos instalar. Por ejemplo, si queremos instalar `ee`:

```
~$ sudo apt-get install ee
```

El programa `mcedit` es un editor algo más sofisticado que `ee` o `nano` (al menos en apariencia) y es una parte de `mc` (Midnight Commander), un programa muy al estilo del famoso Norton Commander de MS-DOS . Vamos a modificar el archivo `prueba.txt` creado anteriormente. Antes de eso, instalaremos mc, ya que no está instalado por defecto:

```
~$ sudo apt-get install mc
~$ mcedit prueba.txt
```

Con la tecla F2 guardamos los cambios y con dos pulsaciones de ESC (o con la tecla F10) salimos del programa.
Comprobamos ahora que todo se ha grabado bien:

```
~$ cat prueba.txt
Hola, aquí estoy aprendiendo Linux.
Me encanta, se pueden hacer muchas cosas.
```

Hemos dejado para el final al editor de Linux por excelencia, se trata de `vi`. A primera vista es el más difícil de utilizar, lo cual es cierto, y parece que tiene menos opciones, pero muy al contrario se trata de un programa muy potente. Cualquier "linuxero" que se precie debe saber manejar bien este programa.
Añadiremos una línea más al fichero prueba.txt. Para ello, seguiremos los siguientes pasos:

```
~$ vi prueba.txt
```

- Pulsar la letra "i" para entrar en modo "edición".
- Escribiremos el texto.
- Pulsar la tecla ESC para salir del modo "edición".
- Teclear ":" + "w" + INTRO para grabar los cambios.
- Teclear ":" + "q" + INTRO para salir del programa.

Comprobamos una vez más que todo está bien grabado:

```
~$ cat prueba.txt
Hola, aquí estoy aprendiendo Linux.
Me encanta, se pueden hacer muchas cosas.
¡Pronto dominaré el editor Vi!
```

Es más que recomendable realizar el tutorial llamado `vimtutor`.

RESUMEN

• Todo usuario necesita un **nombre** y una **contraseña** para entrar en el sistema.

• La información se almacena físicamente en **directorios** y **subdirectorios** (carpetas y subcarpetas).

• Hay una serie de directorios predefinidos como /bin, /dev, /home, /etc, /var, etc. para todos los sistemas Linux.

• Hay **rutas absolutas,** que comienzan por el carácter "/", y que definen una ruta efectiva completa y **rutas relativas**, que no comienzan por el carácter "/", y cuya ruta efectiva sería la concatenación del directorio actual con esa misma ruta relativa.

• Los comandos vistos en este capítulo son los siguientes:

Comando	*Acción*	*Ejemplo*
pwd	muestra el directorio actual	pwd
ls	lista ficheros y directorios	ls -l
cd	cambia de directorio	cd mp3/wim_mertens
mkdir	crea uno o varios directorios	mkdir cartas facturas
cat	visualiza un fichero	cat /var/log/dmesg
more	visualiza un fichero pantalla a pantalla	more /var/log/dmesg
less	visualiza un fichero pantalla a pantalla y permite retroceder	less /var/log/dmesg
head	visualiza las primeras filas de un fichero	head -n5 /var/log/dmesg
tail	visualiza las últimas filas de un fichero	tail /var/log/dmesg
touch	crea un fichero vacío	touch listado.txt
ee	editor de textos muy simple	ee listado.txt
mcedit	editor de textos que forma parte de Midnight Commander	mcedit listado.txt
vi	editor de textos muy potente	vi listado.txt
apt-get	instala y desinstala programas	apt-get install mc
man	muestra ayuda sobre un determinado comando	man ls

EJERCICIOS

En ocasiones, la respuesta a los ejercicios no se puede completar únicamente con el material teórico que se proporciona en este capítulo y el alumno debe, por tanto, buscar en otras fuentes complementarias. En los ejercicios de este capítulo se recomienda consultar las páginas `man`.
Las soluciones a los ejercicios se encuentran al final del libro.

Los ejercicios están clasificados según su nivel de dificultad:

	Fácil. El concepto viene explicado en el capítulo.
	Dificultad media. Es necesario relacionar conceptos y/o buscar información en
	Internet. Difícil. Hace falta una investigación concienzuda

1.  ¿En qué directorio se encuentran los ficheros de configuración del sistema?

