

CURSO GUADALINEX

La Shell Bash



Juan Alonso - Fermín Rubio - Paco Villegas

22 de abril de 2004

Índice general

1. La Shell Bash	5
1.1. La Shell Bash	5
1.1.1. ¿Qué es una shell?	5
1.1.2. Características básicas de la Shell.	6
1.1.3. Variables de entorno de la Bash	6
1.1.4. Ficheros de inicio de la bash	7
1.1.5. Personalizando el Prompt	8
1.1.6. Los Alias	9
1.1.7. Historia de órdenes.	10
1.1.8. Los Builtins (Órdenes internas)	10
1.2. Redirección	10
1.2.1. Redirección de la salida (>)	11
1.2.2. Redirección de la entrada (<)	11
1.2.3. Tuberías	12
1.3. Comandos de la Shell	12
1.3.1. Comandos simples	12
1.3.2. Listas de comandos	12
2. Comandos básicos de Unix/Linux	15
2.1. Introducción	15
2.1.1. Convenciones en cuanto a la sintaxis	17
2.1.2. Comodines	17
2.2. Resumen de comandos	18
2.2.1. Ayuda	18
2.2.2. “Construir” comandos	18
2.2.3. Gestión de usuarios y grupos	18
2.2.4. Manipulación de archivos y directorios	18
2.2.5. Localización de archivos	18
2.2.6. Procesamiento de archivos	19
2.2.7. Guardar y comprimir ficheros	19
2.2.8. Procesos de control	19
2.2.9. Control de usuarios	20
2.2.10. Administrar ficheros	20
2.2.11. Comunicaciones y redes	20
2.2.12. Comandos de Impresión	20
2.2.13. Módulos del kernel	20
2.2.14. Varios	20
2.3. Algunos ejemplos	21
2.3.1. “Construir” comandos	21
2.3.2. Manipulación de archivos y directorios	22
2.3.3. Localización de archivos	24
2.3.4. Procesamiento de archivos	26

2.3.5.	Empaquetando y comprimiendo ficheros.	26
2.3.6.	Control de tareas	29
2.3.7.	Administrar ficheros	34
2.3.8.	Comunicaciones y redes.	36
3.	Programa Midnight Commander	41
3.1.	Introducción	41
3.1.1.	Inicio de una sesión	41
3.1.2.	Soporte de Ratón	42
3.1.3.	Teclas	43
3.2.	Barra de Menú	44
3.2.1.	Menús Izquierdo y Derecho	44
3.2.2.	Menú de Archivo	46
3.2.3.	Menú de Utilidades	50
3.2.4.	Menú de Opciones	52
3.3.	Barra inferior	57
3.4.	Ejecutando Comandos del Sistema Operativo	57
3.4.1.	El Comando cd Interno	57
3.5.	Sistema de Ficheros Virtual	57
3.5.1.	Sistema de Ficheros FTP	57
3.5.2.	Sistema de Archivos Tar	58
3.5.3.	Sistema de Ficheros de Red	58
4.	Guadalinex como cliente de red	61
4.1.	Introducción	61
4.2.	Otros navegadores Web	61
4.3.	Telnet y ssh	63
4.3.1.	Acceso remoto: telnet	63
4.3.2.	ssh: una solución más segura	64
4.3.3.	Conectar en modo comando y (gráfico)	65
4.4.	FTP y SFTP	65
4.4.1.	ftp	65
4.4.2.	sftp	67
4.4.3.	gFTP	68
4.5.	Samba	72
4.6.	Cajón “de-sastre”	73
4.6.1.	Gnome-netinfo	73
4.6.2.	ettercap	75

Capítulo 1

La Shell Bash

Tras la instalación de todos los programas educativos preguntaremos una y otra vez “ TODO ESTÁ EN EL MENÚ, ¿NO?”. Haremos eso cada vez que el *pringao* nos intente explicar qué son ficheros, carpetas y chorradas de esas. Si nos intenta enseñar una ventana negra en la que hay que ¡¡ESCRIBIR!! (sí, amigos, en pleno siglo 21 hay que escribir cosas) y además ¡¡EN INGLÉS!! le diremos que no lo entendemos y que nos ponga eso en el menú. Si nos dice que no se puede poner eso en el menú haremos referencia a que creíamos que él sabía más de informática... (*Pringao Howto* (o *Windows-es-fácil-Howto*), SANTIAGO ROMERO AKA NOP/COMPILER)

1.1. La Shell Bash

1.1.1. ¿Qué es una shell?

Básicamente, una *shell* (la traducción al pie de la letra de caparazón o concha no parece muy adecuada) es un procesador de órdenes que sirve para ejecutar comandos. En un principio, cuando no había entornos de ventanas, era la forma de poder comunicar nuestras órdenes al sistema operativo. Además de para ejecutar comandos, la shell de Unix/Linux sirve como lenguaje de programación y para combinar comandos, formando otros más complejos.

Específicamente, *bash* (la shell que presenta Linux por defecto) es un intérprete de un lenguaje de órdenes compatible con *sh* (una shell anterior) que ejecuta órdenes leídas desde la entrada estándar¹ o desde un fichero. Bash también incorpora características útiles tomadas de otras shells como la Korn y la C shell (*ksh* y *csh*).

El nombre de bash viene de *Bourne Again Shell*². Bash está pensado con la intención de ser una implementación conforme con la especificación POSIX de Shell y Herramientas, de la IEEE (Grupo de Trabajo 1003.2 de la IEEE).

Ya que hemos mencionado POSIX, debemos decir que es un estándar³ (normas escritas en papel, para que lo veamos de una forma más práctica) que pretende definir un Sistema Operativo Abierto⁴ definido por la IEEE (que es una organización internacional de estándares). Linux intenta cumplir con los estándares POSIX.

Para poder estudiar más sobre la shell bash:

- Los howtos traducidos en el INSFLUG.
- La completa página man de la shell Bash

¹Más adelante veremos más sobre esto. Por ahora, podemos identificar la entrada estándar con el teclado y la salida estándar con la pantalla.

²Algo así como la shell Bourne viene de nuevo, aunque una traducción más libre quedaría como “la Bourne Shell Contraataca”.

³Los estándares de derecho (de iure), emitidos por organismos independientes y reconocidos, son importantes porque permiten la independencia de un determinado fabricante y fomentan la interoperabilidad de distintos sistemas. Un “estándar” de hecho (de facto), simplemente puede reconocer el monopolio de un fabricante.

⁴Abierto en el sentido de no perteneciente a ninguna empresa o grupo y con unas reglas claras para poder operar con él.

- *Bash Reference Manual* Bash Reference Manual <http://www.gnu.org/manual/bash-2.05a/>

1.1.2. Características básicas de la Shell.

El funcionamiento de la shell es el siguiente:

1. Lee la entrada desde teclado o desde un fichero.
2. Divide la entrada en palabras y operadores, obteniendo los comandos.
3. Realiza las expansiones correspondientes y las redirecciones de salida.
4. Ejecuta la o las órdenes.
5. Espera (opcionalmente) a que terminen las órdenes y devuelve un valor de estado de finalización. El valor de estado 0 (cero) significa finalización sin errores y un valor distinto de cero indica el código de error producido.

1.1.3. Variables de entorno de la Bash

La shell utiliza las variables de entorno para *afinar* ciertos detalles del comportamiento del sistema. Algunas de estas variables de entorno, ya predefinidas, que utiliza bash son:

HOME El directorio de comienzo del usuario.

PATH Una lista de directorios separados cada uno de ellos por el carácter dos puntos (:) que nos indica en qué directorios busca la shell para encontrar los comandos. Escoge el comando que primero encuentre, en caso de que pueda encontrarse en varios sitios. Si no lo encuentra dentro de esta lista de directorios, nos devolverá un error con el mensaje “Comando no encontrado” o “command not found”.

PS1 El prompt (o indicador de inicio) que presenta la bash al usuario.

PWD El directorio de trabajo actual.

Para ver el contenido de una variable concreta basta con teclear:

```
$echo $nombre_var
```

➡ **Para practicar:** Comprobar el valor de cada una de las variables anteriores. Por ejemplo, el valor de la variable PATH en mi máquina y para el usuario que el comando es:

```
$ echo $PATH
/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R6/bin:/usr/local/sbin:
/usr/local/bin:/usr/games:/usr/local/java/j2re1.4.1/bin
```

Podemos también definir nuestras propias variables mediante las órdenes:

\$MIVAR=valor (damos valor a la variable de entorno MIVAR) Por ejemplo:

```
$miedad = 21
```

```
$minombre = "pepe pinto"
```

define dos variables cuyos contenidos son explícitos.

Si pusiéramos como valor de la variable un comando, por ejemplo ls

```
$listado=ls
```

podríamos invocarlo de la siguiente forma:

```
$$listado
```

(el primer símbolo de \$ es el prompt y el segundo sirve para obtener el valor de la variable).

En cualquier momento podemos ver el valor de todas las variables de entorno definidas en nuestra shell con el comando `set`.

Con `$export nombre_var` (exportamos la variable para que sea visible en esta shell y todos los procesos hijos⁵ de esta shell)

➔ **Para practicar:** ejecuta

```
$echo "me llamo " $minombre " y tengo " $miedad " años"
$set|less
```

1.1.4. Ficheros de inicio de la bash

Dependiendo de la shell con que entre el usuario al sistema, se ejecutan una serie de ficheros que le configuran su entorno de trabajo. Existen unos ficheros generales que se ejecutan para todos los usuarios que entran al sistema con una misma shell (como por ejemplo el `/etc/profile` para las shell Bourne y Korn), y otros específicos para cada usuario y que se encuentran en su directorio HOME. Estos ficheros de inicialización son utilizados para establecer el camino de búsqueda de ficheros ejecutables, establecer protección por defecto de los ficheros que se creen, tipo de terminal desde el que se trabaja y otras variables de entorno. Estos ficheros son:

/etc/profile con él configuramos información de entorno de usuario para todos los usuarios del sistema. Es del root. Se lee una sola vez cuando se inicia el sistema y, dependiendo de la distribución, en él se establecen:

- el prompt por defecto
- el path por defecto
- el tamaño máximo de los ficheros que podemos crear
- los permisos por defecto para los ficheros que creemos.
- tamaño de los ficheros de historial
- ...

~/ .bash_profile permite introducir información específica para cada usuario para las shell del sistema. Se lee sólo una vez cuando el usuario accede en el sistema. En él hay una llamada que hace que se ejecute `.bashrc`.

~/ .bashrc información/configuración específica de un usuario para la shell bash. Puede modificar los valores que se cargaron para el conjunto de usuarios. Su contenido se lee cada vez que se entra en el sistema y cada vez que se abre un nuevo shell bash.

Cuando la bash es llamada como una shell interactiva⁶ de comienzo, lo primero que hace es leer y ejecutar los comandos que se encuentran en el fichero `/etc/profile`. Después, pasa al fichero `~/ .bash_profile`.

Cuando se trata de una shell interactiva pero que no es de comienzo, el fichero que ejecuta es `~/ .bashrc`.

El fichero `/etc/profile`, como hemos comentado antes, se encarga de que tengamos el entorno listo para trabajar, se ejecuta al entrar cualquier usuario del sistema y es modificable sólo por el superusuario, mientras que los que se encuentran bajo el directorio HOME (~) de cada usuario son configurables y personalizables por éstos. Veamos posible contenido para este fichero:

⁵Ya hablaremos sobre los procesos. Por ahora, sepamos que la shell ejecuta los comandos que le introducimos como procesos hijos. La shell se encarga de que nazcan, realicen su tarea y mueran cuando finalicen.

⁶La que vemos normalmente y le introducimos comandos. Cuando termina el comando, nos presenta otra vez el símbolo del sistema para continuar.

```
# /etc/profile: system-wide .profile file for the Bourne shell (sh(1))
# and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).
PATH="/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games"
if [ "$PS1" ]; then
  if [ "$BASH" ]; then
    PS1='\u@\h:\w\$ '
  else
    if [ "`id -u`" -eq 0 ]; then
      PS1='# '
    else
      PS1='$ '
    fi
  fi
fi
export PATH
# Añadido por /usr/bin/eurocastellanizar
if [ -f /etc/language-euro-es ]; then source /etc/language-euro-es; fi
# fin cambios
umask 022
```

1.1.5. Personalizando el Prompt

Como vimos antes, el valor de la variable PS1 es lo que se nos presenta en el prompt del sistema.

El valor predeterminado es PS1="\u@\h:\w\\$". Podemos verlo ejecutando el comando

```
$echo $PS1
```

Existen algunos valores que podemos utilizar para modificarlo a nuestro antojo.

\d la fecha en el formato "Día-Semana Mes Día" (ejemplo, "Tue May 26") en inglés	\v la versión de bash (e.g., 2.00)
\e un carácter de escape (ESC) ASCII (033)	\V la distribución de bash, versión + nivel de parches (e.g., 2.00.0)
\h el nombre del computador hasta el primer ‘.’	\w el directorio de trabajo en curso
\H el nombre del computador con dominio completo	\W el nombre base del directorio de trabajo
\n salto de línea	!\ el número de historia de esta orden
\r retorno de carro	\# el número de orden de este comando en la shell actual
\s el nombre del shell. El nombre base del ejecutable de la shell (la porción que sigue a la última barra inclinada)	\\$ si el UID ⁷ efectivo es 0 (el super-usuario root), un #. Si no lo es, un \$
\t la hora actual en el formato de 24 horas HH:MM:SS	\nnn el carácter correspondiente al número octal nnn
\T la hora actual en el formato de 12 horas HH:MM:SS	\ una barra inclinada invertida
\@ la hora actual en el formato de 12 horas con indicador AM/PM	\[empieza una secuencia de caracteres no imprimibles, que pueden emplearse para insertar una secuencia de control del terminal en el indicador
\u el nombre de usuario del usuario actual	\] termina una secuencia de caracteres no imprimibles

⁷Identificador de usuario.

➡ Para practicar:

- ejecuta `$echo $PS1`
- haz un `su` a `root` y repite la orden anterior
- Comprobar el resultado de establecer `PS1="\t [\u@\h:\W] "`

Para ampliar sobre este tema se puede consultar el HOWTO: *Bash Prompt COMO*.

1.1.6. Los Alias

Los alias permiten que una cadena (normalmente un comando complejo) se sustituya por una sola palabra cuando se emplee esta como la primera palabra de una orden simple. ¿Qué conseguimos con esto? Economía de escritura, pues con sólo teclear una palabra estaremos realizando una labor más compleja con menos caracteres que teclear.

Los alias se crean y muestran con la orden `alias`, y se quitan con la orden `unalias`.

La sintaxis para definirlos es⁸:

```
alias [-p] [nombre[=valor] ...]
```

Por ejemplo, con

```
$ alias ll="ls -laF"
```

conseguimos con sólo dos caracteres ("ll", mnemónico de Listado Largo) realizar la misma función que con siete (`ls -laF`).

Para eliminar un alias utilizamos

```
unalias [-a] [name ... ]
```

```
$ unalias ll
```

elimina el alias creado anteriormente.

El shell mantiene en memoria una lista de los alias definidos que podemos visualizar con la orden `alias`.

Cuando se ejecuta una orden el shell mira si la primera palabra, si no está entrecomillada, tiene un alias. Si es así, la palabra se reemplaza con el texto del alias. El nombre del alias y el texto por el que se reemplaza, pueden contener cualquier entrada válida para el shell, incluyendo metacaracteres, con la excepción de que el nombre del alias no puede contener un `=`. La primera palabra del texto de reemplazo se comprueba también para ver si es un alias, pero si es un alias idéntico al que se está expandiendo, no se expande una segunda vez. Esto significa que uno puede poner un alias `"ls"` a `"ls -F"`, por ejemplo, y `bash` no intenta expandir recursivamente el texto de reemplazo. Si el último carácter del valor del alias es un blanco, entonces la siguiente palabra de la orden que sigue al alias también se mira para la expansión de alias.

No hay ningún mecanismo para poder usar argumentos en el texto de reemplazo⁹.

Si queremos definir un alias de forma permanente tendremos que hacerlo en el fichero `~/.bashrc`, de lo contrario se borrará de la memoria cuando salgamos del sistema.

➡ Para practicar: Crear un alias que permita que funcione el comando `cd..` .

```
$alias cd..="cd .."
```

Probar que funciona y eliminarlo después con

```
$unalias cd..
```

⁸Los corchetes indican que los parámetros son opcionales y no se tienen que escribir.

⁹Si se necesitan, debería emplearse mejor una función del shell.

1.1.7. Historia de órdenes.

Cuando se habilita la opción `-o history` (opción que ya está normalmente por defecto¹⁰), el shell da acceso a la historia de órdenes: lista de órdenes tecleadas con anterioridad.

El texto de los últimos mandatos se guarda en una lista de historia. El shell almacena cada orden en la lista de historia antes de la expansión de parámetros y variables (el número de órdenes almacenadas en la lista se define en la variable `HISTSIZE`, por omisión 500). En el arranque, la historia se inicia a partir del fichero nombrado en la variable `HISTFILE` (por omisión `~/.bash_history`¹¹). `HISTFILE` se trunca, si es necesario, para contener no más de `HISTFILESIZE` líneas.

➔ **Para practicar:** Comprobar los valores (por defecto) anteriores con

```
$ echo $HISTSIZE
$ echo $HISTFILE
```

Para visualizar la lista:

```
$history
o mejor
$history | less
```

Podemos recorrer las órdenes anteriores con las teclas de flecha hacia arriba y flecha hacia abajo.

➔ **Para practicar**

1. Ejecutar el último comando `echo` con

```
$!e
```

2. Tras la salida del comando

```
$history
```

ejecutar

```
$!comando_número_línea
```

1.1.8. Los Builtins (Órdenes internas)

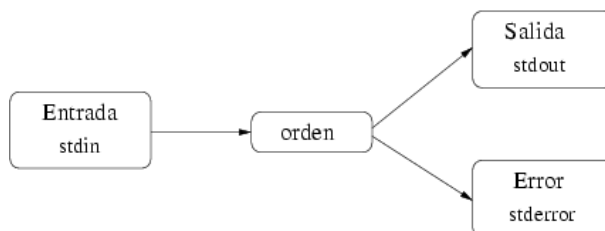
Los *builtins* (u órdenes internas) son comandos que ya vienen implementados dentro de la propia `bash`. No hay que buscar un ejecutable externo porque la propia `bash` lo lleva incorporado. Por ello, se ejecutan mucho más rápido. Algunos de ellos son:

cd Que como ya sabemos nos cambia de directorio de trabajo.

pwd Que nos indica en qué directorio estamos situados.

1.2. Redirección

La ejecución de un comando generalmente responde al siguiente esquema:



¹⁰Podemos ponerla con `set -o history`

¹¹Podemos ver con un editor el contenido de este fichero

En la figura se observa que el comando, si necesita algún dato de entrada, lo habitual es que lo reciba a través del teclado, que es la entrada por defecto (*stdin*). Si la ejecución del comando conlleva la devolución de alguna información, esta se envía a la pantalla, que es el dispositivo de salida por defecto (*stdout*). Si se produce un error en la ejecución del comando, el mensaje correspondiente se envía por el dispositivo de errores por defecto (*stderr*), que es también la pantalla.

Este comportamiento puede modificarse con lo que denominamos **redirección**.

1.2.1. Redirección de la salida (>)

Supongamos que deseamos guardar la salida del comando `dmesh` para posteriormente analizarla con tranquilidad. Para ello basta con ejecutar:

```
$ dmesh >mensajes.txt
```

Con ello, la salida que hubiera aparecido por pantalla, se ha guardado en el fichero `mensajes.txt`.

Si el fichero especificado existe, se trunca a longitud cero, es decir, se borra previamente su contenido. Si no existe, se crea.

Posteriormente podremos ver el contenido del fichero.

Añadir a la salida redirigida (>>)

Como hemos comentado más arriba, la redirección de salida (>) borra previamente el contenido del fichero especificado. Si queremos añadir la salida conservando el contenido anterior del fichero, debemos utilizar el signo (>>).

Por ejemplo, el comando `df -h` devuelve información del espacio de disco ocupado en el sistema. Para hacer un seguimiento del consumo de disco podemos ejecutar periódicamente el comando que sigue y no perderemos los valores que vamos almacenando, sino que se irán acumulando en el fichero.

```
$df -h >>consumo_disco.txt
```

1.2.2. Redirección de la entrada (<)

La redirección de la entrada hace que el comando tome como entrada el fichero especificado.

Por ejemplo, con la orden

```
$ cat </etc/passwd
```

el comando `cat` (que muestra por salida estándar lo que recibe por la entrada estándar) recibe como entrada el fichero `/etc/passwd` y con esta otra

```
$ mail root <mensaje_a_root.txt
```

enviamos un mensaje al root, el fichero `mensaje_a_root.txt` que `mail` recibe como entrada.

➡ Para practicar: Redirección

1. Al final de cada apartado, comprobar el resultado, por ejemplo con

```
$more prueba
```

- a) Crear un fichero de texto

```
$echo "Hola Mundo" >prueba
```

- b) Añadir al final "cruel"

```
$echo "cruel" >>prueba
```

- c) Almacenamos en `prueba` los ficheros (ocultos y directorios) de nuestro `$HOME`, en una columna y ordenados por tiempo de creación

```
$ls -alt >prueba
```

- d) ¿Y si lo ordenamos alfabéticamente?

```
$sort <prueba >prueba_o
```

- e) ¿Qué pasa ahora?
\$more prueba >>prueba

1.2.3. Tuberías

Una tubería es una secuencia de una o más órdenes separadas por el carácter "|" (barra vertical). El formato de una tubería es:

```
orden1 [ | orden2 ... ]
```

La salida estándar de `orden1` se conecta a la entrada estándar de `orden2`. Esta conexión se realiza antes que cualquier redirección especificada por la orden.

Cada orden en una tubería se ejecuta como un proceso separado (esto es, en una subshell).

Por ejemplo, para contar el número de líneas de un fichero, ejecutaríamos:

```
$cat fichero | wc -l
```

Su explicación es que con el comando `cat` visualizamos el contenido del fichero, pero esta salida, en vez de ir a la pantalla, se mete en la tubería que va hacia la entrada de la orden `wc` (de *word count*, contador de palabras) que con su opción `-l` nos dice el número de líneas que ha leído.