2.  Para entrar en un sistema Linux hace falta a) nombre de usuario, contraseña y dirección IP, b) nombre de usuario y contraseña o c) únicamente una contraseña..

3.  Muestra el contenido del directorio actual.

4.  Muestra el contenido del directorio que está justo a un nivel superior.

5.  ¿En qué día de la semana naciste?, utiliza la instrucción `cal` para averiguarlo.

6.  Muestra los archivos del directorio /bin

7.  Suponiendo que te encuentras en tu directorio personal (/home/nombre), muestra un listado del contenido de /usr/bin a) con una sola línea de comando, b) moviéndote paso a paso por los directorios y c) con dos líneas de comandos.

8.  Muestra todos los archivos que hay en /etc y todos los que hay dentro de cada subdirectorio, de forma recursiva (con un solo comando).

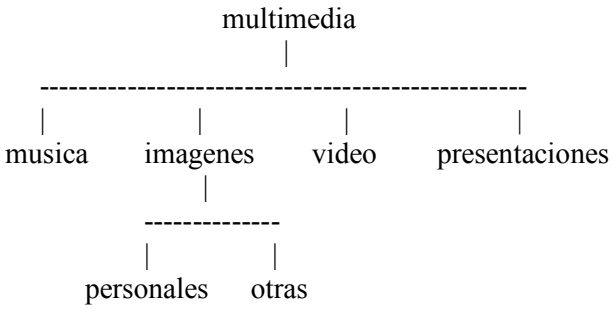
9.  Muestra todos los archivos del directorio /usr/X11R6/bin ordenados por tamaño (de mayor a menor). Sólo debe aparecer el nombre de cada fichero, sin ninguna otra información adicional.

10.  Muestra todos los archivos del directorio /etc ordenados por tamaño (de mayor a menor) junto con el resto de características, es decir, permisos, tamaño, fechas de la última modificación, etc. El tamaño de cada fichero debe aparecer en un formato “legible”, o sea, expresado en Kb, Mb, etc.

11.  Muestra todos los archivos del directorio /bin ordenados por tamaño (de menor a mayor). Sólo debe aparecer el tamaño y el nombre de cada fichero, sin ninguna otra información adicional. El tamaño de cada fichero debe aparecer en un formato “legible”, o sea, expresado en Kb, Mb, etc.

12.  Muestra el contenido del directorio raíz utilizando como argumento de `ls` una ruta absoluta.

13. Muestra el contenido del directorio raíz utilizando como argumento de `ls` una ruta relativa. Suponemos que el directorio actual es `/home/elena/documentos`.
14. Crea el directorio `gastos` dentro del directorio personal.
15. ¿Qué sucede si se intenta crear un directorio dentro de `/etc`?
16. Muestra el contenido del fichero `/etc/fstab`
.
17. Muestra las 10 primeras líneas del fichero `/etc/bash.bashrc`
18. Crea la siguiente estructura de directorios dentro del directorio de trabajo personal:



19. Crea un fichero vacío dentro del directorio `musica`, con nombre `estilos_favoritos.txt`
20. Utiliza tu editor preferido para abrir el fichero `estilos_favoritos.txt` e introduce los estilos de música que más te gusten. Guarda los cambios y sal.
21. Muestra todo el contenido de `estilos_favoritos.txt`
22. Muestra las 3 primeras líneas de `estilos_favoritos.txt`
23. Muestra la última línea de `estilos_favoritos.txt`
24. Muestra todo el contenido del `fichero estilos_favoritos.txt` excepto la primera línea. Se supone que no sabemos de antemano el número de líneas del fichero.

RESUMEN

• Utilización de los símbolos comodín:

Ejemplos	*Significado*
`*`	Cualquier cadena de caracteres.
`*f*`	Cadena de caracteres que contienen una f.
`z*`	Cadena de caracteres que empieza por z y le sigue cualquier cosa.
`a?`	Una cadena formada por dos caracteres, el primero una a y el segundo, cualquier carácter.
`[Dd]ocumento`	Puede ser Documento o documento.
`A[a-z][0-6]`	Una cadena formada por la A mayúscula seguida de cualquier letra minúscula, seguida a su vez de un dígito del 0 al 6.

• Los comandos vistos en este capítulo son los siguientes:

Comando	*Acción*	*Ejemplo*
cp	copia archivos o directorios	cp *.txt correspondencia/
mv	mueve o renombra archivos o directorios	mv palabras.txt texto.txt
rm	borra archivos o directorios	rm -R cosas/basurilla
rmdir	borra directorios	rmdir viejo

EJERCICIOS

En ocasiones, la respuesta a los ejercicios no se puede completar únicamente con el material teórico que se proporciona en este capítulo y el alumno debe, por tanto, buscar en otras fuentes complementarias. En los ejercicios de este capítulo se recomienda consultar las páginas `man`.
Las soluciones a los ejercicios se encuentran al final del libro.

Los ejercicios están clasificados según su nivel de dificultad:

	Fácil. El concepto viene explicado en el capítulo.
	Dificultad media. Es necesario relacionar conceptos y/o buscar información en
	Internet. Difícil. Hace falta una investigación concienzuda

1.  Muestra todos los archivos del directorio actual que son imágenes `jpg`.

2.  Muestra todos los archivos del directorio `/usr/bin` que empiecen por la letra `j`.

3.  Muestra los archivos que empiecen por `k` y tengan una a en la tercera posición, dentro del directorio `/usr/bin`.

4.  Muestra los archivos del directorio `/bin` que terminen en `n`.

5.  Muestra todos los archivos que hay en `/etc` y todos los que hay dentro de cada subdirectorio, de forma recursiva.

6.  Crea un directorio en tu directorio de trabajo con nombre `prueba`. Copia el archivo `gzip` del directorio `/bin` al directorio `prueba`. Crea un duplicado de `gzip` con nombre `gzip2` dentro de `prueba`.

7.  Cambia el nombre de prueba a prueba2. Crea prueba3 en el mismo nivel que prueba2 y mueve todos los ficheros de prueba2 a prueba3. Borra prueba2.

8.  Crea un fichero vacío con nombre "*?Hola caracola?*". ¿Se puede? En caso de que se pudiera, ¿sería recomendable poner nombres así? Razona la respuesta.

9.  Crea un directorio con nombre `multimedia_pruebas` y copia en él todo el contenido del directorio `multimedia`. A continuación crea en `multimedia/video/` dos ficheros, uno con nombre `peliculas.txt` y otro con nombre `actores.txt`. Edita el fichero `peliculas.txt` e introduce el nombre de tu película favorita. A continuación, crea en `multimedia_pruebas/video/` otro fichero que también tenga por nombre `peliculas.txt`, edítalo y esta vez escribe el nombre de tus cinco películas favoritas. Ahora haz una copia de todo el contenido de `multimedia` en `multimedia_prueba` de tal forma que sólo se copien los contenidos nuevos, es decir, si hay coincidencia en el nombre de un archivo se respetará el que se haya modificado más recientemente. Para comprobar que se ha hecho todo correctamente, basta mirar si en multimedia_prueba/video está el archivo vacío `actores.txt` y además el archivo `peliculas.txt` debe contener 5 películas y no 1.

10.  Borra el directorio multimedia/imagenes/otras. El sistema debe pedir al usuario que confirme el borrado.

11.  Mueve el archivo peliculas.txt, que está dentro de multimedia/video, al directorio que está justo a un nivel superior. Ahora el archivo debe llamarse mis_peliculas.txt en lugar de películas.

GRUPOS, USUARIOS Y PERMISOS

4.1 ¿POR QUÉ EXISTEN GRUPOS, USUARIOS Y PERMISOS?

Vimos en un capítulo anterior que los ficheros deben estar organizados en directorios (carpetas) con el fin de tenerlos ordenados y poder localizarlos convenientemente.
Volvamos a nuestro ejemplo de la oficina. Cada papel está en su sitio, hay carpetas y subcarpetas y todo está organizado. Ahora bien, el contable deberá tener acceso por ejemplo a las carpetas donde se encuentras las facturas y los recibos pero no tienen por qué tener acceso a la información sobre desarrollo de productos o marketing. En un sistema Linux, las carpetas y los archivos funcionan de esta manera. Por ejemplo, los archivos de configuración que se encuentran en el directorio `/etc` sólo pueden ser modificados por el administrador del sistema. Esto previene que cualquier usuario pueda cambiar información crítica y estropear algo.

4.2 ¿QUÉ ES EL SUPERUSUARIO?

El superusuario, administrador del sistema o simplemente el `root`, es un usuario especial que tiene privilegios para cambiar la configuración, borrar y crear ficheros en cualquier directorio, crear nuevos grupos y usuarios, etc.