En esta característica se apoya gran parte de la elegancia de los sistemas Unix/Linux. Con comandos simples podemos llegar a realizar acciones verdaderamente complejas.

➡ Para practicar

1. La orden `sort` ordena alfabéticamente líneas de ficheros de texto. Crea, por ejemplo con `gedit`, un fichero de texto `alumnos.txt`, con los apellidos y nombres de un grupo de alumnos(no los introduzca ya ordenados)
Ejecuta la orden

```
cat alumnos.txt | sort
```


Ahora ejecuta

```
cat alumnos.txt | sort >alumnos_ordenados.txt
```


y visualiza el fichero `alumnos_ordenados.txt`
2. El comando `grep` envía a la salida estándar (o a la especificada) las líneas que concuerden con un patrón.
Ejecuta las ordenes

```
ls /etc >dir_etc
```



```
cat dir_etc | grep sh
```

1.3. Comandos de la Shell

1.3.1. Comandos simples

Un comando simple es la clase de comandos que nos encontramos más frecuentemente. Consiste en una secuencia de palabras separadas por blancos. La primera palabra especifica el comando a ejecutar, seguido por unas opciones (como por ejemplo "`ls -l`"¹²) o unos argumentos (`cat /etc/profile`¹³).

1.3.2. Listas de comandos

Una lista de comandos es una secuencia de comandos simples o tuberías separados por uno de los operadores `;`, `&`, `&&`, or `|`, y terminada por `;`, `&`, o retorno de carro.

Si un comando se termina con el operador de control `&`, la shell ejecuta el comando de forma asíncrona en una subshell. Esto se conoce como ejecutar el comando en segundo plano (*background*). En este caso,

¹²En este caso la opción es `l`, y el guión sirve para indicar que lo que viene detrás es una opción. Una opción normalmente modifica el comportamiento de un comando.

¹³El fichero `/etc/profile` es un argumento. Los argumentos normalmente indican sobre qué actúa el comando.

la shell no espera a que el comando termine e inmediatamente aparece otra vez el indicador de inicio (prompt), mientras el comando se ejecuta por detrás de ella.

Los comandos separados por ';' se ejecutan secuencialmente, uno detrás de otro.

```
$ comando1; comando 2
```

La shell espera a que terminen los comandos en su turno correspondiente. Por ejemplo:

```
$ cd /home/Thales; ls
```

primero se posiciona en el subdirectorio /home/Thales y después lista los ficheros de ese directorio.

Los operadores de control permiten ejecuciones condicionales.

El efecto de

```
comando1 && comando2
```

es que comando2 se ejecutará si y sólo si comando1 termina de forma satisfactoria (devuelve un código de cero).

En cambio, en la lista

```
comando1 || comando2
```

el comando2 se ejecutará si y sólo si comando1 falla (devuelve un código distinto de cero).

➡ Para practicar: comando tee

Podemos conseguir guardar la salida de un comando en un fichero y dirigirla también a la salida estándar usando el comando tee. El nombre del comando viene de que se comporta como una T de fontanería. El caudal que llega por una rama, pasa por la T y sale por los otros dos orificios.

Por ejemplo, supongamos que deseamos ver los usuarios de nuestra máquina y guardarlos en un fichero ordenados, escribiremos:

```
$ cut -f1 -d: /etc/passwd | sort | tee usuarios.txt
```

Explicuemos un poco el comando:

- `cut -f1 -d: /etc/passwd` → Obtiene del fichero /etc/passwd el primer campo (f1 de *field*), especificando como separador de campo (-d de delimitador) el carácter :
- `sort` → ordena alfabéticamente los nombres de usuario
- `tee usuarios.txt` → guarda el resultado en el fichero usuarios.txt y además lo dirige a la salida estándar. Son las dos salidas de la T.

Capítulo 2

Comandos básicos de Unix/Linux

Como regla general, se podría decir lo siguiente: "Todo lo que se puede hacer en modo gráfico, se puede hacer también en modo texto, a base de comandos. Pero no todo lo que se puede hacer en modo texto, se puede hacer en modo gráfico". (*FAQ sobre Linux para principiantes* - es.comp.os.linux)

2.1. Introducción

En este apartado veremos los comandos más usuales de Linux. Ni están todos ni tiene sentido ver todas y cada una de las opciones de ellos. Para ampliar sobre algunos de ellos os remitimos a las páginas de ayuda de cada comando, a las infopages, así como a los manuales comentados en la primera entrega. Y sobre todo el uso que vayamos haciendo de ellos.

Ante la duda de si es necesario conocer los comandos la respuesta es clara: sí. **Al menos los más usuales.** Creemos que es necesario saber qué se puede hacer aunque a veces necesitemos la chuleta con la orden que nos permita saber cómo hacerlo. Si sólo nos dedicásemos a usar Linux como un entorno de oficina es posible que el número de comandos necesarios fuese mínimo, pero si deseamos administrar nuestro sistema Linux no queda más remedio que ampliar el conocimiento sobre ellos.

El tema sobre comandos se ha dividido en dos partes: por un lado tenéis una referencia rápida de qué hace cada uno. Por otro, se han analizado con más detalle aquellos que tienen más utilidad.



Recordar de nuevo la facilidad de uso que representa la autocompletación de comandos. Cuando queramos ejecutar un comando, no tenemos que conocer su nombre exacto ni el del fichero que le pasamos como parámetro para poder trabajar con él. Así, por ejemplo, si deseamos saber qué comandos comienzan por las letras `wh` escribiremos

```
$ wh
```

y tras pulsar la tecla **[Tab]** dos veces, nos aparecerán las concordancias encontradas en nuestro `path`.

```
whatis      which      whiptail   whoami
whereis     while      who        whois
```

Si la concordancia es única, se autocompletará el comando pulsando una sola vez la tecla.



Para “abrir” boca un mini resumen de la equivalencia entre los comandos más usuales del DOS y los de Linux

Cuadro 2.1: Del DOS a Linux

Descripción	DOS/Windows	Linux
Ayuda	help	man
Copiar ficheros	copy	cp
Contenido de un fichero	type	cat
Renombra un fichero	ren	mv
Mover ficheros/directorios	move	mv
Lista archivos	dir	ls
Borra archivos	del	rm
Borra la pantalla	cls	clear
Terminar una sesión	exit	exit
Crea un directorio	mkdir	mkdir
Borra un directorio	rmdir	rmdir
Cambiar de directorio	cd	cd
Cambiar atributos de ficheros	attrib	chmod
Cambiar la fecha	date	date
Compara ficheros	fc	diff
Memoria libre	mem	free
Imprimir un fichero	print	lpr
Editar un fichero	edit	mcedit
Mandar paquetes	ping	ping
Configuración interfaz de red	ipconfig	ifconfig
Configuración interfaz de red	winipcfg	ifconfig

➔ Para practicar: Paquete mtools

El paquete `mtools` se instala por defecto, trae los comandos: `mcopy`, `mdir`, ... similares a los de MS-DOS, la única diferencia es que hemos de anteponer una `m` al comando. Por ejemplo:

```
$mcopy a:* /home/thales
```

copia el contenido del floppy en el subdirectorio indicado. Es interesante resaltar que **para usarlos no es necesario montar el floppy**.

El fichero de configuración de este paquete es `/etc/mtools.conf`. En general no hay que modificarlo nunca pero si algo no funciona bien puede que tengamos que ajustarlo a nuestro sistema.

1. Comprobar qué comandos componen el paquete usando
\$info mtools
2. Formatear un disquete con la orden
\$mformat a:
3. Listar el contenido del disquete con `mdir`
\$mdir a:
4. Copiar el fichero¹ `~/ .bashrc` al disquete usando las `mtools`:
\$mcopy ~/ .bashrc a:
5. Montar el disquette creado con la orden `mount` y comprobar que la copia se ha realizado bien.
6. El sistema mantiene una lista de los sistemas de ficheros montados actualmente, en el fichero `/etc/mtab`. Se puede ver el contenido del fichero utilizando el comando `mount` sin argumentos:
\$ mount

¹El caracter `~` referencia el home de ese usuario.

2.1.1. Convenciones en cuanto a la sintaxis

La sintaxis común a todos los comandos es:

```
comando [opciones][parámetro_1] parametro_2 ...
```

donde las opciones y los parámetros son opcionales si van entre corchetes e imprescindibles cuando van solos². Si además algún parámetro va seguido de tres puntos suspensivos es para indicar que pueden incluirse cuantos parámetros de ese tipo se quieran.

Las opciones, en general se le pasan al comando como una serie de valores precedidos por un guión, por ejemplo:

```
$ df -h -l
Filesystem Size Used Avail Use% Mounted on
/dev/hdc1 2.3G 1.5G 754M 66% /
/dev/hdc5 1.5G 728M 740M 50% /datos
/dev/hda5 994M 318M 676M 32% /mnt/d
```

nos informa de la utilización del espacio en disco del sistema de ficheros. Al pasarle como opciones:

-h (**--human-readable**) añade una letra indicativa del tamaño, como M para megabytes, a cada tamaño

-l hace que se limite el listado a los sistemas de ficheros locales, no en máquinas remotas que pudieran estar montados por NFS, por ejemplo.

En general, esta forma de poner las opciones es equivalente a poner un solo guión y los valores de las opciones a partir del guión como una cadena de caracteres. Así la orden anterior es equivalente a escribir:

```
$ df -hl
```

2.1.2. Comodines

De igual manera que en sistemas DOS³, en Linux se puede hacer uso de comodines para hacer referencia a nombres de archivos, las posibilidades son:

* igual que en sistemas DOS, el comodín se sustituye por cualquier cadena de caracteres

? la interrogación también tiene el uso habitual, se sustituye por cualquier carácter, pero sólo uno.

[..] El uso de corchetes permite hacer referencia a un solo carácter, las posibilidades son:

- hacer referencia a un solo carácter pero con la obligatoriedad de estar comprendido en los valores listados entre corchetes:

```
$ ls ed[89]linux
```

en este caso se mostrarían los ficheros cuyo nombre sea de la forma ed9linux o ed8linux

- hacer referencia a un rango de valores separados por un guión:

```
$ ls ed[7-9]linux
```

en esta caso se mostrarían todos los ficheros cuyo nombre fuese de la forma ed7linux, ed8linux o ed9linux.

2

■ Ejecutar `$ man free` para comprobar que todas las opciones y parámetros son opcionales.

■ El comando `write`, que sirve para enviar un mensaje a otro usuario conectado al sistema, necesita al menos el argumento `user`.

³Más bien DOS lo tomó de Unix. Recordemos que Unix es de finales de los años 60.

- invertir el rango anteponiendo el signo !

```
$ ls ed[!1-8]linux
```

en este caso se mostrarán todos los ficheros con tercer carácter arbitrario y distinto de los números 1 al 8 (ambos inclusive)

Se pueden mezclar entre ellos, así:

```
$ ls ed?[7-9]*
```

mostraría todos los ficheros cuyo nombre de fichero verifique:

1. Sus dos primeros caracteres son “ed”
2. El tercer carácter puede ser cualquiera
3. El cuarto carácter es un número comprendido entre 7 y 9
4. El resto de caracteres pueden ser cualesquiera

2.2. Resumen de comandos

2.2.1. Ayuda

apropos Busca las páginas de ayuda que contienen la clave que especifiquemos

info Permite el acceso a la ayuda *online* de un comando

man Para visualizar las páginas man

whatis Busca palabras completas en la base de datos *whatis*

2.2.2. “Construir” comandos

alias Se usa para definir abreviaturas para los comandos largos. También nos muestra una lista con las abreviaturas ya definidas

type Indica cómo interpretaría la shell el comando pasado como argumento

unalias Para eliminar las abreviaturas que previamente hemos definido con *alias*

2.2.3. Gestión de usuarios y grupos

chgrp Cambia el grupo de un archivo

chmod Cambia los permisos de acceso de ficheros

chown Cambia el usuario y grupo propietarios de ficheros

groups Muestra los grupos en los que está un usuario

addgroup Crea un nuevo grupo

delgroup Borra un grupo

newgrp Para pasar a tener los derechos de un grupo

passwd Para asignarle la contraseña a un usuario

umask Establece la máscara de creación de ficheros

adduser Para añadir un usuario

userdel Permite eliminar un usuario

2.2.4. Manipulación de archivos y directorios

cd Cambia el directorio de trabajo

cp Copia ficheros y directorios

file Determina el tipo de un fichero

ls Nos muestra el contenido de un directorio (*dir*, *vdir* son versiones de *ls*)

ln Permite crear enlaces entre ficheros

mkdir Crea directorios

mv Mueve (renombrar) ficheros

rm Borra ficheros o directorios

rmdir Borra directorios vacíos

pwd Muestra el nombre del directorio de trabajo actual

touch Actualiza la fecha de un archivo a la actual

2.2.5. Localización de archivos

find Busca ficheros en un árbol de directorios

locate Permite localizar archivos basándose en una base de datos que se va actualizando periódicamente

whereis Localiza los ficheros binarios, fuentes y páginas del manual correspondientes a un programa

which Muestra el path del archivo de comandos pasado como argumento

2.2.6. Procesamiento de archivos

cat Concatena archivos y también muestra su contenido usando la salida estándar

cmp Compara dos archivos

csplit Divide un archivo en secciones determinadas por líneas de contexto

cut Imprime secciones de líneas de un archivo de entrada

dd Convierte y copia un fichero

diff Busca diferencias entre dos archivos o directorios

expand Convierte las tabulaciones en espacios

fold Permite ajustar las líneas de texto al ancho que especifiquemos

grep, egrep, fgrep Muestran líneas de ficheros que concuerdan con un patrón

head Muestra la parte inicial de un archivo (por defecto 10 primeras líneas)

less Muestra archivos en pantalla de una vez paginando la salida, permite volver atrás

more Filtro que muestra un archivo pantalla a pantalla (es mejor less)

nl Numera las líneas de un archivo que no estén en blanco

paste Combina líneas de ficheros

patch Aplica el comando `diff` actualizando el archivo original. Aplica un “parche”

sed Editor de texto no interactivo

sort Ordena las líneas de archivos de texto

split Divide un archivo en varias partes (por defecto de 1000 líneas en 1000 líneas)

tac Invierte el orden de las líneas de un archivo. Cat al revés.

tail Muestra las últimas líneas (10 por defecto) de un documento

tr Cambia unos caracteres por otros

uniq Borra las líneas duplicadas de un archivo ordenado

wc Muestra el número de bytes, palabras y líneas de un archivo

xargs Construye y ejecuta órdenes desde la entrada estándar

zcat Igual que `cat` pero sobre ficheros comprimidos

zless Actúa como `less` pero sobre archivos comprimidos

zmore Igual que `more` pero sobre ficheros comprimidos

2.2.7. Guardar y comprimir ficheros

compress Comprime (o expande) archivos

gunzip Expande ficheros

gzip Comprime/expande ficheros

tar Para empaquetar y desempaquetar archivos y directorios

uncompress Expande archivos

bzip2 Comprime ficheros con una ratio mejor que los anteriores

bunzip2 Descomprime ficheros comprimidos con `bzip2`

2.2.8. Procesos de control

at Permite planificar la ejecución de tareas

bg Permite ejecutar un proceso interrumpido que está en segundo plano

cron Para planificar órdenes o procesos de forma periódica en el tiempo

fg Sigue con un proceso interrumpido anteriormente, pero en primer plano

free Muestra la cantidad de memoria libre y usada en el sistema

halt Cierra el sistema

jobs Lista la tabla de trabajos en ejecución

kill Termina un proceso

ldd Nos muestra las librerías compartidas que necesitamos para ejecutar un programa

nice Ejecuta un programa con la prioridad de planificación modificada

ps Informa del estado de los procesos

printenv Imprime parte o todo el entorno

pstree Proporciona un árbol de los procesos en ejecución

reboot Reinicia el sistema

shutdown Cierra el sistema

sync Vuelca a disco los *buffers* del sistema de archivos

uname Imprime información del sistema

2.2.9. Control de usuarios

chfn Cambia los datos de un usuario

chsh Cambia el shell

groups Imprime los grupos en los que está un usuario

id Muestra los identificadores de usuario y de grupo

last Muestra los últimos accesos al sistema

passwd Cambia contraseñas

su Ejecuta una shell con identificadores de grupo y de usuario distintos

2.2.10. Administrar ficheros

df Informa de la utilización del espacio de disco en sistemas de ficheros

du Lista el espacio ocupado por los archivos o directorios

fdformat Formatea un disquete

fdisk Manipulador de tablas de particiones para Linux

fsck Chequea y repara un sistema de archivos de Linux

mkfs Construye un sistema de ficheros de Linux

mknod Crea ficheros especiales de bloques o caracteres

mkswap Construye un área de intercambio para Linux

mount Monta un sistema de ficheros

swapoff Deshabilita dispositivos o ficheros de intercambio

swapon Habilita dispositivos o ficheros de intercambio

tty Imprime el nombre del fichero del terminal conectado a la entrada estándar

umount Desmonta sistemas de ficheros

2.2.11. Comunicaciones y redes

finger Proporciona información sobre los usuarios conectados al sistema

mail Programa destinado al envío y recepción de correo

mesg Permite permutar la posibilidad de recibir mensajes de otros usuarios

talk Permite establecer una “charla” con otro usuario

wall Manda un mensaje o un archivo a todos los usuarios que admitan mensajes con **write**

w Muestra qué usuarios están conectados y qué están haciendo

who Muestra información de los usuarios conectados al sistema

write Manda un mensaje a la pantalla de un usuario

2.2.12. Comandos de Impresión

lpq Muestra los trabajos en la cola de impresión

lpr Envía un trabajo a la impresora o pone en cola un trabajo de impresión

lprm Elimina un trabajo de la cola

lpstat Permite comprobar el estado de los trabajos de impresión

2.2.13. Módulos del kernel

depmod computa las dependencias entre módulos

lsmod lista los módulos activos

insmod carga un módulo en el kernel

rmmod descarga un módulo cargable

2.2.14. Varios

cal Calendario

clear Borra la pantalla

date Proporciona o ajusta la fecha y hora del sistema

dmesg Permite ver los mensajes de inicio del sistema

echo Muestra el texto/contenido de la variable

env Muestra el entorno actual de trabajo con todas sus variables

exit Cierra el shell actual

nohup Permite que un comando se ejecute aunque se cierre la sesión, y sin salida a un **tty**

time Tiempo que tarda en ejecutarse un comando

2.3. Algunos ejemplos

En esta sección vamos a ver varios ejemplos de cómo se utilizan algunos de los comandos anteriores⁴. Hay grupos de comandos del resumen anterior que ya se han visto, y, por tanto, no se ponen de nuevo aquí; entre ellos estarían los comandos de impresión, los de gestión de usuarios, ayuda, etc.

Para algunos comandos hemos seguido el convenio de poner:

comando

sintaxis_usual

2.3.1. “Construir” comandos

En el capítulo sobre la Shell Bash ya se ha visto y comentado el funcionamiento de estos comandos. Retomemos algunos aspectos más sobre ellos.

alias

alias [-p] [nombre[=valor] ...]

➔ Para practicar

Un alias es la posibilidad que tenemos de usar un nombre diferente para cualquier comando. Uno de los usos más “típicos” del comando alias consiste en definir en el fichero ~/.bashrc la serie de “alias”

```
alias ls='ls --color -sF'
alias cd.='cd ..'
```

(y otros)

así, cuando ejecutemos el comando ls veremos los ficheros/directorios de distintos colores y podremos usar cd. como sinónimo de cd ..⁵. Antes de ponerlos en el fichero ~/.bashrc debemos practicar con ellos desde la línea de comandos, así, si hemos realmente definido el alias ls anterior y ejecutamos

```
$ ls
```

comprobaremos que, dependiendo de qué tipo de fichero estemos considerando, se ve de distinto color:

```
[root@novo /ppp]# ls
total 16
1 Thalesa          1 ip-up*          1 options
1 arrakis          1 ip-up.local*    0 pap-secrets@
1 averroes         1 _Thalesa*       1 peers/
1 chap-secrets     1 _arrakis*       1 uni2
2 connect-errors   1 _averroes*
1 ip-down*         1 _uni2*
```

type

type comando

Si ahora ejecutamos

```
$ type ls
```

obtendremos:

```
ls is aliased to 'ls --color -Fs'
```

⁴Las páginas man dan una información exhaustiva de los comandos.

⁵Puede que sobre el papel no se vea muy claro, lo que ocurre es que en Linux hay que dejar un espacio entre el cd y los puntos, esto en DOS no importa. Al usar este alias da igual que dejemos el espacio o no.

unalias

```
unalias nombre_alias...
```

Por último, con `unalias` podemos quitar alias, así si ejecutamos

```
$ unalias ls
```

y después

```
$ type ls
```

obtendremos:

```
ls is hashed (/bin/ls)
```

es decir, `ls` se quedaría con las opciones que tiene por defecto.

2.3.2. Manipulación de archivos y directorios

La mayoría de los comandos que aparecen en este grupo son conocidos ya por los que venimos del MSDOS, lo que ocurre es que puede que se nos haya olvidado su nombre completo⁶. Otros tienen un uso tan inmediato que con el pequeño resumen de su función consideramos que es suficiente.

cd

```
cd [directorio]
```

Retomemos a nuestro linuxero THALES, que se encuentra trabajando en su directorio de usuario `/home/Thales`.

Tiene que moverse por el árbol de directorios y desplazarse al directorio raíz, para ello ejecuta:

```
$ cd /
```

Después se mueve a

```
$ cd /etc/sysconfig
```

para ver el contenido de un fichero. Una vez terminada la labor, vuelta a casa

```
$ cd
```

y listo, el sistema lo lleva a `/home/Thales`.