> **IMPORTANTE:** ES PELIGROSO TRABAJAR COMO SUPERUSUARIO, SE PUEDE DAÑAR EL SISTEMA DE FORMA IRREVERSIBLE. EL LECTOR DEBE ESTAR SEGURO DE LO QUE HACE CUANDO TRABAJE COMO SUPERUSUARIO.

Una vez hecha esta aclaración, pasemos a hacer algo como `root`:

```
$ touch /etc/prueba.txt
touch: no se puede efectuar `touch' sobre «/etc/prueba.txt»: Permiso denegado
$ sudo touch /etc/prueba.txt
$ ls /etc/pru*
/etc/prueba.tx
t
```

Hemos intentado primero crear el fichero `prueba.txt` en el directorio `/etc` como usuario normal y acto seguido hemos obtenido un error de "Permiso denegado", lo que quiere decir que un usuario sin privilegios no puede hacer eso. A continuación lo hemos intentado como administrador, para ello hemos usado el comando `sudo`, tras lo que se nos ha preguntado la clave del administrador. Esta vez sí lo hemos conseguido. No tendría mucho sentido que el sistema no preguntase por la clave, ya que en ese caso cualquiera podría ejecutar comandos como administrador con el peligro que ello supone.

4.3 PERMISOS

La información sobre grupos, usuarios y permisos se puede obtener mediante el comando `ls` junto con la opción `-l`. Vamos a ver los permisos que tiene establecidos el fichero `whatis` que se encuentra en el directorio `/usr/bin`.

```
$ ls -l /usr/bin/whatis
-rwxr-xr-x 1 root root 87792 2008-03-12 14:24 /usr/bin/whatis
```

En la primera columna aparecen los **permisos**, en la tercera se indica el **usuario** (en este caso es el administrador del sistema) y en la cuarta columna aparece el nombre del **grupo** (que en este caso coincide con el de usuario).

Vamos a ver qué significan exactamente los caracteres de la primera columna:

-	r	w	x	r	-	x	r	-	x
Tipo de fichero.	Permisos para el dueño del fichero.			Permisos para el grupo al que pertenece el fichero.			Permisos para el resto de usuarios		

r	Permiso de **lectura**.
w	Permiso de **escritura**.
x	Permiso de **ejecución**.

El tipo de fichero se indica en la siguiente tabla:

Tipo de fichero	
l	Enlace simbólico.
c	Dispositivo especial de caracteres.
b	Dispositivo especial de bloques.
p	FIFO (estructura de datos).
s	Socket (comunicaciones).
-	Ninguno de los anteriores. Puede ser un fichero de texto, un binario, etc.

En el caso que nos ocupa tenemos un carácter "-" como tipo de fichero, porque se trata de un binario (un programa). El dueño del fichero tiene los permisos `rwx`, lo que quiere decir que puede leer, escribir y ejecutar el fichero. Que tiene permiso para escribir significa que puede borrarlo, cambiarle el nombre o editarlo. Tanto el grupo como el resto de usuarios tienen los permisos `r-x`, lo que significa que pueden utilizarlo (pueden leerlo y ejecutarlo) pero no lo pueden modificar.

4.4 ¿QUIÉNES SOMOS? (whoami, groups)

Antes de empezar a crear usuarios, crear grupos y cambiar permisos, debemos saber quiénes somos y a qué grupo o grupos pertenecemos. Aunque, en principio, entremos en el sistema como un determinado usuario, podemos utilizar `su` para ejecutar comandos como otro usuario distinto, siempre y cuando sepamos la contraseña de ese otro usuario.

```
$ whoami
luisjose
$ su alumno
Contraseña:
$ whoami
alumno
```

Para volver a ser el usuario original basta con utilizar `exit`.

```
$ whoami
alumno
$ exit
exit
$ whoami
luisjose
```

Con el comando `groups` se puede ver a qué grupo pertenecemos.

```
luisjose@luisjose-xps1330:~$ groups
luisjose adm dialout cdrom floppy audio dip video plugdev scanner lpadmin admin
netdev powerdev sambashare
```

Se pueden especificar uno o más usuarios detrás de `groups`. Eso nos dirá a qué grupos pertenece cada uno de ellos.

```
luisjose@luisjose-xps1330:~$ groups alumno root
alumno : alumno
root : root
```

4.5 GESTIÓN DE GRUPOS (groupadd, groupdel, groupmod)

Los comandos `groupadd`, `groupdel` y `groupmod` permiten crear, borrar y modificar grupos respectivamente.