Pero siempre se olvida algo, necesita volver al directorio en el que se encontraba anteriormente (`/etc/sysconfig`) y ejecuta:

```
$ cd -
```

cp

```
cp [opciones] fuente destino
```

Una vez en el raíz, recuerda que tiene que copiar el fichero `/home/Thales/curso/entrega_3.sgml` al subdirectorio `/ed04linux`, para hacer esto escribe:

```
$ cp /home/Thales/curso/entrega_3.sgml /ed04linux
```

file

```
file archivo...
```

Como no recuerda con qué aplicación lo hizo escribe:

```
$ file /ed04linux/entrega_3.sgml
```

y ve en el terminal que:

```
entrega_3.sgml: gzip compressed data, deflated, original filename,
```

```
last modified: Sun Feb 20 20:46:19 2004, max compression, os: Unix
```

con lo que recuerda que ese fichero lo comprimió y que antes de hacer nada con él debe descomprimirlo, tras hacerlo (véase 2.3.5), ejecuta de nuevo:

⁶¿quién se acuerda ya del `edlin`?

```
$ file /ed04linux/entrega_3.sgml
```

y el resultado ahora es:
entrega_3.sgml: exported SGML document text
es decir, es un documento en formato sgml.

ls

```
ls [opciones] [archivo, directorio]
```

Quizás, junto con `cd`, el comando más usado en Linux sea `ls` (o alguna de sus variantes); `ls` muestra el contenido de un directorio en un listado que por defecto está ordenado alfabéticamente. La sintaxis básica es:

```
$ ls [opciones] [archivo, directorio]
```

donde:

opciones las más importantes son:

- a** Muestra todos los archivos (hasta los “ocultos”, los que empiezan por “.”)
- f** Muestra el contenido de los directorios en el orden en el que están almacenados en el disco.
- i** Muestra el inodo de los archivos listados.
- m** Lista los directorios separando los nombres por comas.
- r** Invierte el orden usual de mostrar el directorio
- s** Muestra el tamaño de los archivos.
- t** Ordena los archivos por fecha de creación, primero los más recientes.
- R** Muestra recursivamente el directorio y sus subdirectorios.

Ya hemos visto anteriormente algunos ejemplos de este comando.

mkdir

```
mkdir [-p7] directorio...
```

Continuemos con THALES. Ahora tiene que crear un nuevo directorio en el subdirectorio `/ed04linux` donde guardar los gráficos de la entrega cuatro; tras situarse en `/ed04linux` escribe:

```
$ mkdir -p graficos/entrega_4
```

listo, ya tiene su flamante directorio `entrega_4` (con la opción `-p` se ha creado, si no existía, el subdirectorio `graficos`). Después ejecuta `cd` para situarse de nuevo en su directorio de usuario.

mv

```
mv [-i8] origen destino
```

pero necesita mover el fichero `peguin.png` que se encuentra en `~/curso` al nuevo directorio creado y entonces usa:

```
$ mv curso/penguin.png /ed04linux/graficos/entrega_4/
```

⁷Con esta opción crea los directorios intermedios en caso de que no existan

⁸Pregunta antes de sobrescribir un archivo de destino que ya exista.

rm

```
rm [opciones] archivo
```

Ahora recuerda que ya no necesita el fichero original `entrega_3.sgml` (estaba en `/home/Thales/curso`) y decide borrarlo:

```
$ rm curso/entrega_3.sgml
```

rmdir

```
rmdir directorio...
```

Se da cuenta de que ni ese directorio (`/home/Thales/curso/`) ni su contenido los va a usar más y decide borrarlo, para esto escribe:

```
$ rmdir /home/Thales/curso/
```

y recibe un error del sistema, y es que `rmdir` sólo borra directorios vacíos (¿qué se la va a hacer? a grandes males...), así que escribe:

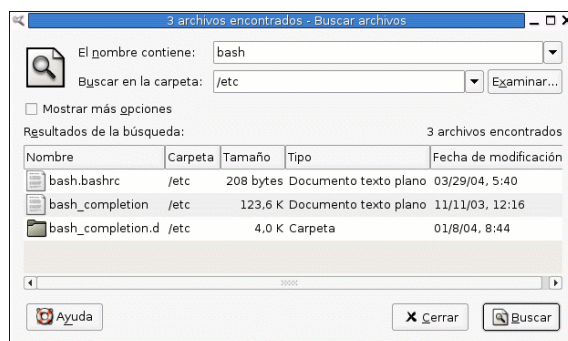
```
$ rm -r /home/Thales/curso/
```

y listo, ha borrado el directorio `curso` y todos los archivos, directorios y subdirectorios contenidos en él.⁹

➔ **Para practicar: Probar el uso de los comandos anteriores.**

2.3.3. Localización de archivos

Con `mc` o con `gnome-search-tool` (en modo gráfico: **Acciones**→**Buscar archivos**) la búsqueda de ficheros está “tirada”.

**locate**

```
locate patrón
```

Otra forma de buscar ficheros es usar el comando `locate`. Previamente debemos actualizar la base de datos de nombres de ficheros, para ello escribiremos:

```
# updatedb &
```

ejecutamos el comando como root y además en segundo plano, ya que tiene que localizar y almacenar todos los nombres de todos los ficheros y esto hace que su finalización no sea inmediata. Al ejecutar el

⁹Cuidado con esta orden. La “r” viene de Recursivo ¿os imagináis qué pasaría si como root escribís?:

```
# rm -r /
```

Incluso si en un directorio cualquiera ejecutamos como root la siguiente orden, pensando en eliminar los ficheros ocultos que empiezan por punto:

```
# rm -r .*
```

Recordaremos la película del aprendiz de brujo en la que las escobas no paraban de traer cubos de agua y desearemos pararlo como sea, aunque casi siempre demasiado tarde. Lo que ha ocurrido es que una de las expansiones de `.*` será el fichero `..` que es precisamente el directorio superior. ¡¡¡Socorro, ayuda!!!

comando en segundo plano, podremos seguir trabajando en nuestro terminal mientras que se ejecuta. Tan sólo tendremos que actualizar la base de datos de ficheros cada vez que se haya producido un cambio sustancial.

Cuando queramos buscar cualquier fichero que esté en la base de datos generada usaremos

```
$ locate patron
```

y en pantalla se nos mostrarán todas las concordancias con ese `patron`.

Por ejemplo

```
$ locate mpg
```

produce una salida como la que sigue (no tiene por qué coincidir)

```
/usr/lib/vlc/audio_filter/libmpgatofixed32_plugin.so
```

```
/usr/lib/vlc/demux/libmpga_plugin.so
```

```
/usr/lib/vlc/demux/libmpgv_plugin.so
```

```
/usr/lib/xmms/Input/libmpgl23.so
```

```
/usr/local/share/guadalinex/ejemplos/Videos/Spot25.mpg
```

```
.....
```

find

```
find [camino...] [expresión]
```

Con `find` podemos encontrar archivos basando su búsqueda en distintas características de los mismos. Su sintaxis básica es:

```
find [camino...] [expresión]
```

El número de opciones de `find` es muy elevado (`$man find`)

➔ Para practicar: `find` y `locate`

1. Uso de `find`

- Encuentra todos los archivos que hay en el directorio actual y en sus subdirectorios con extensión `.txt`

```
$ find . -name "*.txt"10
```

- Encuentra los ficheros con permisos `777`

```
$ find * -perm 77711
```

2. Localizar con `locate` los archivos de nombre `internet/Internet`, para eso hemos de añadir el parámetro `-i`

```
$locate -i internet
```

which

`which` comando

Si queremos conocer el path completo de un determinado comando, como por ejemplo de `startx`, usaremos:

```
$ which startx
```

la respuesta sería:

```
/usr/bin/X11/startx
```

¹⁰Las dobles comillas son necesarias para que no interprete el asterisco.

¹¹En estos dos casos, el asterisco significa el camino y sí queremos que se expanda, por eso no va entre comillas.

whereis

whereis comando

si no nos basta con esta información y, además, queremos saber qué páginas del manual acompañan al programa escribiremos:

```
$ whereis startx
```

el resultado es:

```
startx: /usr/X11R6/bin/startx      /usr/bin/X11/startx
```

➡ Para practicar

1. Buscar el path del comando `ls` y las *páginas de manual* de este comando.
2. Como un usuario normal hacer lo mismo con el comando `fdisk` ¿por qué `which` no lo encuentra?
3. Usando el comando `su` pasar a ser root y hacer de nuevo el apartado anterior.
4. Si no sabes muy qué puede estar pasando ejecuta:

```
echo $PATH
```

primero como usuario y después como root. ¿Qué diferencias observas?


2.3.4. Procesamiento de archivos

Con respecto a las órdenes para procesar archivos lo idóneo es saber que están ahí y las posibilidades que tenemos con ellas. Algunas son bastante especializadas y en general su uso puede ser escaso. Os remitimos a las manpages para profundizar en ellas en el caso de necesitarlas.

Las que más se utilizan son `cat` y `less` y ya han aparecido en repetidas ocasiones en su forma más estándar. Con respecto a `dd` comentar que nos permite (entre otras cosas) crear discos de inicio de Linux sin tener que usar `rawrite` ni MSDOS y ya lo hemos utilizado en el “para practicar” de la página 9 de la tercera entrega.

2.3.5. Empaquetando y comprimiendo ficheros.

Linux dispone de múltiples utilidades y programas para comprimir y descomprimir ficheros. Recorde-

mos que en modo gráfico disponemos de File Roller ( **Aplicaciones→Accesorios→File Roller**) que ya lo vimos en la segunda entrega, así que ahora nos detendremos en los más utilizados en modo comando, `tar` para empaquetar y `gzip` para comprimir.

tar

`tar` permite empaquetar o desempaquetar ficheros. El concepto de empaquetar aquí es el de meter varios ficheros y/o directorios en un solo fichero. Posteriormente podremos recuperar esa estructura de ficheros y directorios en el lugar donde queramos. Su sintaxis básica es¹²

```
$ tar opciones archivo.tar [origen]
```

donde:

opciones vamos a analizar:

c para crear archivos empaquetados

x para expandir archivos empaquetados

¹²Para ver todas la opciones de la orden `tar`:

```
$ man tar
```

- t** para mostrar el contenido de un fichero tar empaquetado
- v** almacenamos/visualizamos la información en forma detallada
- f** para indicar que `archivo.tar` es un fichero.
- z** filtrar el archivo a través de `gzip` (tanto para comprimir como descomprimir).
- M** para crear/desempaquetar usando varios discos.

archivo nombre del archivo tar

origen nombre¹³ del directorio/fichero o directorios/ficheros a empaquetar separados por espacios

Supongamos que deseamos empaquetar dos ficheros llamados `ed03linux1.txt` y `ed03linux2.txt`, en un fichero tar de nombre `ed03linux.tar`, escribiremos:

```
$ tar -cvf ed03linux.tar ed03linux1.txt ed03linux2.txt
```

también podíamos haber escrito:

```
$ tar -cvf ed03linux.tar ed03linux?.txt
```

Si lo que queremos es empaquetar el directorio `entrega_3`, en el fichero `entrega_3.tar`, la sintaxis sería:

```
$ tar -cvf entrega_3.tar entrega_3
```

Si lo que deseamos es desempaquetar un fichero tar, en vez de escribir la opción `-c` escribiremos `-x`, así para desempaquetar el contenido de la entrega anterior escribiremos:

```
$ tar -xvf entrega_3.tar
```

Si solamente queremos ver el contenido del fichero empaquetado (tar), ejecutaremos.

```
$ tar -tvf entrega_3.tar
```

gzip

Gzip permite comprimir ficheros, la sintaxis básica es:

```
$ gzip [opciones] archivo
```

dónde:

opciones vamos a analizar:

- d** para descomprimir archivos
- t** para comprobar que la compresión se ha realizado con éxito
- 1-9** nivel de compresión, el 1 indica menor ratio y mayor rapidez, el 9 daría como resultado un archivo más pequeño pero un mayor tiempo de compresión. El nivel por defecto es 6

archivo nombre del archivo a comprimir

Ya tenemos empaquetado nuestro archivo `entrega_3` y deseamos comprimirlo al máximo y verificar que todo está bien, usaremos:

```
$ gzip -9 entrega_3.tar
```

el resultado es el fichero `entrega_3.tar.gz`, si queremos comprobar la “integridad” del fichero escribiremos:

```
$ gzip -tv entrega_3.tar.gz14
```

```
entrega_3.tar.gz: OK
```

Si ahora queremos descomprimir este fichero tenemos dos opciones:

```
$ gzip -d entrega_3.tar.gz
```

o bien escribimos directamente

```
$ gunzip entrega_3.tar.gz
```

¹³Podemos usar comodines

¹⁴La `v` de “verbose”, es decir, para que me muestre más información en el terminal de cómo ha ido el proceso de testeo.

Los dos a la vez: tar y gzip Cabe la posibilidad de empaquetar y comprimir un directorio directamente sin tener que usar un comando tras otro, una posibilidad consiste¹⁵ en escribir:

```
$ tar -czvf nombre_fichero.tar.gz origen
```

o bien

```
$ tar -czvf nombre_fichero.tgz origen
```

en ambos casos, la opción `-z` es la que señala que vamos a comprimir. La extensión `tgz` es equivalente a `tar.gz`.

También podemos descomprimir y desempaquetar un fichero `tar.gz` o un `tgz` usando sólo la orden `tar`, la sintaxis sería:

```
$ tar -xzf fichero.tar.gz
```

o bien

```
$ tar -xzf fichero.tgz
```

➔ Para practicar: “targz”

El objetivo de esta práctica reside en aprender a empaquetar/comprimir directorios. Vamos a trabajar con el subdirectorio `/usr/share/doc/mozilla-browser`.



Notar que se supone que estamos trabajando como un usuario normal y no como el root. Un error en la sintaxis del último comando puede ser desastrosa para el sistema.

1. Empaquetar y comprimir

```
$tar -cvf ~/mozilla.tar /usr/share/doc/mozilla-browser
```

```
$ls -l mozilla.tar
```

(es grande)

```
$gzip mozilla.tar
```

```
$ls -l mozi*
```

(ya es de menor tamaño)

2. Ahora de un tirón

```
$tar -czvf ~/mozilla.tgz /usr/share/doc/mozilla-browser
```

```
$ls -l mozi*
```

¿Salen del mismo tamaño?

3. Descomprimirlo de una pasada:

```
$tar -xzf mozilla.tgz
```

Comprobar que todo ha salido bien

4. Ahora con formato zip:

```
$zip -r ~/mozilla.zip /usr/share/doc/mozilla-browser
```

¿Cuál comprime mejor?

5. Por último, borremos el directorio creado:

```
$rm -r ~/usr
```

¹⁵Sin usar tuberías

bzip2, bunzip2

Para comprimir y descomprimir ficheros existen más herramientas que las ya comentadas pero la única que merece mención especial es bzip2. La extensión de este tipo de ficheros es .bz2¹⁶, para comprimir un fichero escribiremos¹⁷

```
$ bzip2 fichero
y para descomprimir un fichero:
$ bunzip2 fichero.bz2
```

➔ Para practicar

Comprimamos otra vez el subdirectorio /usr/share/doc/mozilla-browser, pero ahora con:

```
$tar -cjvf ~/mozilla.tar.bz2 /usr/share/doc/mozilla-browser18
```

¿Cuál es el más comprimido?

Para descomprimirlo desempaquetarlo:

```
$bunzip2 mozilla.tar.bz2
```

```
$tar -xf mozilla.tar
```

¿Cómo se haría con un sólo comando?

Borremos ahora toda la “basura” generada:

```
$ rm mozi*
$ rm -r ~/usr
```

2.3.6. Control de tareas

Se entiende como *proceso* a cualquier programa en ejecución.

ps

El comando ps lista los procesos en ejecución en ese momento. Por ejemplo:

```
$ ps
```

muestra los procesos en ejecución del usuario; una posible salida es:

```
PID  TTY    TIME    CMD
3846 pts/1 00:00:00 bash
3899 pts/1 00:00:00 ps
```

el PID es el número de indentificador del proceso para la shell y CMD el nombre del proceso.

La sintaxis básica de este comando es:

```
$ ps [opciones]
```

donde las opciones más usuales son:

l Formato grande

u Da el nombre de usuario, la hora de comienzo y el uso de los procesos de este usuario de la máquina.

a Muestra también procesos de otros usuarios.

x Muestra los procesos sin terminal de control.

r Muestra sólo procesos que estén activos.

txx Muestra los procesos controlados por el terminal txx

¹⁶Es un formato con mejores ratios de compresión que los que ofrece gzip; bzip2 comprime ficheros utilizando el algoritmo de compresión de texto por ordenación de bloques de Burrows-Wheeler.

¹⁷Para ampliar es mejor consultar la manpage que acompaña al programa

¹⁸es equivalente a usar:

```
$tar -cvf ~/mozilla.tar /usr/share/doc/mozilla-browser | bzip2
```

Existen otros procesos en ejecución que no han sido listado por el comando anterior, si queremos verlos todos:

```
$ ps -aux | less
```

Primer y segundo plano

Casi todas las shell ofrecen la posibilidad de controlar la ejecución de los procesos y desde esta perspectiva, a los procesos se les conoce también con el nombre de *tareas*.

Generalmente cuando lanzamos un proceso lo hacemos en **primer plano**, introducimos el comando pulsamos enter y cuando el proceso ha terminado deja libre la shell para introducir nuevos comandos. A veces algunos procesos necesitan algún tiempo para terminar y no hacen nada interesante mientras tanto; en este caso lo mejor es lanzarlo en **segundo plano**. Para ello ejecutamos¹⁹

```
$ comando &
```

Conviene clarificar también la diferencia entre **interrumpir** y **suspender** un programa. Cuando interrumpimos un proceso (generalmente con **[Control]+[c]**) este muere, deja de estar en memoria, mientras que si lo suspendemos (generalmente con **[Control]+[z]**), el proceso se para temporalmente y podremos decir al sistema que continúe con la tarea más tarde.

fg, bg, jobs, kill

Supongamos que queremos actualizar nuestra base de datos de nombres de ficheros para el comando `locate`, para ello ejecutamos:

```
# updatedb
```

Observamos que tarda en terminar y suspendemos su ejecución con **[Control]+[z]**. El sistema nos devuelve el mensaje:

```
[1]+          Stopped          updatedb
```

que es autoexplicativo, el [1] es el número de tarea, el signo + señala la última suspendida.

Si ahora queremos que continúe pero en segundo plano, basta con ejecutar²⁰

```
# bg
```

para haber continuado en primer plano,

```
# fg
```

Si el proceso fué lanzado en segundo plano,

```
# updatedb &
```

el sistema devuelve un mensaje como este

```
[1]          1095
```

la tarea 1 con número de proceso (PID) 1095.

Si intentamos detener el proceso con **[Control]+[z]**, el sistema ni se entera (el proceso está corriendo en segundo plano), así que previamente debemos traerlo a primer plano con `$ fg` y después ya podemos suspenderlo, relanzarlo o matarlo.

El comando `jobs` (interno de la shell) informa sobre el estado de los procesos (`ps` también).

Con `kill` podemos matar un proceso, la sintaxis más usual es:

¹⁹ya lo hemos hecho en reiteradas ocasiones a lo largo del curso

²⁰Si hubieran mas tareas suspendidas,

bg %numero_tarea.

fg del inglés *foreground* y **bg** de *background*.

```
kill [señal21] PID...22
```

donde `señal` es opcional y en general toma dos valores

-15 (SIGTERM) es la señal por defecto y no siempre es capaz de “matar” todos los procesos

-9 (SIGKILL) es el “Rambo” de las señales, acaba con cualquier proceso.

Si no especificamos ninguna señal, estamos mandando la señal 15 y de una manera no del todo formal, le estamos diciendo al proceso que se muera por las buenas. Es deseable que sea así, porque si hace caso el proceso, puede cerrar los ficheros, descargar los datos de memoria a disco y decirle a sus hijos (en caso de que los tuviera) que también se mueran por las buenas.

Si nos vemos obligados a utilizar la señal 9, lo matamos bien muerto, sin tiempo a que cierre ficheros ni descargue datos de memoria a disco. Moraleja, intentaremos mandarle primero un `kill` normal y si no hay manera pasaremos a la artillería pesada.

Supongamos que hemos cerrado mozilla y que notamos que el sistema “está lento”, escribimos:

```
$ps -ax
```

y ¡date!, mozilla sigue en ejecución con el PID

```
...
```

```
3940 tty1 S 0:01 /usr/lib/mozilla-1.0.1/mozilla-bin
```

```
...
```

ejecutamos entonces:

```
$ kill -9 3940
```

y listo, se acabó mozilla (en sentido figurado, claro está). Si ahora ejecutamos de nuevo:

```
$ ps -ax
```

no debería aparecer mozilla por ningún lado.

Si la lista de procesos es muy larga también podemos filtrar la salida con `grep`:

```
$ ps -ax | grep mozilla
```

➡ Para practicar

1. Ejecutar la secuencia de comandos

```
# updatedb &
# tar -czf home.tgz /home
[Control]+[z]
# jobs
# ps
```

2. Uso conjunto de `ps` y `kill`

- a) En modo gráfico abrir una xterm y ejecutar en segundo plano el programa `gedit`

```
$ gedit &
```

Comprobar la ID del proceso anterior con

```
$ps -a
```

y la diferencia de usar:

```
$ps -ax
```

```
$ps -aux
```

Matar el programa con:

```
$kill -15 PID_gedit
```

²¹Con

```
$ kill -l
```

podemos visualizar todas las señales posibles (es una `ele` minúscula).

²²En este caso no hemos puesto ni `$` ni `#` ya que el root podrá “matar” procesos de todos los usuarios, pero un usuario tan sólo podrá “matar” los suyos.

3. Acceder al sistema como root y como otro usuario del sistema (llamémosle INVITADO). Después de ver qué tty se corresponde con la sesión abierta de INVITADO con

```
#ps -U INVITADO
```

Pero el INVITADO es un poco gorrón y deseamos mandarlo a tomar viento, para eso matamos esa sesión con el PID del proceso login

```
#kill -9 PID_login
```

Comprobar que ya no hay “invitados”

at



Para poder comprobar lo que se expone sobre el comando at y que funcione el envío del correo hemos de modificar la configuración de exim²³. El cambio consiste en comentar la línea

```
#qualify_domain = andaluciajunta.es
```

del fichero de configuración²⁴ de exim: /etc/exim/exim.conf.

El comando at posibilita planificar la ejecución de tareas; permite que le especifiquemos tanto la fecha como la hora para activarse. Una vez activo, at se encargará de hacer ejecutar las órdenes programadas (órdenes no interactivas). Su sintaxis es:

```
at hora [fecha] lista_comandos
```

Por ejemplo supongamos que son las 3 h pm y hemos quedado a las 4 h pm, somos tan despistados que cuando nos ponemos con el ordenador se nos olvida todo, en ese caso podemos decirle a at que nos avise dentro de una hora escribiendo:

```
$ at now +60 minutes25
```

tras pulsar intro podremos escribir aquello que consideremos oportuno, por ejemplo:

```
at>echo "No te despistes, tienes una cita"
```

cuando terminemos de introducir los comandos deseados pulsaremos [Ctrl]+[d].

A las cuatro at nos enviará un correo con el texto anterior que podremos visualizar con la orden mail.

at permite distintas formas para especificar la fecha y hora en que debe activarse. Así, el tiempo se puede especificar en la forma HHMM o HH:MM para llevar a cabo una tarea en el mismo día. Por ejemplo la orden anterior es equivalente a:

```
$ at 16:00
```

Con at es posible usar midnight (medianoche), noon (mediodía), teatime (4 de la tarde) o tomorrow (mañana). También podemos anteponer a la hora am o pm.

Si queremos que at se ejecute en un día distinto al que estamos, pondremos la fecha en la forma ‘mes día’ por ejemplo, May 12.

Asociado al comando at tenemos los comandos:

atq muestra un listado de los trabajos en espera de ejecución.

atrm para eliminar trabajos en espera.

²³Se trata del agente de transporte de correo (MTA) de Guadalinex: “el cartero”

²⁴Para configurar exim se puede ejecutar el comando eximconfig. Os remitimos a la ayuda instalada del programa para conocer más sobre él.

²⁵También podemos dar el tiempo de espera como un incremento de un número de weeks (semanas), days (días) u hours (horas).

➡ Para practicar: at

Usando el comando `at` programar un trabajo para dentro de 2 minutos (o un tiempo razonable) que:

1. Te mande un correo con el texto que te parezca, una idea: “Curso Linux te saluda”
2. Comprobar/leer el correo con (este comando se ve después en la página 38, pero la práctica mejor ahora):

```
$mail
&l
Para salir
&q
```

cron (anacron)

Se trata de “la herramienta” que usa Linux para planificar tareas. Para conocerla mejor:

```
$man cron
```

Veamos el contenido de un fichero de configuración de `cron` básico:

```
$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file.
# This file also has a username field, that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
25 6 * * * root test -e /usr/sbin/anacron || run-parts --report /etc/cron.daily
47 6 * * 7 root test -e /usr/sbin/anacron || run-parts --report /etc/cron.weekly
52 6 1 * * root test -e /usr/sbin/anacron || run-parts --report /etc/cron.monthly
#
```

Expliquemos un poco este fichero, las líneas tienen la forma:

fecha nombre_usuario comando

- Como siempre, las líneas que comienzan por `#` son comentarios.
- Con las 2 primeras líneas asignamos las variables de entorno con que se va a ejecutar `cron`.
- ¿Qué hacen las 3 últimas?
 - Con ellas se controlan las acciones a realizar cada día (`/etc/cron.daily`), semana (`/etc/cron-weekly`) y mes (`/etc/cron.monthly`)²⁶. Así, por ejemplo:
`/etc/cron.daily/find` ejecuta cada día `updatedb` con ciertos parámetros
 - Para facilitar la coordinación con `anacron` la llamada de `run-parts` para los directorios `/etc/cron.daily`, `/etc/cron.weekly` y `/etc/cron.monthly` es

```
test -e /usr/sbin/anacron || run-parts --report /etc/cron.daily
```

De esta forma, si `anacron`²⁷ se instala será el responsable de la ejecución de esos scripts.

²⁶Para controlar las tareas cada hora `/etc/cron.hourly`

²⁷`anacron` es un programador de tareas similar a `cron`, pero a diferencia de este último no requiere que el sistema esté en ejecución permanentemente.

- Con `run-parts` indicamos que se ejecuten los scripts pasados como parámetro en las fechas seleccionadas al inicio de la línea. El formato en que se codifican las fechas es:

CAMPOS	minuto	hora	día	mes	día de la semana
rango	0-59	0-23	1-31	1-12	0-7

- El 0 y el 7 del día de la semana corresponde al domingo (se puede usar *sun*, *sat*, ...).
- 3-6 equivale a la lista de números 3, 4, 5, 6
- El asterisco significa cualquier valor del rango permitido
- Podemos incrementar un valor con el formato `/numero`. Por ejemplo, podemos conseguir que un comando se ejecute cada 8 horas escribiendo `*/8` en el campo hora.

shutdown

```
shutdown [opciones] tiempo [mensaje]
```

Es un comando para cerrar el sistema. A continuación exponemos su uso más corriente y sus equivalencias.

```
# shutdown -h now
o bien
# halt
```

y que para reiniciar el sistema la orden es:

```
# shutdown -r now
o bien
# reboot
```

Los parámetros que se han pasado son claros, `h` para “halt” y `r` para “reboot”, y la opción “now” por ahora mismo. Comentar que podemos pasarle como argumento el tiempo antes de cerrar/reiniciar el sistema, así en vez de *now* podemos escribir:

```
# shutdown -h +5
```

con lo que el sistema se cerrará dentro de cinco minutos.

uname

```
uname [opciones]
```

Por último, con

```
$ uname -a
```

conseguimos toda la información del sistema, una posible salida es:

```
(Sistema Hosts Versión del núcleo y fecha de la compilación)
```

```
Linux guadalinux 2.4.23-ck1 #5 vie ene 16 11:17:17 CET 2004 i686 GNU/Linux
```

Si no ponemos ninguna opción sólo se nos muestra el nombre del sistema operativo.

2.3.7. Administrar ficheros

Algunos de los comandos que disponemos para administrar sistemas de ficheros como `mkfs`, `fsck`, `mount` o `umount`, ya se vieron en la tercera entrega, así que ahora sólo mencionaremos algunos más:

df

```
df [opciones] [sistema_archivos]
```

```
$ df
```

cuya salida podría ser (no tiene por qué coincidir):

S.ficheros	Bloques de 1k	Usado	Dispon	Uso %	Montado en
/dev/hdc1	2392739	1523644	770136	66 %	/
/dev/hdc5	1585200	765078	738193	51 %	/datos

es autoexplicativa. Si deseamos tener una salida más “comprensible” podemos escribir:

```
$ df -h
```

en cuyo caso la información se nos mostrará como sigue:

S.ficheros	Bloques de 1k	Usado	Dispon	Uso %	Montado en
/dev/hdc1	2.3G	1.5G	752M	66 %	/
/dev/hdc5	1.5G	747M	721M	51 %	/datos

du

```
du [opciones] [nombre_archivo...]
```

```
$ du
```

Lista el espacio ocupado por los archivos o directorios que cuelgan desde donde se invoca. Si se ejecuta sin argumentos es poco práctico. Uno de los modos más corrientes de uso es:

```
$ du -sh directorio
```

da el total de espacio ocupado por ese directorio en formato *humano*, es decir, añade una letra indicativa del tamaño, como M para megabytes:


```
104M directorio
```

fdformat

```
fdformat device
```

Cuando queramos formatear un disco flexible escribiremos (sin tener el disco montado):

```
$ fdformat /dev/fd028
```

Recordar que existe una utilidad gráfica para formatear disquetes a la que se accede con  **Aplicaciones→Accesorios→Formato de disquetes**, a la que puede accederse desde un terminal gráfico ejecutando el comando

```
$ gfloppy
```

fdisk

Desde Linux podemos ejecutar el `fdisk` de Linux para visualizar o modificar las particiones del disco duro. La sintaxis es:

```
# fdisk device
```

es en modo texto, un poco más árido que el `fdisk` del DOS y menos intuitivo que `cfdisk` y por supuesto que `QtParted` que ya utilizamos en la instalación de Guadalinex.

²⁸En el supuesto de que nuestro disco flexible sea `/dev/fd0`. Para ver los parámetros de `fdformat`, ejecutar:
`man fdformat`

➡ Para practicar:

Comprobar el espacio ocupado/disponible:

```
$df -h
```

Espacio que ocupa nuestro directorio de trabajo:

```
$du -sh ~/
```

Particiones del disco:

```
#fdisk -l /dev/hda
```

2.3.8. Comunicaciones y redes.

Aunque el curso no incluya ningún tema sobre comunicaciones y redes, mencionaremos algunos comandos propios de este apartado.

finger

Proporciona información sobre los usuarios conectados al sistema. Su sintaxis es:

```
finger [-lmsp] [usuario...] [usuario@host...]
```

por ejemplo:

```
$ finger
```

muestra entre otros datos: el directorio de conexión, el nombre completo, la fecha de conexión, etc. Si queremos que la información sea más detallada escribiremos:

```
$ finger -l
```

Si sólo queremos información del usuario THALES escribiremos:

```
$ finger Thales
```

who

```
who [opciones]
```

Con who podemos ver los usuarios conectados a nuestro sistema. Además, nos muestra el terminal en el que están conectados y la hora de conexión. who imprime la siguiente información por cada usuario que actualmente está conectado al sistema:

- nombre de la cuenta (login name)
- terminal
- fecha y hora de conexión
- nombre de ordenador remoto o terminal X

Si sólo queremos información sobre el usuario que ejecuta la orden escribiremos

```
$ who -m
```

w

```
w [usuario]
```

w nos da información sobre los usuarios que están conectados en ese momento y sobre sus procesos.

```
$ w
```

```
00:13:00 up      6:59,  4 users,   load average: 0,51,  0,54,  0,31
USER  TTY  FROM  LOGIN@  IDLE   JCPU   PCPU   WHAT
chico  tty1  -      23:54   17:59   1.75s   0.01s   /bin/sh /usr/bin/
juan   :0    -      17:14   17:14   ?xdm?  21:37   0.32s   /usr/bin/gnome-session
juan   pts/0  :0.0   17:20   0.00s   0.20s   0.00s   w
chico  pts/1  :1.0   23:55   17:41   0.14s   0.14s   bash
```

En la primera línea nos informa de

- la hora actual
- cuánto lleva el sistema funcionando
- cuántos usuarios están conectados y
- las cargas medias en los anteriores 1, 5 y 15 minutos.

A continuación para cada usuario se muestra:

- nombre de login,
- nombre de tty,
- nodo remoto,
- tiempo de conexión,
- tiempo inactivo,
- JCPU²⁹,
- PCPU³⁰ y
- la línea de comando del proceso en curso.

write

```
write usuario [terminal]
```

El comando `write` nos permite mandar un mensaje a otro usuario conectado al sistema. Previamente ese usuario debe tener `mesg`³¹ en “y”, en el caso de que ese usuario tenga `mesg` en “n” el sistema nos avisará con el mensaje de error:

```
write: "usuario" has messages disabled
```

Supongamos que deseamos enviar un mensaje al usuario THALES, y que tiene activa la opción de que le envíen mensajes, en ese caso escribiríamos

```
$ write Thales
```

se nos avisará de que estamos en modo de edición, una vez escrito el texto que deseemos enviar como mensaje:

```
Hola Thales, que no se te olvide la cita.
```

```
pulsaremos [Ctrl]+[d] y el mensaje será enviado.
```

²⁹JCPU es el tiempo usado por todos los procesos bajo el terminal tty.

³⁰PCPU es el tiempo usado por el proceso en curso.

³¹el comando `mesg` permite activar o desactivar la posibilidad de recibir mensajes de otros usuarios, funciona como un “interruptor” con sólo dos estados y o n. Estos estados se le tienen que pasar como parámetros.

wall

```
wall [archivo]
```

Si queremos enviar un mensaje no a un solo usuario, sino a todos los usuarios conectados, usaremos `wall`. Si lo que queremos es mandar un fichero escribiremos:

```
$ wall <fichero.txt
```

y el contenido de este fichero será enviado al terminal de todos los usuarios conectados al sistema.

mail

```
mail [usuario]
```

En el caso de que nuestro usuario de destino **no** esté conectado el mejor comando para comunicarnos con él es `mail`. Si queremos enviar un mensaje a `THALES` escribiremos³²:

```
$ mail Thales33
```

```
Subject: Cita
```

```
Te recuerdo que tenemos una cita
```

tras escribir el texto del mensaje pulsamos `[Ctrl]+[d]` (o insertamos un “punto” al inicio de una nueva línea) y se nos mostrará la opción

```
Cc:
```

por si queremos mandar una copia del mensaje a otro usuario.

Cuando `THALES` se conecte al sistema, éste le avisará de que tiene un correo (si se conecta en modo texto):

```
You have new mail.
```

Tanto en modo texto como gráfico, ejecutando `mail` podrá visualizarlo, borrarlo, etc:

```
$ mail
```

```
Mail version 8.1.2 01/15/2001. Type ? for help.
```

```
"/var/mail/Thales": 1 message 1 new
```

```
>N 1 paco@andaluciajun Tue Apr 13 15:24 13/370 Cita
```

```
&
```

Por ejemplo, para visualizar el correo tan sólo tiene que pulsar sobre el número que hay antes del mensaje:

```
& 1
```

para borrarlo ejecutar

```
& d 1
```

para más ayuda pulsar ?

```
& ?
```

para salir

```
& q
```

Si queremos enviar un mensaje a un usuario de otra máquina escribiremos:

```
$ mail usuario@nombre_maquina
```

Para ampliar sobre este comando lo mejor es mirar en la ayuda.

³²Para que funcione tal cual se expone en los apuntes véase el **Stop** de la página 32

³³Esta es la sintaxis en cualquier distribución. En Guadalinux debemos escribir `mail usuario_destino@localhost` o bien `usuario_destino@nombre_maquina`, por ejemplo `mail Thales@guadalinux` (si así se llama nuestra máquina).

Debido a la configuración de `mail` en Guadalinux, la dirección del remitente aparecerá como `usuario_remitente@andaluciajunta.es`

➡ **Para practicar.**

Comprobar los ejemplos de los comandos de esta sección .

Capítulo 3

Programa Midnight Commander

Este administrador de ficheros para Linux constituye una herramienta única para facilitarnos y, sobre todo, para acelerar todas las operaciones que diariamente debemos realizar con nuestro ordenador. Posiblemente se encuentre entre los programas más útiles que podemos encontrar para cualquier sistema operativo y más concretamente para Linux. (*Midnight Commander*. Sólo Programadores Linux n 7. DAVID ESPADA GARCÍA)

3.1. Introducción

Si trabajamos en modo texto o desde una `xterm`, tenemos a nuestra disposición un programa (tipo Comandante Norton) que es de un valor inestimable para movernos por el sistema de ficheros de un sistema Linux, se trata de:

mc¹ Entorno visual para sistemas tipo Unix. *Midnight Commander* es un navegador de directorios/gestor de ficheros para sistemas operativos tipo Unix.



Desarrollaremos bastantes aspectos del programa que no hay que conocer en su totalidad. Para trabajar este capítulo os recomendamos una primera lectura rápida que os dé una idea de las posibilidades que tiene este programa y, después, cada uno adecuará el nivel de profundidad o temas de interés a sus necesidades.

3.1.1. Inicio de una sesión

Lo ejecutaremos con la orden:

```
$ mc
```

La pantalla de *Midnight Commander* está dividida en cuatro partes. La mayor parte del espacio de la pantalla se usa para los dos paneles de directorio. Por defecto, la segunda línea más inferior de la pantalla es la línea de comandos del shell, y la línea inferior muestra las etiquetas de las teclas de función.

¹ Este documento se basa en la página `man` del programa `mc`, lo único que se ha hecho es “resumir” y actualizar algunos aspectos de dicha página y añadir gráficos explicativos del programa. Para una mayor información sobre el programa os remitimos a dicha página de ayuda.

Izquierdo			Derecho		
Archivo			Archivo		
Nombre	Tamaño	FechaMod	Nombre	Tamaño	FechaMod
/auto	4096	17 sep 2003	/.Trash	4096	21 feb 14:27
/bin	4096	15 ene 10:14	/.fluxbox	4096	20 feb 12:57
/boot	4096	16 ene 12:59	/.gconf	4096	22 feb 13:16
/cdrom	4096	17 sep 2003	/.gconfd	4096	22 feb 13:20
/dev	24576	17 abr 11:09	/.gimp-1.3	4096	20 feb 12:57
/etc	8192	17 abr 11:09	/.gnome	4096	20 feb 13:05
/floppy	4096	16 sep 2003	/.gnome2	4096	22 feb 13:20
/grabadora	4096	19 feb 13:56	/.gnome2_private	4096	20 feb 13:05
/home	4096	20 feb 13:04	/.mc	4096	21 feb 13:42
/initrd	4096	16 sep 2003	/.metacity	4096	20 feb 13:05
/lib	4096	8 ene 20:37	/.mozilla	4096	20 feb 18:27
/lost+found	16384	30 dic 16:52	/.nautilus	4096	20 feb 18:21
/mnt	4096	19 feb 13:54	/.smb	4096	21 feb 14:26
/opt	4096	16 sep 2003	/.sodipodi	4096	20 feb 19:44
/proc	0	17 abr 11:09	/.thumbnails	4096	20 feb 18:56
/root	4096	12 mar 18:42	/.xcdroast	4096	20 feb 19:12
/sbin	4096	15 ene 10:14	/Desktop	4096	21 feb 14:27
/tmp	4096	17 abr 11:32	~Documents	14	20 feb 12:57
/usr	4096	8 ene 23:33	~Mis documentos	4096	20 feb 12:57
/var	4096	8 ene 20:52	...		
/home					

Ayudita: M-p y M-n permiten acceder a la historia de órdenes.
 \$ **1 Ayuda 2 Menú 3 Ver 4 Editar 5 Copiar 6 RenMov 7 Mkdir 8 Borrar 9 Menú 10 Salir**

La línea superior es la línea de la barra de menú. Podemos activar la barra de menú si pulsamos en la primera línea de la pantalla con el ratón o pulsamos la tecla **F9**².

El *Midnight Commander* provee una vista de dos directorios al mismo tiempo. Uno de los paneles es el panel actual (una barra de selección está en el panel actual). La mayoría de las operaciones tienen lugar en el panel actual. Algunas operaciones con ficheros como renombrar y copiar por defecto utilizan el directorio del panel de-seleccionado como destino.

Podemos ejecutar comandos del sistema desde el *Midnight Commander* simplemente escribiéndolos. Todo lo que escribamos aparecerá en la línea de comandos del shell y cuando pulsemos **Enter**, *Midnight Commander* ejecutará la línea de comandos.

➔ Para practicar

Vamos a copiar el fichero `sources.list` en nuestra carpeta de usuario escribiendo directamente, para esto es necesario conocer la ruta exacta de dónde se encuentra e indicarle exactamente dónde deseamos copiarlo:

```
# cp /etc/apt/sources.list /home/usuario3
```

Pulsando con el ratón sobre las teclas de función o usando las teclas **F1-F10**⁴ tenemos posibilidad de ejecutar las operaciones más comunes.

3.1.2. Soporte de Ratón

Midnight Commander viene con soporte de ratón. Una doble pulsación sobre un fichero intentará ejecutar el comando si se trata de un programa ejecutable; y si la extensión del fichero tiene un programa asociado para la extensión del fichero, se ejecutará dicho programa.

Además, es posible ejecutar los comandos asignados a las etiquetas de las teclas de función pulsando con el ratón sobre ellas.

Si se pulsa con el botón del ratón en la línea divisoria superior del panel de directorio, se realiza un scroll (desplazamiento) de una página hacia atrás. Asimismo, una pulsación sobre la línea divisoria inferior

²Lógicamente si estamos trabajando en un entorno gráfico, esta línea se puede activar directamente haciendo “clic” con el ratón.

³Fíjate que tenemos que trabajar como root, ya que de lo contrario no tendríamos los permisos adecuados para poder copiar ese fichero en concreto; recuerda que sólo el “jefe” puede hacerlo.

⁴Si ejecutamos `mc` desde un terminal de GNOME nos encontraremos con un problema a la hora de intentar usar la tecla **F10** [Salir], ya que se usa en los terminales gráficos para acceder a los menús. Si deseamos poder usarla desde `mc`, pulsaremos en **Edit** → **Combinaciones de teclas** del menú del terminal gráfico y marcaremos la casilla correspondiente a:

Deshabilitar la tecla de acceso a menús (F10 por omisión)

produce un scroll de una página hacia delante. Este método de la línea divisoria funciona también con el Visor de Ayuda y el Árbol de directorios.

Si estamos ejecutando *Midnight Commander* con el soporte de ratón, podemos obtener el comportamiento por defecto del ratón (cortando y pegando texto) manteniendo pulsada la tecla *Shift*.

3.1.3. Teclas

Algunos comandos en *Midnight Commander* implican el uso de las teclas Control (algunas veces representado por CTRL o CTL) y Meta (algunas veces denominado ALT o incluso Compose).



Algunas de las combinaciones de teclas que a continuación se van a describir no tendrán efecto cuando lo ejecutemos desde un entorno gráfico ya que prevalecen las combinaciones de teclas del entorno gráfico sobre éste; pero sí funcionarán correctamente desde un entorno de texto.

En este manual⁵ usaremos las siguientes abreviaturas:

C-<chr> significa mantener pulsada la tecla Control mientras pulsamos el carácter <chr>. Así C-f sería: manteniendo pulsada la tecla Control teclear f.

M-<chr> significa mantener pulsada la tecla Meta⁶ o [ALT Gr] mientras pulsamos el carácter <chr>. No todos los teclados tiene la tecla Meta ubicada en el mismo lugar y se puede dar el caso de que en algunos teclados no esté disponible dicha tecla. Si no funcionase con [ALT Gr], probaríamos con la tecla [ALT].

Para una descripción detallada de las combinaciones de teclas véase la página man del programa, aquí sólo vamos a comentar aquellas más “importantes” que impliquen acciones que no son realizables usando los menús del programa.

Merece la pena comentar que los cursores y las teclas Inicio, Fin, etc realizan las labores “usuales” a las que estamos habituados.