Vamos a crear los grupos oficina_malaga, oficina_jaen y oficina_madrid

```
$ groupadd oficina_malaga
groupadd: incapaz de bloquear el fichero de grupos
$ sudo groupadd oficina_malaga
$ sudo groupadd oficina_jaen
$ sudo groupadd oficina_madrit
```

Vemos que si intentamos crear un grupo como usuario sin privilegios obtenemos un error. Para manejar grupos y usuarios es necesario ejecutar los comandos con privilegios de administrador, por tanto deberemos teclear `sudo` antes del comando en cuestión.

Hemos escrito mal el nombre del segundo grupo, ¡que no cunda el pánico!, este problema se puede solventar con `groupmod`.

```
$ sudo groupmod -n oficina_madrid oficina_madrit
```

La directiva de la empresa ha decidido cerrar la oficina de Jaén para ahorrar costes y pasar los recursos a la oficina de Málaga, así que no hará falta el grupo `oficina_jaen`. Lo podemos borrar con `groupdel`.

```
$ sudo groupdel oficina_jaen
```

4.6 GESTIÓN DE USUARIOS (adduser, userdel, usermod)

La gestión de usuarios, al igual que la de grupos, exige que los comandos se ejecuten con los privilegios del administrador del sistema. Se puede escribir `sudo` antes de cada comando, o se puede hacer lo siguiente:

```
$ sudo bash
```

Note el lector que el prompt ha cambiado. Ahora se muestra un carácter "`#`" en lugar de un "`$`". A partir de ahora, todos los comandos se ejecutarán con privilegios de administrador del sistema. Hay que acordarse de volver al usuario inicial mediante `exit`.

Es necesario dar de alta a dos usuarios para el grupo `oficina_malaga` y uno para `oficina_madrid`. Habrá un cuarto usuario que estará yendo y viniendo de una oficina a otra, por tanto se le dará de alta en las dos.

```
# adduser pedro --ingroup oficina_malaga
# adduser ana --ingroup oficina_malaga
# adduser berta --ingroup oficina_madrid
# adduser laura --ingroup oficina_malaga
# adduser laura oficina_madrid
```

Hemos matado dos pájaros de un tiro. Hemos creado los usuarios y al mismo tiempo los hemos incluido dentro de los grupos correspondientes. Estos dos pasos se pueden hacer de forma independiente.
El usuario `laura` pertenece a dos grupos. En primer lugar se ha creado el usuario y al mismo tiempo se ha añadido al grupo `oficina_malaga` con la opción `–ingroup`. Para añadir un usuario existente a un grupo, se utiliza `adduser` sin opciones.

```
# groups ana berta laura
ana : oficina_malaga
berta : oficina_madrid
laura : oficina_malaga oficina_madrid
```

Es importante destacar que se ha utilizado **`adduser`** y no `useradd`. Este último se considera un comando de bajo nivel y se recomienda utilizar el primero.

Al crear los usuarios, se nos han pedido las claves, no obstante estas claves se pueden cambiar con el comando `passwd`.

```
# passwd pedro
# passwd ana
# passwd laura
```

Recuerde el lector salir del modo `root` con el comando `exit` cuando no tenga que hacer tareas que requieran privilegios de administrador.

```
# exit
```

De ahora en adelante, simplemente se indicará con el carácter "`$`" que se trabaja como usuario sin privilegios y con el carácter "`#`" que se trabaja como `root`.

Cabe señalar que para cada usuario, se crea por defecto un directorio dentro de `/home`. Cuando un usuario se conecta al sistema, "aterriza" en ese directorio. Es lo que hemos denominado anteriormente como el directorio de trabajo.

```
$ ls /home/
alumno  ana  berta  ftp  laura  luisjose  pedro
```

4.7 CAMBIO DE GRUPO Y DE DUEÑO (chown, chgrp)

Imaginemos que el fichero `informe.txt` ha sido creado por el usuario `pedro`. Por defecto, el dueño de un archivo es el usuario que lo crea, en este caso `pedro`. El grupo del usuario `pedro`, como hemos visto antes es `oficina_malaga`.

```
$ su pedro
$ cd
$ pwd
/home/pedro
$ touch informe.txt
$ ls -l
-rw-r--r-- 1 pedro oficina_malaga  0 2009-03-19 12:46 informe.txt
```