Comentemos algunas de ellas:

Enter. Si hay algún texto en la línea de comandos (la de la parte inferior de los paneles), entonces se ejecuta ese comando. Si no hay texto en la línea de comandos y la barra de selección está situada sobre un directorio, entonces *Midnight Commander* realiza un *chdir*⁷ al directorio seleccionado y recarga la información en el panel; si la selección es un fichero ejecutable entonces es ejecutado. Por último, si la extensión del fichero seleccionado coincide con una de las extensiones en el fichero de extensiones entonces se ejecuta el comando correspondiente.

Tab Cambia el panel actual. El panel activo deja de serlo y el no activo pasa a ser el nuevo panel activo. La barra de selección se mueve del antiguo panel al nuevo, desaparece de aquél y aparece en éste.

Insertar Para marcar ficheros (y/o directorios) como seleccionados podemos usar la tecla Insertar (Ins). Para deseleccionar ficheros, basta con repetir la operación sobre los ficheros y/o directorios antes marcados.

+ (más) Usado para seleccionar (marcar) un grupo de ficheros.

- (menos) Usaremos la tecla - para deseleccionar un grupo de ficheros.

Con respecto a las teclas útiles para evitar la excesiva escritura cuando se introducen comandos del shell, tenemos:

C-Enter. Copia el nombre de fichero seleccionado a la línea de comandos. Si estamos ejecutándolo desde el entorno gráfico la combinación de teclas sería [CTRL]+[ALT]+[Enter]

M-Tab. Realiza una Terminación automática (completion) del nombre de fichero, comando, variable, nombre de usuario y host.

C-x p, C-x C-p. La primera secuencia de teclas copia el nombre de la ruta de acceso actual a la línea de comandos, y la segunda copia la ruta del otro panel a la línea de comandos.

C-q. El comando cita (quote) puede ser utilizado para insertar caracteres que de otro modo serían interpretados por *Midnight Commander* (como el símbolo '+')

⁵Sólo en la parte correspondiente al programa mc.

⁶En nuestros teclados dispondremos de ALT Gr

⁷Del inglés “change directory” - cambiar directorio

M-p, M-n. Usaremos esas teclas para navegar a través del histórico de comandos. M-p devuelve la última entrada, M-n devuelve la siguiente.

3.2. Barra de Menú

La barra de menú aparece cuando pulsamos **[F9]** o pulsamos el botón del ratón sobre la primera fila de la pantalla. La barra de menú tiene cinco submenús: "Izquierdo", "Fichero", "Comando", "Opciones" y "Derecho".

Izquierdo	Archivo	Utilidades	Opciones	Derecho
-----------	---------	------------	----------	---------

Los Menús Izquierdo y Derecho nos permiten modificar la apariencia de los paneles de directorio izquierdo y derecho.

El Menú de Fichero lista las acciones que podemos realizar sobre el fichero actualmente seleccionado o sobre los ficheros marcados.

El Menú de Comandos lista las acciones más generales y que no guardan relación con la selección actual de ficheros.

Una vez activo un menú podemos usar la letra marcada con mayúsculas y amarillo para acceder a ese comando del menú.

3.2.1. Menús Izquierdo y Derecho

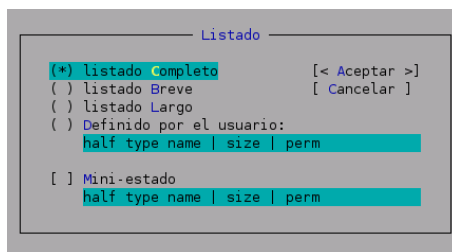
La presentación de los paneles de directorio puede cambiarse desde los menús Izquierdo y Derecho.

Izquierdo	Archivo	Utilidades	Opciones	Derecho
<div>Modo de listado... Vista rápida C-x q Información C-x i Árbol Ordenar... Filtro... conexión por FTP... conexión por shell... actualizaR C-r /.mozilla /.nautilus /.netscape /.netscape6 /.. /..</div>		<div><div>TamañoFechaMod</div><div>DIR-ANI</div><div>409611abr23:12/.Trash</div><div>409617abr17:51/.gconf</div><div>409617abr20:01/.gconfd</div><div>409617abr18:17/.gimp-1.2</div><div>40969mar00:44/.gnome</div><div>409617abr18:16/.gnome-desktop</div><div>409611abr23:13/.gnome2</div><div>40969mar00:44/.gnome2_private</div><div>409611mar18:44/.kde</div><div>40969mar00:50/.lyx</div><div>409617abr20:09/.mc</div><div>40969mar00:44/.metacity</div><div>409612mar00:17/.mozilla</div><div>40969mar00:44/.nautilus</div><div>409611mar18:42/.netscape</div><div>409611mar18:42/.netscape6</div><div>409611mar18:42/..</div></div>		

Ayudita: Puede "Llevar al panel" los resultados de "Buscar archivos" y usarlos.
[fermin@studio fermin]\$
1Ayuda 2Menú 3Ver 4Editar 5Copiar 6RenMov 7Mkdir 8Borrar 9Menú 10Salir

Modo de listado... El modo de listado⁸ se usa para visualizar ficheros y sus atributos. Hay cuatro modos diferentes: Completo, Breve, Largo y Personalizado

⁸Modo por defecto cuando entramos en el programa.

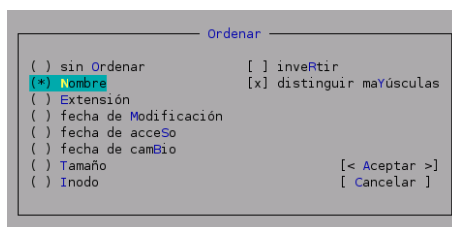


Vista Rápida En este modo, el panel cambia a un visor reducido que muestra el contenido del fichero actualmente seleccionado, si seleccionamos el panel (con la tecla tab o el ratón), tendremos acceso a los comandos usuales del visor.

Información La vista de información visualiza información relacionada con el fichero seleccionado actualmente y, si es posible, información sobre el sistema de ficheros actual.

arBol La vista Árbol muestra “casi” el árbol de directorios.

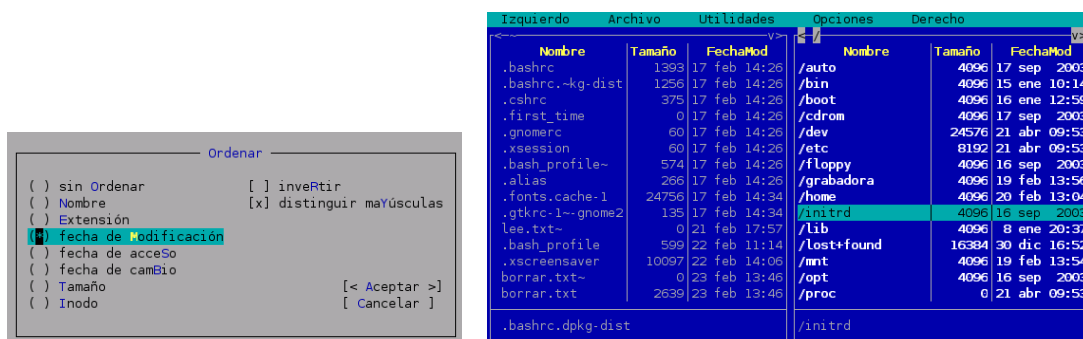
Ordenar ... Los ocho modos de ordenación son: sin ordenar, por nombre, por extensión, por fecha de modificación, por fecha de acceso, por la fecha de modificación de la información del i-nodo, por tamaño y por i-nodo. En el cuadro de diálogo del modo de ordenación podemos elegir el modo de ordenación así como especificar si deseamos que éste se realice en orden inverso chequeando la casilla Invertir.



Por defecto los directorios son ordenados antes que los ficheros pero esto puede ser cambiado desde el Menú de Opciones (opción Mezcla todos los ficheros).

➔ Para practicar

En muchas ocasiones nos encontramos con la necesidad de localizar algún archivo que hemos modificado recientemente, para esto lo mejor es que ordenemos nuestros ficheros por la fecha de modificación ¿verdad?



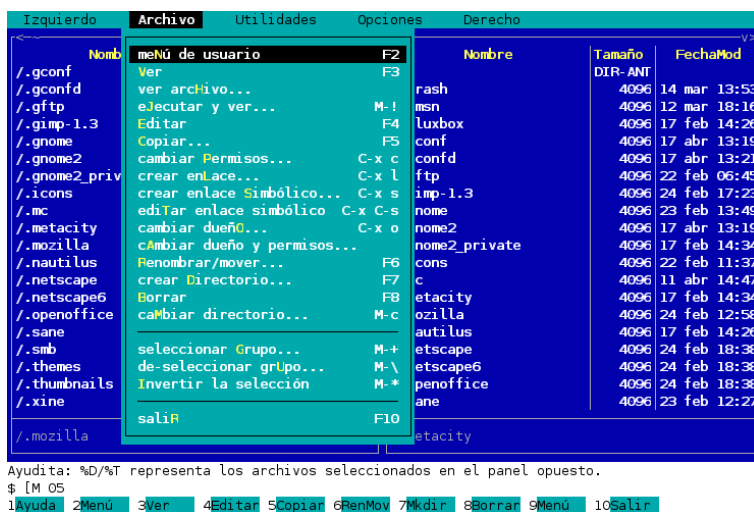
Observaremos que primero nos ordena, según la última modificación realizada, los directorios y a continuación el resto de ficheros. Nos indica la fecha exacta (dd:mes:aaaa) e incluso la hora de la última modificación realizada.

Filtro... El comando de filtro nos permite seleccionar un patrón (por ejemplo *.tar.gz) con el cual los ficheros deben coincidir para ser mostrados. Indiferente al patrón de filtro, los directorios y enlaces a directorios se muestran siempre en el panel de directorios.

actualizaR El comando actualizar recarga la lista de ficheros en el directorio.

3.2.2. Menú de Archivo

Midnight Commander utiliza las teclas de función F1 - F10 como atajos de teclado para los comandos que aparecen en el menú de Fichero.



El menú de Fichero posee los siguientes comandos (teclas rápidas de teclado entre paréntesis):

meNú de usuario(F2) Invoca el Menú de usuario. El menú de usuario otorga una manera fácil de tener usuarios con un menú y añadir asimismo características extras a *Midnight Commander*. Contiene acciones realizables con el fichero actualmente elegido y que pueden ser definidas de una forma sencilla.⁹

Ver (F3) Visualiza el fichero actualmente seleccionado. Por defecto invoca el Visor de Ficheros Interno pero si la opción "Usar visor interno" está desactivada, invoca un visor de ficheros externo especificado por la variable de entorno.

Visor de Ficheros Interno El visor de ficheros interno posee dos modos de pantalla: ASCII y hexadecimal. Para intercambiar entre modos, usaremos la tecla **F4**. Si tenemos el programa *gzip* instalado, se usará automáticamente para descomprimir los ficheros según se necesite.

El visor intentará usar el mejor método posible en nuestro sistema para mostrar la información. El visor interno de ficheros interpretará algunas secuencias de cadenas para activar los atributos de negrita y subrayado, para conseguir una apariencia mejor de nuestros ficheros.

En modo hexadecimal, la función de búsqueda acepta texto entre comillas así como constantes hexadecimales.

⁹Para profundizar sobre este tema se puede acudir a la ayuda del programa, donde se explica la función de cada una de las opciones posibles.

ver fichero ... Permite visualizar un fichero que le pasemos como argumento. Por defecto toma el fichero seleccionado.

Editar (F4) Invoca el editor vi, u otro especificado en la variable de entorno EDITOR, o en el Editor de Ficheros Interno si la opción use_internal_edit está activada.

Editor de Ficheros Interno



El editor interno proporciona la mayoría de funcionalidades de los editores comunes de pantalla completa. Es invocado pulsando **[F4]** indicado por la variable use_internal_edit en el fichero de inicialización. Tiene un tamaño límite de fichero extensible de dieciséis megabytes y edita los ficheros binarios de manera impecable.

El editor es muy fácil de utilizar y no requiere de aprendizaje alguno. Para activar el menú superior basta con pulsar **[F9]** o haciendo “clic” con el ratón sobre la barra superior.

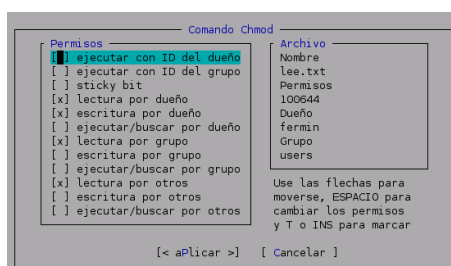
Para conocer la función de las teclas, basta consultar el menú emergente apropiado. Otras teclas son:

- Mayúsculas+cursores producen el resaltado de texto.
- Ctrl-Insert copia al fichero cooedit.clip.
- Mayúsculas-Insert pega desde cooedit.clip.
- Mayúsculas-Supr corta a cooedit.clip, y
- Ctrl-Supr elimina el texto resaltado.
- La tecla de terminación también realiza un Return con un sangrado automático

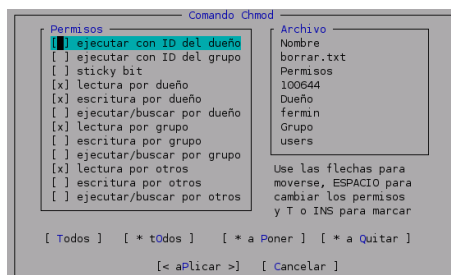
La selección con ratón también funciona.

Copiar (F5) Sobreimpresiona una ventana de entrada con destino por defecto al directorio del panel no seleccionado y copia el fichero actualmente seleccionado (o los ficheros marcados, si hay al menos uno marcado) al directorio especificado por el usuario en la ventana.

cambiar Permisos ... (C-x c) Permite cambiar los permisos de los ficheros. La ventana de Chmod se usa para cambiar los bits de atributo en un grupo de ficheros y directorios. Puede ser invocada con la combinación de teclas C-x c. Si sólo tenemos seleccionado un archivo, la ventana que aparece es:



Si realizamos una selección múltiple de archivos la ventana de Chmod tiene dos partes - Permisos y Archivo



En la sección Archivo se muestran el nombre del fichero o directorio y sus permisos en formato numérico octal, así como su propietario y grupo.

En la sección de permisos hay un grupo de campos que corresponden a los bits de atributos del fichero. Conforme cambiamos los bits de atributo, podemos ver el valor octal cambiando en la sección Archivo.

Para aceptar los atributos, usaremos la tecla **Enter**. Cuando trabajamos con un grupo de ficheros o directorios, basta pulsar con el ratón sobre los bits que queremos activar o desactivar. Una vez seleccionados los bits que queremos cambiar, seleccionamos uno de los botones (Marca activa o marca desactiva).

Finalmente podemos usar:

[Todos], que actuará sobre todos los ficheros marcados.

[* Todos] actúa sólo sobre los atributos marcados de los ficheros seleccionados

[* a Poner] Activa los bits marcados de los atributos de los ficheros seleccionados

[* a Quitar] Borra los bits marcados de los atributos de los ficheros seleccionados

[<aPlicar>] Activa los atributos de un fichero

[Cancelar] cancela el comando Chmod.

crear enLace (C-x l) Crea un enlace al fichero actual.

crear enlace Simbólico (C-x s) Para aquellos que no conozcan qué son los enlaces: crear un enlace físico o duro (hard) a un fichero es algo parecido a copiar el fichero, salvo que el fichero original y el destino representan el mismo fichero físico, los mismos datos reales. Por ejemplo, si editamos uno de esos ficheros, todos los cambios que realicemos aparecerán en ambos ficheros. Hay quien llama a los enlaces alias o accesos directos.

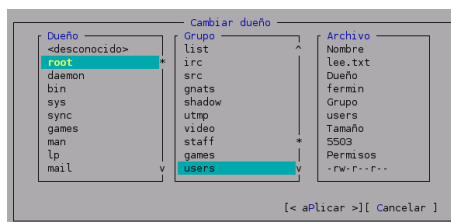
Un enlace físico aparece como un fichero real. Después de crearlo, no hay modo de decir cuál es el original y cuál el enlace. Si borramos uno de ellos el otro aún seguirá intacto. Es muy difícil advertir que los ficheros representan la misma imagen. Usaremos estos enlaces cuando no necesitemos saberlo.

Un enlace simbólico es una referencia al nombre del fichero original. Si el fichero original se borra, el enlace simbólico queda sin utilidad. Es bastante fácil advertir que los ficheros representan la misma imagen.

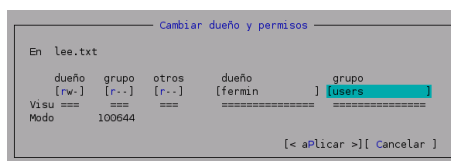
Midnight Commander muestra un símbolo "@" delante del nombre del fichero si es un enlace simbólico a alguna parte (excepto a un directorio, caso en que muestra una tilde (~)). El fichero original al cual el enlace apunta es mostrado en la línea de estado si la opción Mini status está habilitada. Usaremos enlaces simbólicos cuando queramos evitar la confusión que pueden causar los enlaces físicos.

ediTar enlace simbólico (C-x C-s) Permite modificar un enlace simbólico.

cambiar dueño ... (C-s o) Permite cambiar los permisos de los ficheros. El comando Chown permite cambiar el propietario/grupo de un fichero. La tecla rápida para este comando es C-x o.



cAmbiar dueño y permisos ... Permite cambiar al fichero seleccionado el dueño y los permisos



Renombrar/mover... (F6) Sobreimpresiona una ventana de entrada que por defecto apunta al directorio en el panel no seleccionado y mueve el fichero actualmente seleccionado (o los ficheros marcados si hay al menos uno) al directorio especificado por el usuario en la ventana. Durante el proceso, podemos pulsar C-c ó ESC para anular la operación.

crear Directorio (F7) Sobreimpresiona una ventana de entrada y crea el directorio especificado.

Borrar (F8) Borra el fichero actualmente seleccionado o los ficheros marcados en el panel activo.

caMbiar directorio... (M-c) Usaremos el comando Cambiar de directorio si tenemos llena la línea de comandos y queremos hacer un cd a algún lugar.

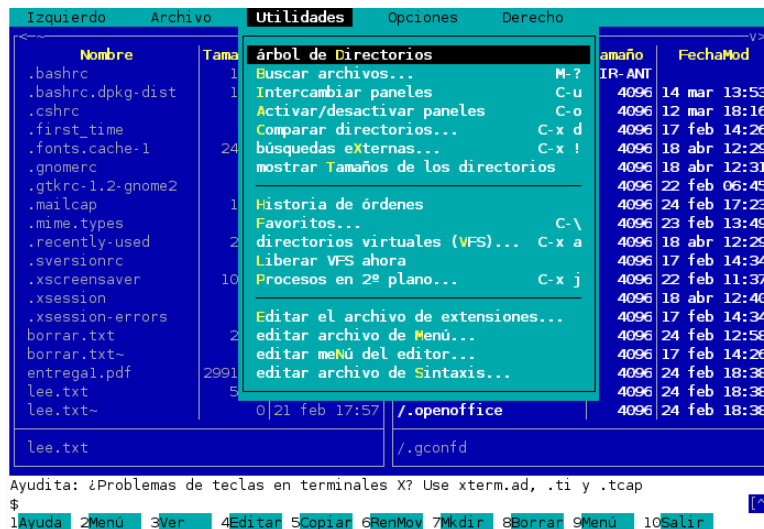
seleccionar Grupo ... (+) Se usa para seleccionar (marcar) un grupo de ficheros. Para marcar directorios en vez de ficheros, la expresión debe empezar o terminar con '/'.

de-seleccionar grUpo (\) Utilizado para deseleccionar un grupo de ficheros. Es la operación antagonista al comando Selecciona grupo.

Invertir la selección (*) Si queremos invertir los ficheros seleccionados. Deselecciona los marcados y marca los no marcados.

saliR (F10) Finaliza *Midnight Commander*.

3.2.3. Menú de Utilidades

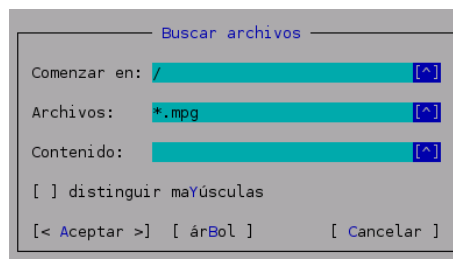


árbol de Directorios Muestra una ventana con estructura de árbol con los directorios.

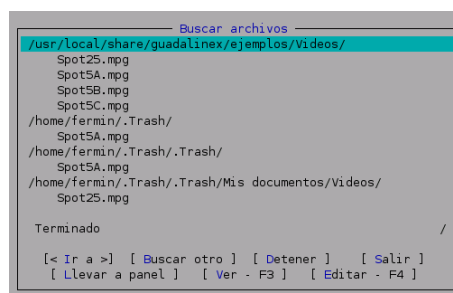
Buscar archivos... (M-?) Permite buscar un fichero específico o ficheros con un contenido determinado.

➔ Para practicar

Por ejemplo, vamos a buscar los ficheros de vídeo en formato mpg que tenemos. Tendremos que indicarle dónde debe comenzar a buscar (/ le estamos indicando que en raíz); qué tipo de archivos (*.mpg - todos los que tengan de extensión mpg)



Nos devolverá los siguientes resultados:



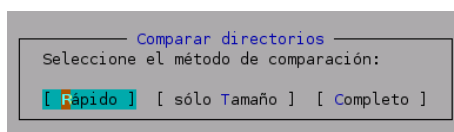
Desde aquí podemos:

- [Ir a] la localización donde se encuentra el fichero marcado en video inverso
- [Buscar otro] realizar una nueva búsqueda
- [Detener] la búsqueda
- [Salir] de la operación que estamos realizando
- [Llevar a panel] los resultados de la búsqueda al panel de mc
- [Ver - F3] si es posible, ya que en este caso, al ser un vídeo no lo podremos ni ver, ni editar en modo texto, lógicamente
- [Editar - F4] idem

Intercambiar paneles (C-u) Intercambia los contenidos de los dos paneles de directorios.

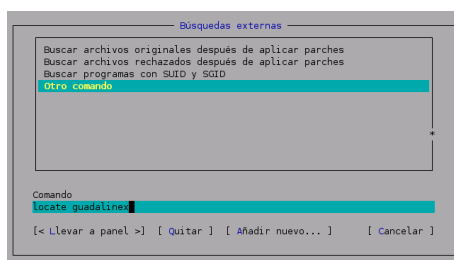
Activar/desactivar paneles (C-o) Muestra la salida del último comando del shell.

Comparar directorios... (C-x d) Compara los paneles de directorio uno con el otro. Hay tres métodos de comparación:



- El método rápido compara sólo el tamaño de fichero y la fecha.
- El método completo realiza una comparación completa octeto a octeto.
- El método de comparación de sólo tamaño sólo compara los tamaños de fichero y no chequea los contenidos o las fechas.

búsquedas eXternas ... (C-x !) Con este comando podemos ejecutar un programa “externo” y la salida de ese programa se visualiza en el panel actual.



Por ejemplo, si utilizamos el comando

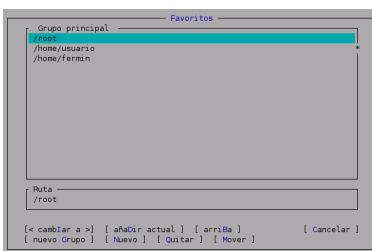
`locate guadalinux`

la salida del comando aparecerá en el panel marcado en ese momento. En este caso, el comando lo que hace es buscar ficheros/directorios cuyo nombre sea guadalinux.

mostrar Tamaños de los directorios

Historia de órdenes Muestra una lista de los últimos comandos utilizados. El comando seleccionado se copia a la línea de comandos.

Favoritos (C-\) Realiza el cambio desde el directorio actual a los directorios seleccionados por nosotros que usemos más a menudo de forma más rápida.



directorios virtuales (VFS) ... (C-x a) Nos permite acceder más rápido a nuestro directorio de usuario.

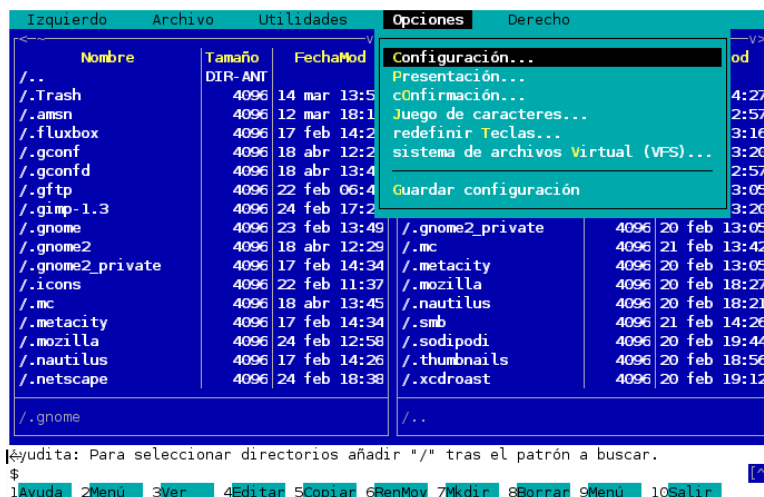
Procesos en 2º plano... (C-x j) Permite controlar el estado de cualquier proceso de *Midnight Commander* en segundo plano. Podemos parar, reiniciar y eliminar procesos en segundo plano desde aquí.

Editar el archivo de extensiones ... Nos permite especificar los programas a ejecutar para intentar ejecutar, ver, editar y realizar un montón de cosas sobre ficheros con ciertas extensiones (terminaciones de fichero). Por ejemplo, asociar la extensión de los ficheros de audio de SUN (.au) con el programa reproductor adecuado. Las extensiones de ficheros permiten que al pulsar la tecla **Intro** o hacer un doble clic sobre un fichero, intentar verlo, editarlo o arrastrar con el ratón otro fichero encima, se ejecute la aplicación seleccionada en el fichero de extensiones para la extensión en cuestión. En el fichero de extensiones debe existir una línea de entrada para cada extensión que queramos que el *Midnight Commander* interprete, esa entrada comienza en la columna primera; las líneas que siguen y que empiezan con un espacio en blanco o una tabulación, son las diferentes acciones que queremos definir para esa extensión.

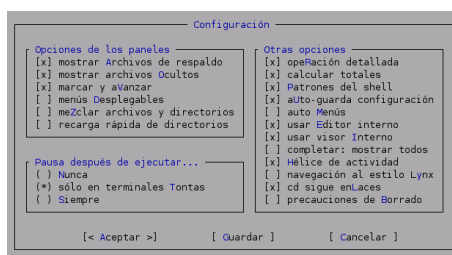
editar archivo de Menú ... El menú de usuario es un menú de acciones útiles que puede ser personalizado por el usuario. Cuando accedemos al menú de usuario, se utiliza si existe el fichero .mc.menu del directorio actual, pero sólo si es propiedad del usuario o del root y no es modificable por todos. Si no se encuentra ese fichero, se intenta con ~/.mc/menu, y si no, mc utiliza el menú por defecto para todo el sistema /usr/lib/mc/mc.menu.

3.2.4. Menú de Opciones

El comando Opciones muestra un diálogo desde el cual podemos cambiar la configuración de *Midnight Commander*.



Configuración ... Permite acceder a una ventana con la que podemos activar o desactivar algunas opciones para configurar el programa. Las opciones están activas si tienen un asterisco o "x" delante. Esas opciones están divididas en tres grupos: Opciones de los Paneles, Pausa después de ejecutar y Otras Opciones.



Opciones del panel:

- **mostrar Archivos de respaldo:** por defecto, *Midnight Commander* no muestra ficheros terminados en '~' (copias de seguridad).
- **mostrar archivos Ocultos:** *Midnight Commander* mostrará todos los ficheros que comienzan con un punto (como `ls -a`).
- **marcar y aVanzar:** Por defecto, cuando marcamos un fichero (con C-t o la tecla Insert) la barra de selección se desplaza hacia abajo.
- **mezclar archivos y directorios:** cuando esta opción está habilitada, los ficheros y directorios se muestran mezclados. Si la opción está desactivada, los directorios (y enlaces a los mismos) se muestran al principio de la lista, y el resto de ficheros a continuación.
- **recarga rápida de directorios:** esta opción está desactivada por defecto. Si la activamos, *Midnight Commander* usará un truco para determinar si los contenidos del directorio han cambiado. El truco consiste en recargar el directorio sólo si el nodo-i del directorio ha cambiado; esto significa que las recargas suceden sólo cuando los ficheros son creados o borrados. Si lo que cambia es el nodo-i de un fichero en el directorio (cambia el tamaño del fichero, cambia el modo o propietario, etc) la pantalla no se actualiza. En esos casos, si tenemos la opción activada, deberemos releer el directorio manualmente (con C-r).

Pausa después de ejecutar: Tras ejecutar nuestros comandos, *Midnight Commander* puede quedarse en 2º plano, de tal modo que podamos examinar la salida del comando. Hay tres posibles selecciones para esta variable:

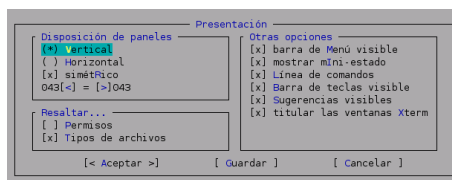
- Nunca: significa que no queremos ver la salida de nuestros comandos. Si estamos utilizando la consola Linux o un xterm, podremos ver la salida del comando pulsando C-o.
- sólo en terminales Tontas: obtendremos un mensaje de pausa y que no son capaces de mostrar la salida del último comando ejecutado (en realidad, cualquier terminal que no sea un xterm o una consola de Linux).
- Siempre: el programa realizará una pausa después de ejecutar todos nuestros comandos.

Otras opciones:

- operación detallada: hace que las operaciones de copia, renombrado y eliminación de ficheros sean detalladas (p.e., muestra un cuadro de diálogo para cada operación). Si tenemos un terminal lento, podríamos querer desactivar la operación detallada. Ésta es automáticamente desactivada si la velocidad de nuestro terminal es menor de 9600 bps.
- calcular totales: si tenemos activa esta opción el *Midnight Commander* suma el total del tamaño de los ficheros así como el número de ficheros antes de cualquier operación de copia, renombrado o borrado. Se obtendrá una barra de progreso más exacta en estos procesos a cambio de pérdida de velocidad. Esta opción no tiene efecto si no está activa operación detallada.
- Patrones del shell: por defecto los comandos de Selección, Deselección y Filtro usarán expresiones regulares del estilo del shell. Para realizar esto se hacen las siguientes conversiones: el '*' es remplazado por cero o más caracteres; la '?' por exactamente un carácter y '.' por el punto literal.
- auto-guarda configuración: si esta opción está activada, cuando salimos de *Midnight Commander* las opciones configurables de *Midnight Commander* se guardan en el fichero `~/.mc/ini`.
- auto Menús: Si está activada, el menú de usuario es invocado al arrancar. Útil para menús contruidos por personas ajenas a Unix.
- usar Editor interno: si está activada, el editor de ficheros incorporado es utilizado para editar ficheros. Si está desactivada, se usará el editor especificado por la variable de entorno EDITOR. Si no se especifica ninguno, se usará vi como editor de ficheros interno.
- completar: mostrar todos: por defecto el *Midnight Commander* muestra todas las posibles maneras de completarse si hay ambigüedad al darle a M-Tab dos veces, la primera vez completa todo lo posible y pita en caso de ambigüedad. Para poder ver todas las posibles maneras de completarse después de presionar M-Tab la primera vez hay que activar esta opción.
- Hélice de actividad: cuando está activa muestra un guión rotando en la esquina superior derecha para indicar que hay un trabajo en curso.
- navegación al estilo Lynx: si está activa podemos usar las flechas del teclado para hacer chdir automáticamente siempre que la selección actual sea un subdirectorio y la línea de comandos esté vacía.
- cd sigue los enlaces: si está seleccionada, hace que *Midnight Commander* siga la "cadena lógica" de directorios. Cuando está deseleccionada, *Midnight Commander* sigue la estructura real de directorios, así si hemos entrado en un directorio a través de un enlace y ejecutamos el comando `cd ..`, este comando nos trasladará al padre real del directorio actual y no al directorio donde se encontraba el enlace.
- Precauciones de Borrado: si está activa, provocará que *Midnight Commander* pida confirmación cuando borremos ficheros.

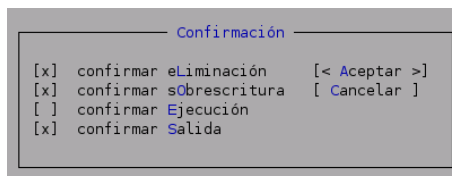
- `cd` sigue los enlaces: si está seleccionada, hace que *Midnight Commander* siga la “cadena lógica” de directorios. Cuando está deseleccionada, *Midnight Commander* sigue la estructura real de directorios, así si hemos entrado en un directorio a través de un enlace y ejecutamos el comando `cd ..`, este comando nos trasladará al padre real del directorio actual y no al directorio donde se encontraba el enlace.

Presentación ... La ventana de presentación nos da la posibilidad de cambiar la presentación general de la pantalla.

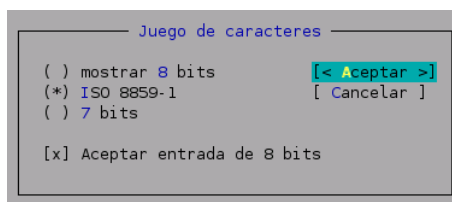


Por ejemplo, podemos especificar cuándo son visibles la Barra de Menú, la línea de comandos, la línea de sugerencias de xterm y la Barra de teclas de Función. Además, podemos dividir la pantalla en dos paneles verticales u horizontales, la división puede ser simétrica o bien podemos indicar una división asimétrica.

cOnfirmación ... Accedemos a un diálogo desde el cual podemos especificar qué acciones queremos que sean confirmadas antes de ser realizadas.



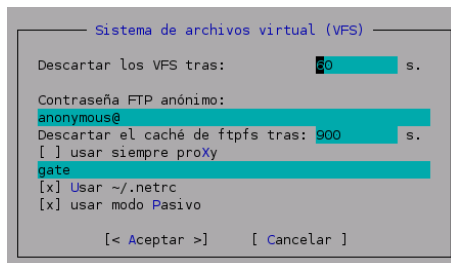
Juego de caracteres ... Desde él podemos seleccionar qué caracteres es capaz de visualizar nuestro terminal.



Si queremos escribir y visualizar correctamente en castellano (acentos y ñ) tendremos que tener activas las opciones del gráfico.

redefinir Teclas ... Este comando nos permite comprobar si nuestras teclas F1-F20, Inicio, Fin, etc. funcionan adecuadamente en nuestro terminal.

sistema de archivos Virtual (VFS)... Muestra un diálogo desde el cual podemos especificar algunas opciones relacionadas con VFS¹⁰ (Sistema de Archivos Virtual).



Esta opción nos proporciona el control sobre la caché de información del Sistema de Ficheros Virtual (VFS).

Midnight Commander guarda en memoria la información relacionada con alguno de los sistemas de ficheros virtuales para acelerar el acceso a los ficheros en el sistema de ficheros. Para acceder a los contenidos de ficheros comprimidos (por ejemplo, los ficheros tar comprimidos) este programa debe crear un archivo temporal descomprimido en el disco. Como la información en memoria y los archivos temporales ocupan recursos podríamos querer ajustar los parámetros de la información con caché para disminuir la utilización de memoria o aumentar la velocidad de acceso a los sistemas de ficheros más utilizados.

El sistema de ficheros Tar es bastante inteligente a la hora de manejar sus ficheros: sólo carga las entradas de los directorios y cuando necesita usar la información contenida en el fichero tar, va y la toma.

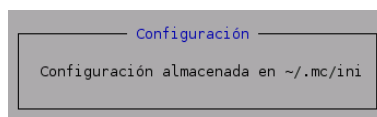
Los ficheros tar normalmente se guardan comprimidos, y debido a la naturaleza de esos ficheros (las entradas de directorio de los ficheros tar no están allí esperando a que las carguemos nosotros), el sistema de ficheros tar tiene dos posibilidades: cargar todo el fichero tar descomprimido en memoria o descomprimir el fichero en el disco en una localización temporal y acceder entonces al fichero descomprimido como a un fichero tar normal.

Ahora, dado que a todos nos encanta navegar por los ficheros, incluidos los tar, sobre el disco, es común que salgamos de un fichero tar y volvamos a entrar en él después. Puesto que la descompresión es lenta, *Midnight Commander* mantendrá en memoria la información durante una cantidad de tiempo limitado, después de alcanzado el momento, todos los recursos de memoria asociados con el sistema de ficheros serán liberados. El período por defecto es de un minuto.

El FTP File System mantiene el listado del directorio en la caché. El tiempo de finalización de la caché lo podemos configurar, un valor bajo para esta opción puede hacer más lenta cualquier operación en el FTP File System, porque cualquier operación va acompañada por una pregunta del servidor ftp.

Además, podemos definir un proxy para hacer transferencias ftp y configurar *Midnight Commander* para que siempre lo use.

Guardar configuración Guarda los valores actuales de los menús Izquierdo, Derecho y Opciones. También se guarda un pequeño grupo de otros valores.



Si activamos la opción Auto-guarda configuración, MC guardará siempre la configuración actual al salir.

Existen también configuraciones que no pueden ser cambiadas desde los menús. Para cambiarlas tendremos que editar el fichero de configuración.

¹⁰Véase ??

3.3. Barra inferior

Desde aquí podremos ejecutar las acciones más comunes y cotidianas:

1 Ayuda 2 Menú 3 Ver 4 Editar 5 Copiar 6 RenMov 7 Mkdir 8 Borrar 9 Menú 10 Salir

Podemos acceder a ellas haciendo “clic” con el botón izquierdo de nuestro ratón o pulsando las teclas de función desde [F1] a [F10]

3.4. Ejecutando Comandos del Sistema Operativo

Podemos ejecutar comandos tecleándolos directamente en la línea de entrada de *Midnight Commander*, o seleccionando el programa que queremos ejecutar con la barra de selección en uno de los paneles y pulsando Enter.

Si pulsamos Enter sobre un fichero que no es ejecutable, *Midnight Commander* chequea la extensión del fichero seleccionado con las extensiones en el Fichero de Extensiones. Si se produce una coincidencia se ejecutará el código asociado con esa extensión.

3.4.1. El Comando cd Interno

El comando cd es interpretado por *Midnight Commander*, no es pasado al shell de comandos para ser ejecutado.

Substitución de Tilde La tilde (~) será substituida por nuestro directorio de inicio, si añadimos un nombre de usuario tras la tilde, entonces será substituido por el directorio de entrada al sistema del usuario especificado.

Directorio Anterior Podemos saltar al directorio donde estábamos anteriormente de la mano del nombre de directorio especial ‘-’ del siguiente modo:

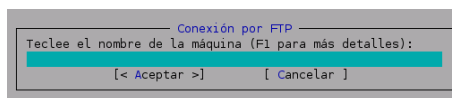
```
cd -
```

3.5. Sistema de Ficheros Virtual

El selector del Sistema de Ficheros Virtual (SFV) permite a *Midnight Commander* manipular ficheros no localizados en el sistema de ficheros Unix.

3.5.1. Sistema de Ficheros FTP

El ftpfs permite manipular ficheros en máquinas remotas, para usarlo ahora, deberíamos intentar usar el comando del panel “Conexión FTP” (accesible desde los menús laterales iz. y der.)



en el campo que aparece escribiremos:

```
usuario:contraseña@máquina
```

cuando conectemos en el panel que teníamos seleccionado nos aparecerá la lista de ficheros de la máquina remota. Luego podremos navegar y copiar ficheros como si todo estuviera pasando en nuestro propio ordenador.

También podemos realizar directamente la conexión utilizando el comando `cd` con la ruta:

```
ftp://[!][usuario[:contraseña]@]maquina[:puerto][directorio- remoto]
```

los elementos, usuario, puerto y directorio-remoto son opcionales (Entre corchetes, []). Si especificamos el elemento usuario, entonces *Midnight Commander* intentará entrar en la máquina remota como ese usuario, en otro caso usará nuestro login. El elemento opcional contraseña, si está presente es la contraseña de acceso usada para autenticar la conexión.

Ejemplos:

```
cd ftp://ftp.nuclecu.unam.mx/linux/local
cd ftp://tsx-11.mit.edu/pub/linux/packages
cd ftp://!behind.firewall.edu/pub
cd ftp://guest@remote-host.com:40/pub
cd ftp://miguel:xxx@server/pub
cd ftp://ftp.um.es/pub
```

Para acceder a lugares tras un cortafuegos usaremos el prefijo `ftp://!` (es decir, con una exclamación tras la doble barra) para hacer que *Midnight Commander* utilice un proxy para realizar la transferencia ftp. Pondremos el proxy en el cuadro de diálogo Sistema de Ficheros Virtual.

3.5.2. Sistema de Archivos Tar

El sistema de ficheros tar y los ficheros tar comprimidos pueden consultarse usando el visor interno, es decir, seleccionando el fichero y pulsando F3. *Midnight Commander* permite navegar por ficheros del tipo *.tgz*, *tar.gz*, *.Z*, *.rpm*, *.deb*, etc. Si accedemos a ficheros empaquetados, podemos navegar por los subdirectorios que pudieran contener, además podemos realizar algunas operaciones sencillas como copiar ficheros a y desde un directorio normal, o leer los contenidos de ficheros de texto.

Para ficheros tar también es posible usando el comando `chdir`. Para cambiar nuestro directorio a un fichero tar, cambiaremos nuestro directorio actual al fichero tar utilizando la sintaxis:

```
tar:NombreDeFichero.tar[Directorio-dentro_de-tar]
```

Normalmente basta con apuntar a un fichero tar y pulsar Return para entrar en el fichero tar.

3.5.3. Sistema de Ficheros de Red

El sistema de ficheros de *Midnight Commander* es un sistema de ficheros de red básico que nos permite manipular ficheros en una máquina remota como si estuviesen accesibles localmente. Para utilizar esto, la máquina remota debe estar ejecutando el programa servidor `mcserv`.

Para conectar a una máquina remota, sólo necesitamos hacer un `chdir` a un directorio especial cuyo nombre sigue el siguiente formato:

```
mc:[usuario@]máquina[:puerto][directorio-remoto]
```

Los elementos usuario, puerto y directorio-remoto son opcionales. Si especificamos el elemento usuario entonces *Midnight Commander* intentará acceder a la máquina como ese usuario, si no usará nuestro login.

El elemento puerto es utilizado cuando la máquina remota se ejecuta en un puerto especial (véase la página del manual de `mcserv` para mayor información sobre puertos); finalmente, si el elemento directorio remoto está presente, nuestro directorio actual en la máquina remota será éste.

Ejemplos:

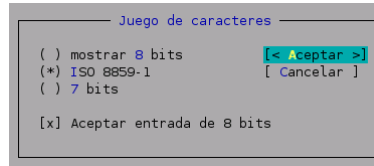
```
mc:ftp.nuclecu.unam.mx/linux/local
mc:Pepe@foo.edu:11321/privado
mc:Gabriel@Dersu.ArocaSoft.es:1235/SolidarityWare
```

➡ Para practicar

No es necesario conocer a fondo todo lo expuesto para utilizar el programa. Si controlamos las cuestiones siguientes será suficiente por ahora:

1. Cambiar las opciones del programa mc para que permita ver y escribir acentos. ¿Cómo hacerlo?
Sencillo

Opciones→Juego de caracteres



2. Editar con el programa mc (o con mcedit) el fichero `/etc/issue` y cambiar el mensaje de inicio en modo texto a:

Hola desde Guadalinex

Kernel \r en un \m

- Con el mc
 - # mc
 - Marcar el fichero en vídeo inverso y **Fichero→Editar** o **[F4]+**
 - Con el programa mcedit:
 - # mcedit /etc/issue
3. Entrando como root vamos a seleccionar el directorio `/etc/sysconfig` y copiarlo a nuestro directorio de root. Para hacer ésto en el panel izquierdo seleccionaremos el citado directorio mientras que en el derecho nos situaremos en el del root, luego sólo tenemos que pulsar **[F5]** y **Aceptar**
 4. Seleccionar los ficheros del directorio antes copiado (**[+]** o **Fichero→seleccionar Grupo**) y borrarlos con **[F8]** (manteniendo el directorio).