Todo esto se puede cambiar. Moveremos el fichero al directorio de trabajo del usuario `laura` y le cambiaremos el dueño.

```
# mv informe.txt /home/laura/
# cd /home/laura/
# chown laura informe.txt
# ls -l
-rw-r--r-- 1 laura oficina_malaga  0 2009-03-19 12:46 informe.txt
```

Ahora el fichero tiene al usuario `laura` como propietario.
Tanto `chown` como `chgrp` se pueden usar con la opción -R para cambiar el dueño o el grupo en un directorio completo, de forma recursiva.

4.8 CAMBIO DE PRIVILEGIOS (chmod)

El comando `chmod` sirve para cambiar los permisos de uno o varios ficheros. Esos mismos permisos que se pueden ver con `ls -l`.

```
$ ls -l
-rw-r--r-- 1 pedro oficina_malaga  0 2009-03-19 15:38 hola_mundo.rb
$ chmod +x hola_mundo.rb
$ ls -l
-rwxr-xr-x 1 pedro oficina_malaga  0 2009-03-19 15:38 hola_mundo.rb
```

Hemos añadido el permiso de ejecución al fichero `hola_mundo.rb`. Vemos que ahora hay tres `x`, la que corresponde al dueño del fichero, la de todos los usuarios que pertenecen al grupo y la del resto de usuarios.

Cuando no se especifica ninguna de estas tres letras correspondientes a los usuarios (u, g, o) como en el ejemplo anterior, se sobreentiende que nos referimos a todos ellos. Se puede indicar de forma explícita con el carácter a (all).

Para entenderlo mejor, en la siguiente tabla, se muestran de forma esquemática, los parámetros del comando `chmod`:

u	g	o	+ -	r	w	x
(user) dueño del fichero	(group) usuarios que pertenecen al mismo grupo	(others) el resto de usuarios	dar permiso quitar permiso	(read) lectura	(write) escritura	(execution) ejecución

Quitaremos ahora el permiso de ejecución para el resto de usuarios (others) y daremos permiso de escritura (write) a los usuarios del mismo grupo (group).

```
$ ls -l
-rwxr-xr-x 1 pedro oficina_malaga  0 2009-03-19 15:38 hola_mundo.rb
$ chmod o-x hola_mundo.rb
$ chmod g+w hola_mundo.rb
$ ls -l
-rwxrwxr-- 1 pedro oficina_malaga  0 2009-03-19 15:38 hola_mundo.rb
```

A este método, que utiliza los caracteres `rwx` se le denomina método simbólico. Podemos utilizar de forma análoga el método numérico.

4 2 1	Total
r w x	4 +2+1=**7**
r w -	4 +2+0=**6**
r - x	4 +0+1=**5**
r - -	4 + 0 +0=**4**
- w x	0 + 2 + 1 = **3**
- w -	0 + 2 + 0 = **2**
- - x	0 + 0 + 1 = **1**

De esta forma, esta línea

```
$ chmod 755 hola_mundo.rb
```

sería equivalente a estas tres

```
$ chmod u+rwx hola_mundo.rb
$ chmod g+rx-w hola_mundo.rb
$ chmod o+rx-w hola_mundo.rb
```

En efecto

```
$ ls -l
-rwxr-xr-x 1 pedro oficina_malaga  0 2009-03-19 15:38 hola_mundo.rb
```

Los permisos de los directorios se pueden cambiar de la misma forma que los ficheros, aunque el significado es algo diferente. Si un directorio tiene el permiso de lectura quiere decir que se puede ver su contenido. Si tiene permiso de escritura, quiere decir que se pueden crear ficheros dentro y si tiene permiso de ejecución quiere decir que se puede entrar dentro.