Capítulo 4

Guadalinex como cliente de red

Cuando trabajas con Linux estás ante un sistema operativo orientado al trabajo con redes de ordenadores. ¿Qué nos empuja a poder afirmarlo tan categóricamente? Ya te darás cuenta poco a poco. (*Manual Avanzado de Linux* de RAÚL MONTERO RIVERO, Ed. Anaya)

Las computadoras son mucho más útiles cuando están conectadas a una red, pero, desgraciadamente, estas redes hacen que las computadoras estén expuestas a un gran número de accesos no autorizados, y los sistemas Linux no son inmunes a este tipo de actividades. (*Hackers en Linux*, BRIAN HATCH y otros)

4.1. Introducción

Linux, tal como lo conocemos, es posible gracias a “La Red”¹ y “La Red” se ha expandido también en gran parte, gracias a Linux. En este capítulo vamos a estudiar algunos de los programas que nos permiten trabajar en Red con nuestro Guadalinex. Algunos de ellos no podremos probarlos a no ser que tengamos un servidor que nos permita la conexión, así que no todas las cuestiones tratadas aquí se podrán trabajar por igual. Dependerá de cómo esté conectado nuestro equipo y de los servicios de red a los que tengamos acceso para poder trabajar un número mayor de aspectos de este tema.

Hay determinados servicios de red que seguramente se están trabajando ya con aplicaciones más que conocidas y comentadas en el curso: el “todoterreno” Mozilla², Ximian Evolution, etc. Así que sólo analizaremos aquellos programas o utilidades “novedosas” a estas alturas del curso.

En todos los ejemplos de este capítulo hemos supuesto que la conexión la hacemos con un usuario de nombre `usuario` y una máquina llamada `tux.midominio.org`. Para conexiones reales tendremos que adecuar estos datos a los de cada uno en particular.

Comencemos esta sección con:

4.2. Otros navegadores Web

Hay un par de “perlas” instaladas en nuestro sistema que muestran su valía cuando estamos obligados a trabajar con un navegador web que no consuma recursos del sistema (por ejemplo cuando navegamos por la red usando otra máquina a la que a su vez nos hemos conectado en red), se trata de `lynx`. Lynx es un

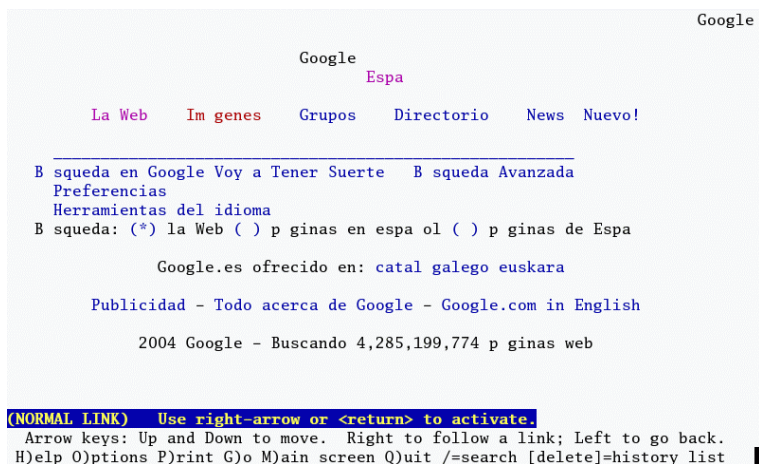
¹Nos referimos a Internet

²Con Mozilla disponemos de una aplicación muy personalizable para

- Navegar por la web
- Gestionar nuestras cuentas de Correo y Noticias
- Crear páginas web

navegador Web un tanto especial, si queremos usarlo para buscar algo en Google, una vez conectados sólo hay que escribir³

```
# lynx http://www.google.es
```

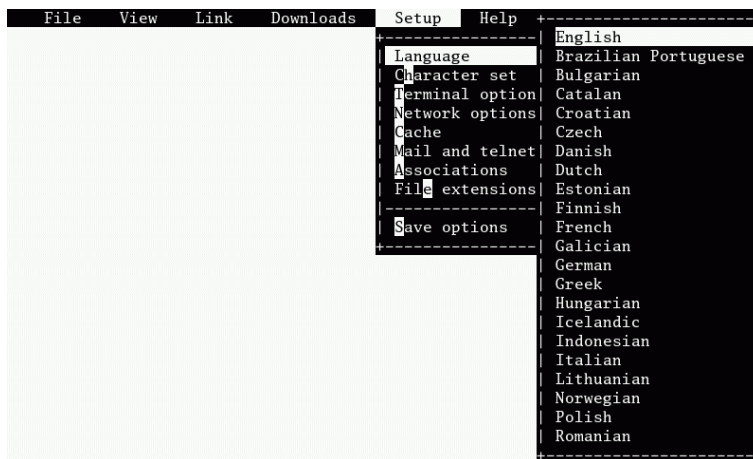


Si deseamos visitar otra página escribiremos `g` y después la URL a la que deseamos acceder. Para movernos por él hemos de usar los cursores y la barra espaciadora y para salir la letra `q`.

Otro navegador, más evolucionado que el anterior y también en modo texto⁴, `links`, para ejecutarlo

```
$links
```

Tiene un menú de contexto al que podemos acceder con la tecla de función **F9**⁵. En primer lugar deberíamos ponerlo en castellano desde el menú **Setup**



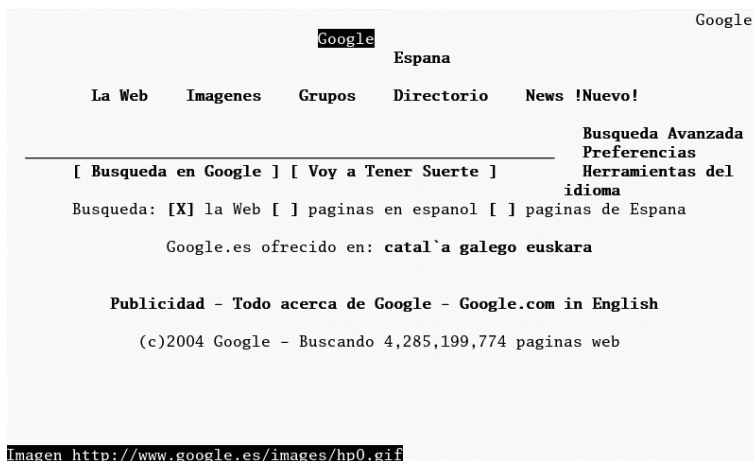
y guardar la configuración (**Save options**). Para conectar con una web usaremos el menú **Fichero** y en URL escribiremos la URL de la página. Otra vez Google

³También podemos iniciarlo sin escribir la dirección y acceder a ella una vez en el programa usando la tecla `g`

⁴No pensaréis que solo existen navegadores en modo texto. ¿Qué os parecería poder bajar el correo también de esta forma? Para ese cometido podemos usar el cliente de correo en texto plano `mutt` (un cliente de correo en texto plano es aquel que solo permite componer mensajes de correo electrónico en texto en formato ASCII).

Un mini manual sobre su uso lo podéis encontrar en <http://linux-cd.com.ar/manuales/rh9.0/rhl-gsg-es-9/sl-eclients-textmail.html>.

⁵También si pulsamos con el ratón sobre la zona en la que aparece el menú.



Con las teclas de cursor o con el menú superior podremos navegar por nuestras web favoritas. Para salir, de nuevo, podemos usar la letra q.

4.3. Telnet y ssh

4.3.1. Acceso remoto: telnet

Dentro de las labores de un administrador de sistemas está el acceso remoto a los mismos, ya sea para buscar información en algún fichero del sistema, para copiar información o ejecutando en remoto algún comando.

Usando `telnet`⁶ podemos acceder a una máquina remota de la misma forma que lo haríamos si estuviéramos sentados delante de la consola y utilizásemos su teclado para introducir los comandos. Los comandos que se teclean por parte del usuario son transmitidos directamente a la máquina remota y la respuesta de ésta es mostrada en la pantalla del usuario. De esta forma el sistema local es transparente al usuario, el cual tiene la sensación de estar conectado directamente a la máquina remota⁷.

Para que podamos iniciar una sesión `telnet` se tienen que dar un par de condiciones:

- Que tengamos una cuenta de usuario en la máquina con la que queremos conectar.
- Que el servidor tenga un servicio de `telnet` activo.

Para acceder al sistema remoto se nos solicitará la identificación para poder entrar al sistema. Por ejemplo⁸ para acceder a la máquina (inexistente) `tux.midominio.org` escribiremos

```
$telnet tux.midominio.org
```

a continuación se nos pedirá el nombre de usuario y la contraseña (de igual forma que si entramos en Guadalinex en modo texto).

➔ **Para practicar** Si podemos acceder a algún servidor vía `telnet`, es interesante probar la posibilidad que nos ofrece Linux de trabajar en modo gráfico con programas situados en otro equipo, para esto tendremos que:

⁶El término `telnet` proviene de *TELEcommunication NETwork*. El punto débil de este tipo de conexiones es que todos los datos se transmitirán en claro en la red. Si un usuario captura los datos que viajan en la red con programas como `tcpdump` o `ettercap` podemos poner en compromiso la seguridad de nuestro sistema.

⁷De esta forma podremos utilizar los recursos de ese ordenador (por ejemplo, ejecutando determinadas aplicaciones matemáticas para las que nuestro ordenador no tiene potencia suficiente).

⁸Previamente deberemos haber establecido la conexión con nuestro proveedor de Internet.

Desde un Xterm de la máquina local ejecutaremos⁹

```
$ xhost +máquina_remota
```

después haremos un telnet a la máquina remota y una vez conectados escribiremos

```
$ export DISPLAY=máquina_local:0
```

por último ya sólo tenemos que ejecutar el comando que deseemos, por ejemplo, podéis probar con

```
$ mozilla &
```

4.3.2. ssh: una solución más segura

ssh Protocolo *Secure Shell*, se usa para conexiones de red cifradas y autenticadas de forma más segura.

Desafortunadamente las conexiones vía telnet tienen un problema grave de seguridad ya que los datos se envían sin cifrar. Así, cualquier intruso puede interceptar nuestros datos y obtener nuestro nombre de usuario y password del sistema además del contenido de la comunicación.

La solución que se adopta es utilizar un sistema alternativo denominado SSH. **ssh** cifra los datos antes de pasarlos a la red, descifrándolos cuando llegan a su destino. El procedimiento de cifrado asegura que el intruso que capture los datos será incapaz de descifrarlos y verlos.

Para iniciar la conexión (seguimos con nuestra máquina ficticia de ejemplo) escribiremos¹⁰:

```
$ ssh -l usuario tux.midominio.org
```

o equivalentemente

```
$ ssh usuario@tux.midominio.org
```

La primera vez que conectemos aparecerá

```
The authenticity of host 'tux.midominio.org (xx.xx.xx.xx)' can't be established.
RSA key fingerprint is 49:8c:9c:10:a9:c5:5d:e2:cd:88:65:f0:dc:02:f4:cf.
Are you sure you want to continue connecting (yes/no)?
```

Escribimos **yes** e **Intro**. Cuando siga, y aparezca

```
Warning: Permanently added 'tux.midominio.org' (RSA) to the list of known hosts.
usuario@tux.midominio.org's password:
```

será el momento de introducir la contraseña.



No pensemos que es algo para “hackers” y que no nos puede afectar. En la actualidad¹¹, los ordenadores de los centros TIC permiten conexiones **ssh**. De esa forma, como los alumnos conocen ya las IPs de las máquinas inician sesiones con otros ordenadores del instituto (por ejemplo, el de la mesa del profesor) y pueden borrar o “trastear” sobre ellos.

Por ejemplo, supongamos que la IP de la mesa del profesor es 192.168.3.24, un alumno/a escribirá:

```
$ssh usuario@192.168.3.24
```

⁹Donde *máquina_remota* es o bien la dirección IP de la máquina remota, o bien el nombre de esa máquina

¹⁰Cada servidor SSH tiene un identificador único y secreto, llamado *host key*, para identificarse frente a los clientes que se conectan. La primera vez que nos conectamos a un servidor, la parte pública de la *host key* se copia en nuestra cuenta local (asumiendo que respondemos **yes**). Cada vez que nos conectemos a este servidor, el cliente SSH comprobará la identidad del servidor remoto con esta clave pública. Dicha clave pública, así como la del resto de máquinas con las que nos vayamos conectando se encuentra guardada en `$HOME/.ssh/known_hosts`

¹¹Se está en fase de solucionarlo.

como todos los ordenadores tienen un sólo usuario de nombre y contraseña `usuario`, cuando se le solicite escribirá los datos que le permiten autenticarse e iniciará una conexión con la máquina del “profe”. Si ahora escribe:

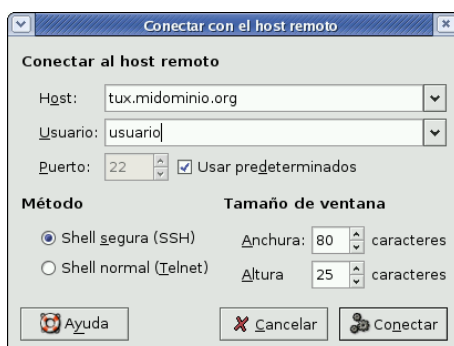
```
$turnoff
```

el “pobre profe” verá que se le apaga la máquina como por arte de magia. Pero puede ser aún peor y que le borren cualquier dato que piensa está a buen recuado en su ordenador.

4.3.3. Conectar en modo comando y (gráfico)

Como siempre la forma sencilla para el final. Podemos conectar vía `telnet` o `ssh` usando el programa `gnome-remote-shell`, para acceder a él podemos usar la cadena de menús de Gnome **Aplicaciones**→**Menú Debian**→**Apps**→**Net**→**Remote Shell** o bien desde una `xterm` escribir

```
$gnome-remote-shell
```



Su uso no presenta problema, sólo hemos de escribir el nombre o IP de la máquina con la que vamos a iniciar la conexión, el tipo de protocolo a usar (`telnet` o `ssh`) y listo, se inicia la conexión.

4.4. FTP y SFTP

4.4.1. ftp

Mediante una conexión `ftp` (*File Transfer Protocol* o Protocolo de Transferencia de Ficheros) podemos cargar y descargar archivos de la red. Este servicio puede verse dividido en dos partes:

- Los usuarios con cuenta en el sistema pueden acceder a su propio sistema de archivos y cargar y descargar información.
- Utilización anónima. En este caso pueden copiarse los ficheros de un servidor, a través de FTP, sin necesidad de usar una contraseña. En general, si nuestra conexión es anónima se nos informará al entrar en el sistema de que se nos aplican ciertas restricciones y que sólo podremos ver aquellas zonas del sistema de ficheros que permiten este tipo de acceso.

Para iniciar en modo comando una sesión `ftp` de este tipo escribiremos:

```
$ ftp tux.midominio.org
```

29 Intentemos una conexión con el servicio de `ftp` anónimo de `rediris`

```
$ ftp ftp.rediris.es
Connected to ftp.rediris.es (130.206.1.5).
220-=(<*>)=-.:. (( Welcome to ftp.rediris.es )) .:.-=(<*>)=
220-You are user number 199 of 1500 allowed
220-<<
220-Bienvenido al FTP anónimo de RedIRIS.
220-Welcome to the RedIRIS anonymous FTP server.
220-
220-Este servidor FTP se ejecuta en una Ultra Enterprise 450 con 4
220-procesadores conectados a varios dispositivos de almacenamiento
220-que totalizan una capacidad superior a 1.8 Terabytes.
220-Parte del hardware fué donado amablemente por Sun Microsystems.
220-This server runs on a 4-processor Sun Ultra Enterprise 450
220-connected different storage devices totalizing 1.8+ Terabytes.
220-Part of the hardware was kindly donated by Sun Microsystems.
220-
220-Este FTP es de acceso público y funciona gracias a la infraestructura
220-(máquinas y técnicos) de Red y Sistemas de Información de RedIRIS;
220-es accesible desde todo el mundo gracias a todas las personas
220-involucradas en el desarrollo de la Internet.
220-
220-Localice ficheros en: http://sunsite.rediris.es/busquedas/?lang=es
220-
220-Find files at: http://sunsite.rediris.es/busquedas/index.en.php?lang=en
220->>
220-Local time is now 20:54 and the load is 2.45. Server port: 21.
220-Only anonymous FTP is allowed here
220 You will be disconnected after 5 minutes of inactivity.
Name (ftp.rediris.es:paco):
```

Como nuestra conexión es anónima escribiremos que somos el usuario anonymous

```
Name (ftp.rediris.es:paco): anonymous
```

para después introducir como contraseña una dirección de correo “válida”

```
Password:
```

y se iniciará la conexión.



Si la conexión no es anónima tendríamos que introducir el nombre del usuario que inicia la conexión así como su palabra de paso.

Para saber qué podemos hacer podemos ejecutar

```
ftp> help
Commands may be abbreviated.  Commands are:

!          debug          mdir          sendport      site
$          dir            mget          put           size
account    disconnect        mkdir         pwd           status
append     exit              mls           quit          struct
ascii      form             mode          quote         system
bell       get              modtime       recv          sunique
binary     glob            mput         reget         tenex
```

bye	hash	newer	rstatus	tick
case	help	nmap	rhel	trace
cd	idle	nlist	rename	type
cdup	image	ntrans	reset	user
chmod	lcd	open	restart	umask
close	ls	prompt	rmdir	verbose
cr	macdef	passive	runique	?
delete	mdelete	proxy	send	

y si deseamos ayuda sobre un comando en concreto

```
ftp> help get
get                receive file
```

Para saber más sobre los comandos del ftp se puede consultar

- El capítulo 9 de la *FAQ sobre Linux para principiantes* http://es.tldp.org/FAQ/FAQ_Linux/Html/FAQ_Linux-9.html
- Estos *Apuntes de ftp*: <http://www.ignside.net/man/ftp/index.php>

Para terminar podemos usar

```
ftp> exit
221 Goodbye. You uploaded 0 and downloaded 0 kbytes.
```

4.4.2. sftp

¿Y qué es el sftp? Se trata de una especie de ftp pero seguro. Es decir, con sftp podemos conectarnos con un servidor de forma similar al clásico ftp, pero en este caso, tanto la autenticación como las transacciones de datos se cifran y aunque algún hacker malvado esté a la “escucha” no podrá obtener nada de nosotros.


Por ejemplo, para que usuario inicie una conexión sftp en modo comando con el servidor tux.midominio.org escribiremos

```
$ sftp usuario@tux.midominio.org
```

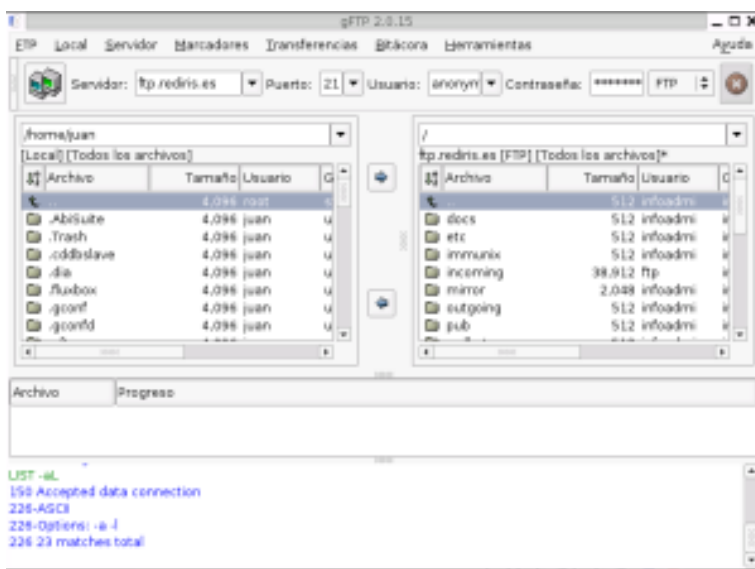
```
paco@eco:~/datos/cursos/4/avanzado/entrega04-2/varios
Available commands:
cd path                Change remote directory to 'path'
lcd path               Change local directory to 'path'
chgrp grp path         Change group of file 'path' to 'grp'
chmod mode path        Change permissions of file 'path' to 'mode'
chown own path         Change owner of file 'path' to 'own'
help                  Display this help text
get remote-path [local-path] Download file
lls [ls-options [path]] Display local directory listing
ln oldpath newpath     Symlink remote file
lnmkdir path           Create local directory
lpwd                  Print local working directory
ls [path]              Display remote directory listing
lumask umask           Set local umask to 'umask'
mkdir path             Create remote directory
put local-path [remote-path] Upload file
pwd                   Display remote working directory
exit                  Quit sftp
quit                  Quit sftp
rename oldpath newpath Rename remote file
rmdir path            Remove remote directory
rm path               Delete remote file
symlink oldpath newpath Symlink remote file
version              Show SFTP version
!command              Execute 'command' in local shell
!                     Escape to local shell
?                     Synonym for help
sftp> help
```

Los comandos de los que disponemos son similares a los del ftp. Pero, ¿por qué ir tan rápido? ¿por qué no vemos ninguno?. Porque para realizar ambos tipos de conexión tenemos una utilidad gráfica¹² que nos puede sacar del atolladero de tener que estudiarlos, se trata de:

4.4.3. gFTP

Guadalinux incorpora el cliente gFTP¹³ que nos permite conexiones en modo ftp y sftp. Se trata de un programa de transferencia de ficheros en modo gráfico que está a nuestra disposición en casi todas las distribuciones de Linux. Podemos acceder a él desde el escritorio con  **Aplicaciones** → **Internet** → **gFTP** o bien desde un terminal con el comando

```
$ gftp &
```



Se ejecuta en una ventana, donde podemos distinguir:

- En la parte superior de la ventana está la barra de menús que permiten acceder a todas las posibilidades del programa.
- Barra de herramientas. En ella indicaremos la dirección del servidor remoto con el que queremos establecer la conexión, el puerto de conexión (el 21 para una conexión FTP, lo pone por defecto), el nombre de usuario y la contraseña que nos identifique¹⁴. Resaltar dos botones:



Para iniciar una conexión una vez introducidos los datos.



Para desconectar



Si, estando conectado, pulsamos de nuevo sobre  se cierra la conexión (podemos hacerlo tam-


bién desde **Servidor** → **Desconectar**). El botón  interrumpe el establecimiento de conexión.

¹²Recordar que con el programa mc (que se estudia en esta entrega) también podemos realizar fácilmente conexiones ftp.

¹³Ya hablamos de él en la segunda entrega

¹⁴Muchos servidores permiten la descarga de ficheros a personas anónimas (que no tienen cuenta en la máquina), en este caso debemos poner como usuario **anonymous** y como contraseña nuestra dirección e-mail.

- Barra de dirección.
- La parte central de la ventana está dividida en dos zonas, el lado izquierdo para el árbol del directorio local y el derecho para el del servidor al que conectemos. El campo superior de este recuadro muestra el directorio activo. Para transferir un fichero basta seleccionarlo y pulsar sobre la flecha de dirección. Desde ellas:

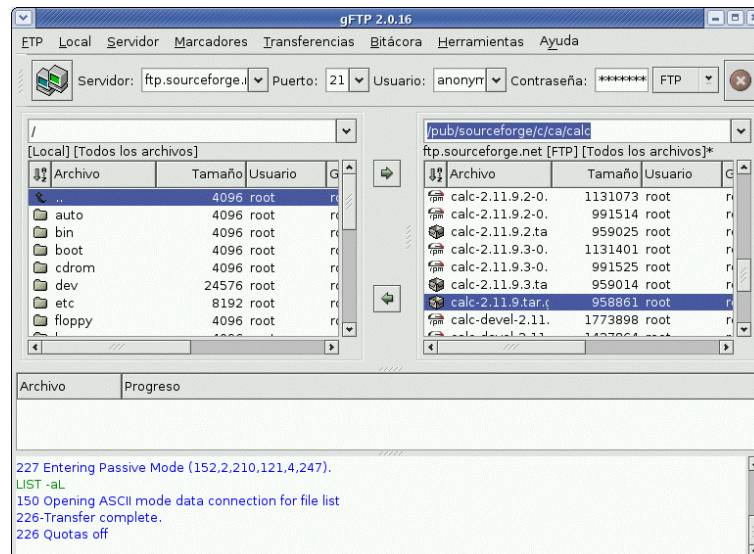
- Pulsando sobre  Archivo Tamaño Usuario Grupo Fecha podemos ordenar los ficheros por nombre, fecha de modificación tamaño, etc
- Si pulsamos sobre una de las ventanas con el botón derecho del ratón obtenemos el mismo resultado que desde **Local** o **Servidor** del menú principal



- Ventana informativa que indica el progreso de las transferencias.
- Ventana de mensajes de la aplicación.

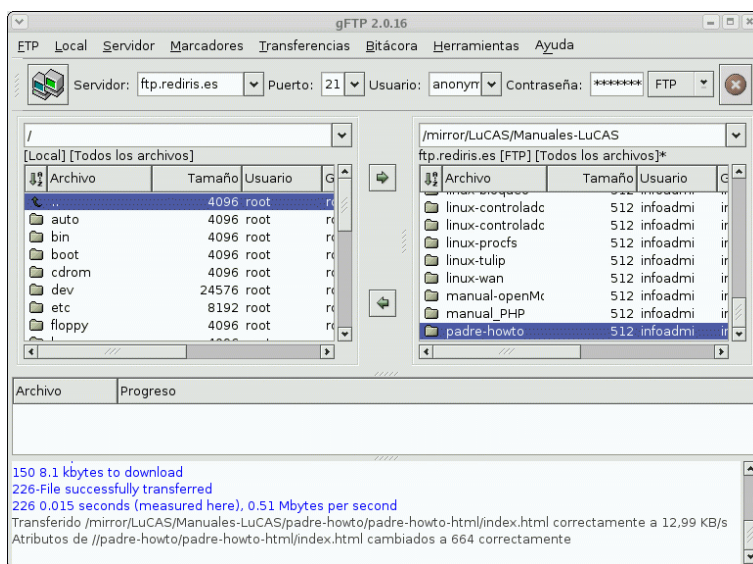
➔ Para Practicar: Rediris, Debian

- Pulsando sobre **Marcadores**→**General Sites** →**Source Forge** del menú principal conseguir llegar hasta la ruta /pub/sourceforge/c/calc/ y bajarse las fuentes (.tar.gz) de la última versión del programa.



Una vez terminada la sesión desconectamos del servidor.

- Conectar con RedIris y bajar el directorio : /mirror/LuCAS/Manuales-LuCAS/padre-howto
Escribiremos en el campo servidor: ftp.rediris.es y, tras movernos por el sistema de ficheros hasta la ruta especificada, bajaremos el directorio:



Conexiones sftp



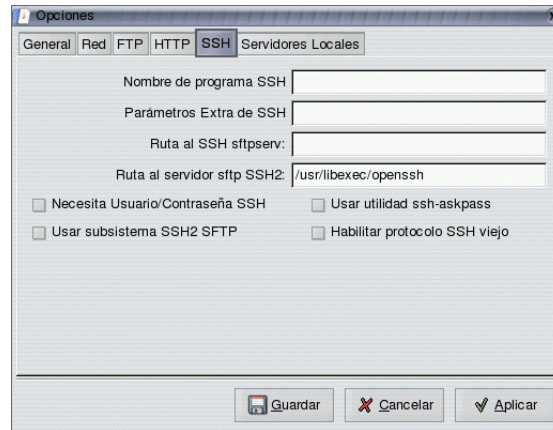
Antes de poder trabajar con él en modo **sftp** hemos de generar un par de ficheros que nos garantizan que la conexión es segura (véase la nota pie de página 64), para eso hemos de trabajar en modo terminal e iniciar una sesión **ssh** con el servidor con el que deseamos conectar para que se generen . Pero OJO, sólo es necesario hacerlo la primera vez que conectemos desde ese ordenador o si al ejecutar el programa comprobamos que no conecta.

Además, para que podamos trabajar con SFTP hemos de cambiar la configuración del programa, para eso pulsamos en el menú principal sobre **FTP** y en la ventana que aparece sobre **Opciones**

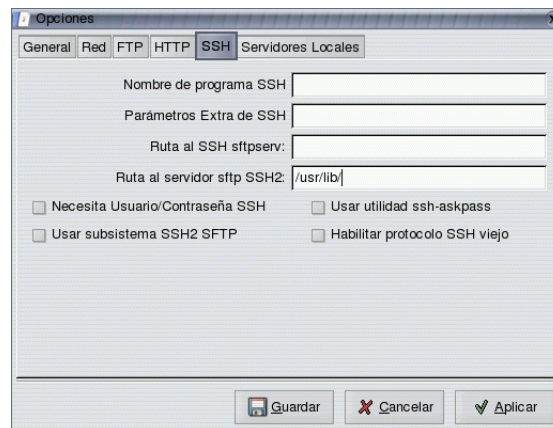


pestaña **SSH** y en el último campo escribimos:

/usr/libexec/openssh si nuestro servidor remoto es un Linux ejecutando la distribución Fedora o Red Hat



/usr/lib si la máquina servidora de ficheros es otra Guadalinex



Guardamos y Aplicamos y ya podemos conectar vía sftp.

Ejemplo Para realizar una conexión solo tenemos que rellenar los campos indicados: servidor, usuario y contraseña.

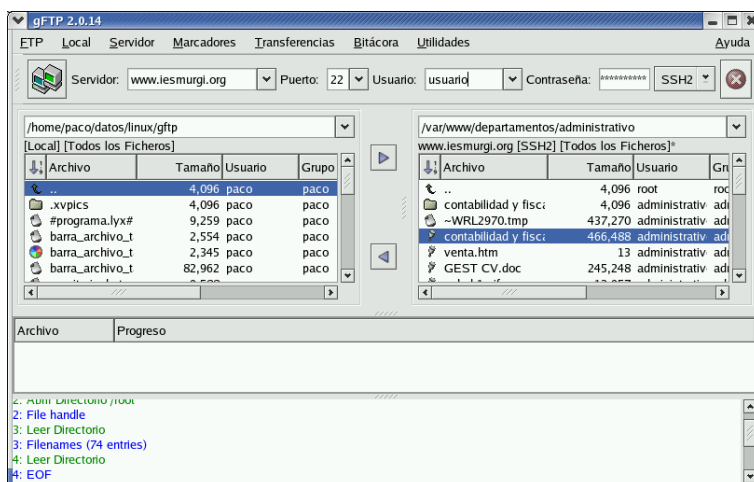
En nuestro caso serán

servidor: tux.midominio.org

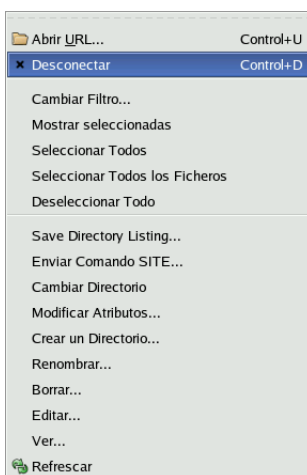
usuario: usuario

contraseña: xxxxxxxxxx

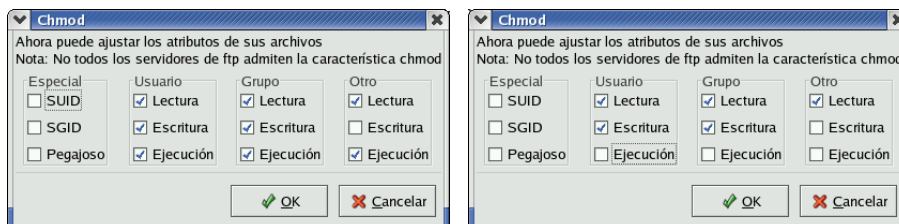
y optar en el menú desplegable que hay junto a **Contraseña** por conexión **SSH2**



Permisos Si iniciamos una conexión autenticada vía sftp o ftp, podemos modificar los permisos de un fichero o directorio de la máquina remota. Para eso sólo tenemos que marcarlo en vídeo inverso (situarnos sobre él) y tras pulsar sobre el botón derecho del ratón



optar por **Modificar Atributos**.




4.5. Samba

Usando Samba podemos compartir y utilizar recursos de sistemas de ficheros Linux e impresoras con sistemas Windows 3.11, 9x, NT, 2000, XP. Samba es rápido y sencillo de configurar. Linux¹⁵ con Samba puede trabajar como servidor y como cliente. Como servidor ofrece recursos (discos e impresoras) para que

¹⁵Por defecto, Guadalinex instala los paquetes samba y samba-common.

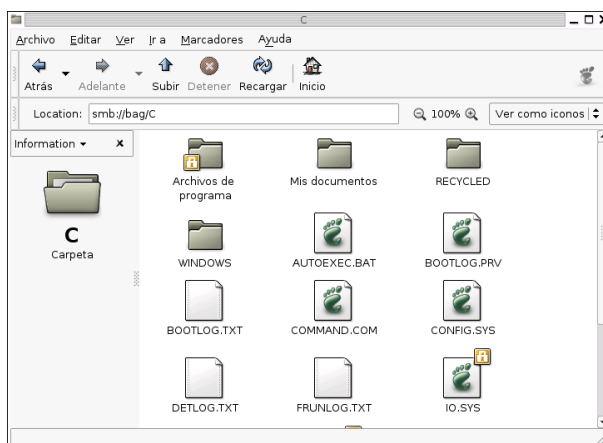
los utilicen las máquinas windows. Como cliente utiliza los servicios ofrecidos por las máquinas windows (discos e impresoras).

Para iniciar una conexión con una máquina en la que se ejecute Windows (u otro Linux como servidor

Samba) pulsaremos sobre el icono  del escritorio de Gnome. A continuación, y si el recurso al que queremos acceder de la red Windows está protegido con nombre de usuario y contraseña escribiremos los datos pedidos.



Como ejemplo de lo que podemos hacer vamos a prestar atención a la captura que sigue. Se trata de una máquina en la que se ejecuta Windows 98 con la unidad C: como recurso compartido y a la que hemos accedido desde Guadalinex.



4.6. Cajón “de-sastre”

Como indica el nombre de esta sección, vamos a comentar un par de aplicaciones que, si bien no son programas clientes de red, sí que nos pueden ayudar a solucionar algunos problemas con nuestra red. Dejamos en el cajón: `mtr`, `wget`¹⁶, `ngrep` para centrarnos en:

4.6.1. Gnome-netinfo

Es un programa muy interesante, se trata de interfaz gráfico de usuario para utilidades comunes de red y, si bien todo lo que se hace en modo gráfico se puede hacer en modo texto, dejaremos esa faceta para el curso de Linux como servidor.

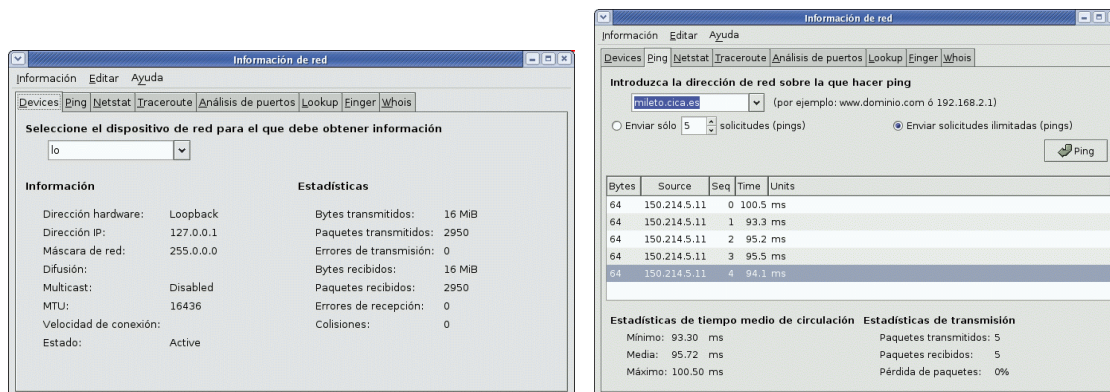
Podemos ejecutarlo con la cadena de menús **Aplicaciones**→**Internet**→**Gnome Network** o bien desde un xterm con:

```
$gnome-netinfo
```

¿Qué significado tiene cada pestaña?

¹⁶Seguro que sale más durante el curso.

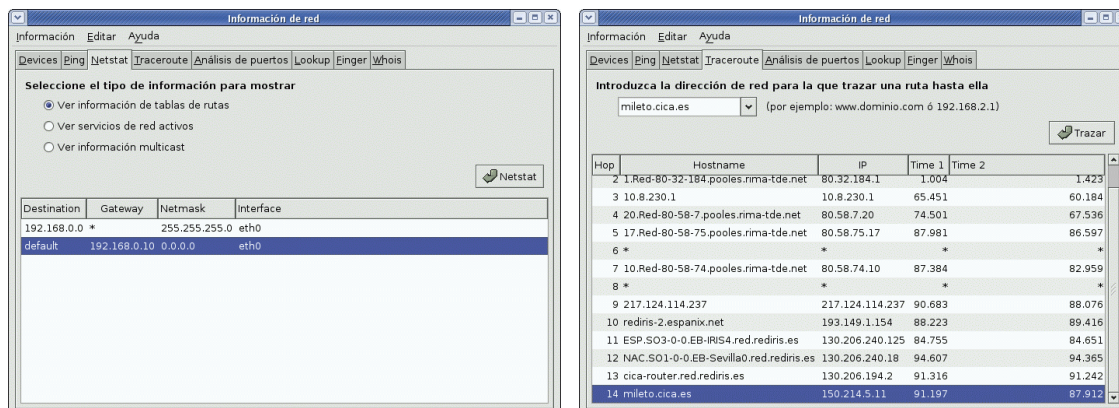
Devices desde aquí obtenemos información detallada sobre nuestra interfaz de red.



Ping El comando ping es una utilidad de red muy práctica. Trabaja de la siguiente forma:

Nuestra máquina envía paquetes de datos a la dirección de destino que, automáticamente, nos responde como si fuera un *boomerang*. Si todo es correcto, nos llegarán los paquetes de vuelta a nuestra máquina.

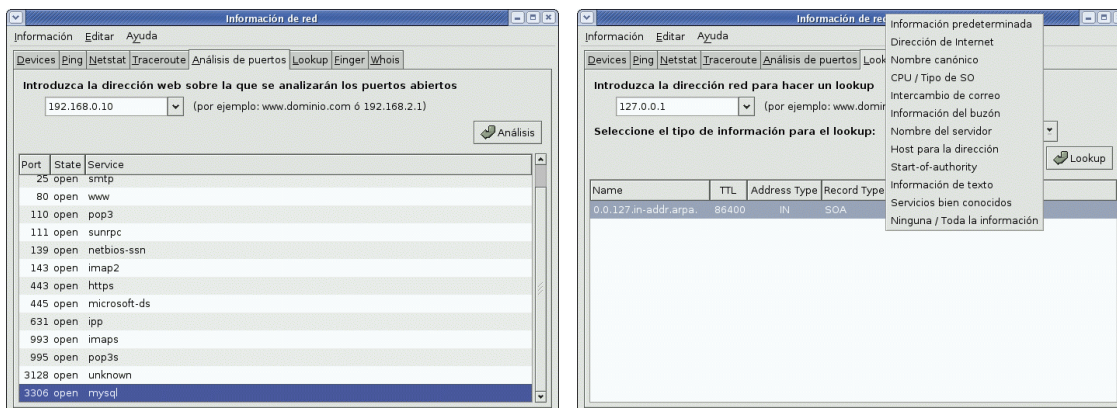
Netstat Muestra las conexiones de red, tabla de rutas, estadísticas de uso de la red, ...



Otra utilidad de diagnóstico es el comando:

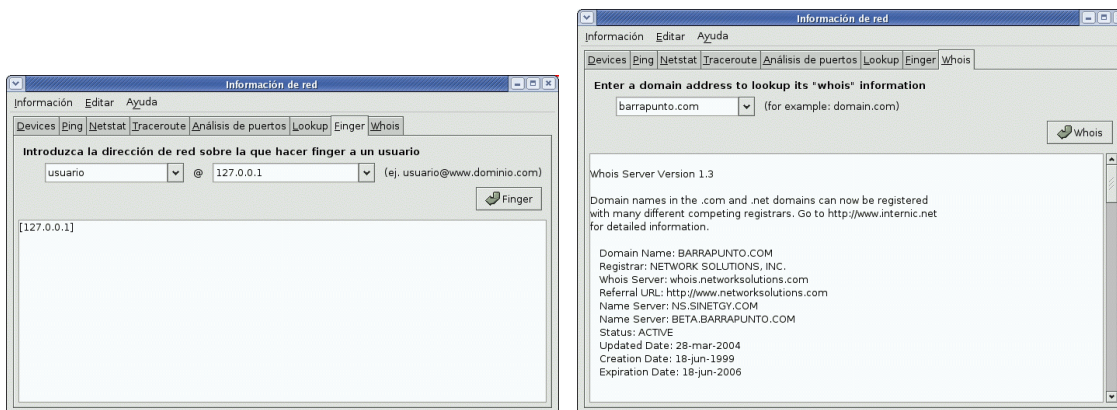
tracert que nos permite ver los sitios por los que van pasando los paquetes en el camino hasta su dirección de destino. Por ejemplo, con los datos de la captura nos indica las redes y routers por los que atraviesan los paquetes desde la máquina en la que se ha lanzado el comando hasta la máquina `miletto.cica.es`

Análisis de puertos se trata de un escáner de puertos, no debemos usarlo “contra” máquinas no conocidas. Su uso se debería restringir a comprobar la seguridad de nuestro sistema para testear qué puertos son los que tenemos abiertos.



LookUp para obtener el nombre de una máquina conocida su IP, para obtener la IP si conocemos su nombre, etc.

Finger permite mostrar (si el servicio está activo) información sobre los usuarios de un sistema (en este caso¹⁷ 127.0.0.1).



Whois permite obtener información sobre los dominios registrados: nombre, empresa que lo registró, etc.

4.6.2. ettercap

Se trata de una utilidad que nos permite capturar el tráfico que circula por una red (un *sniffer*).

Para ejecutarlo sólo tenemos que escribir¹⁸:

```
#ettercap
```

¹⁷Denominada de bucle local (*loopback*), es una dirección especial, que utiliza la propia máquina para acceder a sus procesos locales.

¹⁸Si nuestro terminal no es de al menos 25x80 caracteres dará error.

```

ettercap 0.6.b
SOURCE: 192.168.0.111 00:50:FC:A9:85:2F
DEST : 192.168.0.10 00:20:ED:7A:B6:68

3 hosts in this LAN (192.168.0.111 : 255.255.255.0)
1) 192.168.0.111 1) 192.168.0.111
2) 192.168.0.10 2) 192.168.0.10
3) 192.168.0.120 3) 192.168.0.120

Your IP: 192.168.0.111 MAC: 00:50:FC:A9:85:2F Iface: eth0 Link: SWITCH
Host: Unknown host (192.168.0.10) : 00:20:ED:7A:B6:68
Host: Unknown host (192.168.0.10) : 00:20:ED:7A:B6:68

```

Con la teclas de cursor podemos movernos entre las interfaz de red detectadas y con **Intro** optamos por el interfaz fuente y de destino. Si pulsamos sobre la tecla h obtendremos una pequeña ayuda para la ventana en curso.

```

ettercap 0.6.b
SOURCE: 192
DEST : 19
Help Window
[qQ][F10] - quit
[return] - select the IP
[space] - deselect the IPs
[tab] - switch between source and dest
[aA] - ARP poisoning based sniffing
        . for sniffing on switched LAN
        . for man-in-the-middle technique
[sS] - IP based sniffing
[mM] - MAC based sniffing
[jJ] - Only poisoning - no sniffing
[dD] - delete an entry from the list
[xX] - Packet Forge
[pP] - run a plugin
[ff] - OS fingerprint
[oO] - passive host identification
[cC] - check for other poisoner...
[rR] - refresh the list
[kK] - save host list to a file
[hH] - this help screen

Your IP: 1
Host: Unkno
Host: Unkno
nk: SWITCH

```

Para salir o volver atras se usa la tecla q. Si deseáis saber más sobre este programa se puede consultar: <http://cyruynet.com.ar/ettercap.htm>.