RESUMEN

• Los comandos vistos en este capítulo son los siguientes:

Comando	*Acción*
ls -l	Muestra, entre otras cosas, información sobre los permisos, el usuario y el grupo al que pertenece el fichero.
sudo	Permite ejecutar comandos como root.
su	Cambia de usuario.
whoami	Muestra el nombre del usuario actual.
groups	Muestra el/los grupos/s a los que pertenece el usuario actual.
groupadd	Añade un nuevo grupo.
groupdel	Borra un grupo.
groupmod	Modifica las características de un grupo.
adduser	Añade un nuevo usuario.
userdel	Borra un usuario.
usermod	Modifica las características de un usuario.
passwd	Asigna o cambia la clave de un usuario.
chown	Cambia el dueño de un archivo.
chgrp	Cambia el grupo al que pertenece un archivo.
chmod	Cambia los permisos.

EJERCICIOS

En ocasiones, la respuesta a los ejercicios no se puede completar únicamente con el material teórico que se proporciona en este capítulo y el alumno debe, por tanto, buscar en otras fuentes complementarias. En los ejercicios de este capítulo se recomienda consultar las páginas `man`.
Las soluciones a los ejercicios se encuentran al final del libro.

Los ejercicios están clasificados según su nivel de dificultad:

	Fácil. El concepto viene explicado en el capítulo.
	Dificultad media. Es necesario relacionar conceptos y/o buscar información en
	Internet. Difícil. Hace falta una investigación concienzuda

1.  Completa la siguiente tabla:

654	
	rwxrw-rw-
	rwxrwxrwx
520	
764	
	r--r-----

2. Crea los grupos **oficina1** y **oficina2**.

3.  Crea los usuarios **paco** y **pablo**. Estos usuarios deben pertenecer únicamente al grupo **oficina1**.

4.  Crea los usuarios **alba** y **nerea**. Estos usuarios deben pertenecer únicamente al grupo **oficina2**.

5.  Como usuario **paco** Crea un fichero con nombre **topsecret.txt** en su directorio de trabajo al que únicamente él tenga acceso, tanto de lectura como de escritura.

6.  Crea otro fichero, también como usuario **paco**, con nombre **ventas_trimestre.txt** al que tengan acceso, tanto para leer como para escribir todos los usuarios que pertenezcan al mismo grupo. Se deben dejar los permisos que haya por defecto para el dueño y para el resto de usuarios. Comprueba como usuario **pablo** que puedes modificar el fichero.

7.  Como usuario **alba**, crea un fichero con nombre **empleados.txt** al que pueda acceder cualquier usuario para leer su contenido, y cualquier usuario del mismo grupo para leer o escribir.

8.  Copia el fichero **empleados.txt** al directorio de trabajo de **alumno** (crea también el usuario alumno si no está creado). Cambia el propietario y el grupo al que pertenece el fichero, ahora debe ser **alumno**.

9.  Como usuario **pablo**, copia un programa del directorio **/usr/bin** al directorio de trabajo con un nombre diferente. Por ejemplo **xclock** se puede copiar como **reloj**. Mira los permisos de este

programa. Comprueba que se puede ejecutar. Puede que sea necesario dar permiso para que otros usuarios distintos al actual puedan ejecutar aplicaciones en el entorno gráfico, basta con ejecutar como administrador: `xhost +`.

10.  Cambia los permisos de **reloj** de tal forma que sólo lo pueda ejecutar el propietario del archivo.

11.  Crea el usuario **modesto**, perteneciente a **oficina2**. Dentro de su directorio de trabajo, crea un directorio de nombre **compartido_con_todos**.

12.  Cambia de usuario en el entorno gráfico (botón **salir** y botón **cambiar de usuario**) y entra como modesto. Crea con **OpenOffice.org Calc** los ficheros **telefono_contactos.ods** , **gastos_marzo.ods** y **sueldos.ods**. Inserta varias entradas en cada uno de los ficheros y grábalo todo en el directorio **compartido_con_todos**.

13.  Da permiso de lectura a la carpeta **compartido_con_todos** y a todos los ficheros que contenga para todos los usuarios.

14.  Restringe el acceso de escritura sobre el fichero **telefono_contactos** para que sólo lo puedan modificar los usuarios del grupo al que pertenece su propietario.

15.  Cambia los permisos de **gastos_marzo** para que sólo pueda modificarlo su propietario y leerlo cualquiera del mismo grupo.

16.  Cambia los permisos de **sueldos** para que sólo su dueño tenga acceso a él, tanto para lectura como para escritura.

17.  Si un usuario tiene permiso de lectura sobre un fichero pero ese fichero se encuentra dentro de un directorio sobre el que no tiene permiso de lectura, ¿podrá leer el fichero?, haz la prueba.