

/	el directorio raíz o <i>root</i> del sistema.	Estructura de directorios Unix (Debian)	sigue el estándar FHS (<i>Filesystem Hierarchy Standard</i>)
/boot	todo lo requerido por el kernel para iniciarse.		
/bin	ejecutables (bin aries) esenciales para todos los usuarios.		
/sbin	ejecutables especiales (superuser bin aries) esenciales para la administración del sistema.		
/lib	bibliotecas (lib raries) necesarias para ejecutar el kernel y programas de /bin y /sbin.		
/usr	(U nix s ystem r esources) programas no esenciales y otros archivos compartibles entre sistemas.		
├ /bin	aplicaciones de usuario instaladas por un administrador de paquetes (ej, con apt).		
├ /sbin	equivalente de /usr/bin para superusuarios.		
├ /lib	bibliotecas usadas por /usr/bin y /usr/sbin.		
├ /share	datos independientes de arquitectura (ej, manuales).		
├ /include	ficheros de cabecera .h de la biblioteca estándar de C (ej, stdio.h).		
├ /src	código fuente del kernel y de aplicaciones.		
└ /local	programas compilados manualmente por el superusuario (ej, con make).		
└─ /bin, /sbin, /lib, /share, /include, /src			
/opt	aplicaciones de terceros que necesitan su propio subdirectorio (ej, programas comerciales).		
/var	archivos variables (ej, logs del sistema, colas de impresión, colas de e-mail, etc).		
/etc	archivos estáticos de configuración de programas y del sistema		
├ /fstab	(f ile s ystems t able) montado y configuración de particiones realizado en el arranque.		
├ /mtab	(m ounted file systems t able) lista de sistemas de archivos montados actualmente.		
├ /passwd	información de usuarios (nombre, <i>UID</i> , <i>GID</i> , <i>login shell</i> , directorio <i>home</i> , etc).		
├ /shadow	contraseñas encriptadas de los usuarios.		
├ /group	información de grupos de usuarios (nombre, <i>GID</i> , y miembros).		
├ /gshadow	contraseñas encriptadas de los grupos.		
├ /sudoers	lista de usuarios con capacidad de ejecutar ciertos comandos privilegiados.		
├ /skel	directorio con los archivos por defecto de los directorios personales de los usuarios.		
├ /adduserconf	configuración por defecto de nuevos usuarios creados con adduser .		
├ /network/interfaces	configuración de interfaces, ya sea estática o dinámica (por DHCP).		
├ /resolv.conf	especifica el dominio y los servidores DNS a usar.		
├ /hosts	asociación de nombres de host a direcciones IP (como un DNS local).		
├ /hostname	nombre de host (debe ser uno de los nombres asociados a 127.0.0.1 en /hosts).		
├ /dhcp/dhcpd.conf	configuración de servidor DHCP (qué IP's se asignan a qué clientes).		
├ /sysctl.conf	configuración de tráfico IP (ej, IP <i>forwarding</i>).		
├ /apt/sources.list	servidores usados por apt para descargar paquetes.		
├ /apt/apt.conf.d	directorio con archivos de configuración de apt.		
└ /cron.hourly, daily, weekly, montly	directorios con scripts de ejecución periódica.		
/tmp	ficheros temp orales que se borran al reiniciar el sistema.		
/srv	datos de s ervicios proporcionados por el sistema, como FTP (<i>File Transfer Protocol</i>) o NFS.		
/home	contiene los directorios personales de los usuarios.		
/root	directorio personal del superusuario.		
/media	punto de montaje para medios removibles (USB, CDROM, etc).		
/mnt	punto de montaje para otros sistemas temporales (ej, un sistema de red).		
/dev	seudoficheros de acceso a dispositivos (d evice)s y pseudodispositivos (ej, /null, /random, etc).		
├ /null	fichero que descarta todos los datos que se le escriben.		
├ /sdxx, /hdxx	discos duros, particiones de discos, y volúmenes físicos de LVM.		
└ /random, /urandom	fichero con contenidos aleatorios de mayor o menor calidad.		
/sys	información de dispositivos (ej, nivel de batería, brillo de pantalla, etc).		
/proc	información del sistema.		
├ /cpuinfo	información de la CPU.		
├ /meminfo	información del uso de memoria RAM.		
└ /PID	directorio con información de un proceso con un <i>PID</i> dado.		
└─ /fd	directorio con descriptores de archivos (f ile d escriptors) abiertos.		
└─ /maps	mapa de memoria del proceso.		

Instalación de Software

- Mediante **dpkg** (*Debian Package*), que permite instalar, desinstalar y configurar paquetes en forma de archivos `.deb`, que contienen los ejecutables (junto con bibliotecas necesarios) y metadatos (scripts de configuración, listas de dependencias, etc) de la aplicación. Normalmente el nombre viene dado por `paquete_versión-build_arquitectura.deb` (ej, `gedit_3.38.1-1_amd64.deb`).

dpkg	<code>--install paq.deb</code>	Instala un paquete <code>.deb</code> . Usar la opción <code>--force-all</code> si hay algún error.
	<code>--remove paq</code>	Elimina un paquete <i>manteniendo</i> los archivos de configuración.
	<code>--purge paq</code>	Eliminación completa de un paquete, incluido archivos de configuración.
	<code>--list [patrón]</code>	Lista completa de paquetes instalados ¹ según un patrón dado.
	<code>--status paq</code>	Estado de un paquete.
	<code>--listfiles paq</code>	Lista ficheros de un paquete (archivos de configuración, manuales, etc).
	<code>--search fichero</code>	Identifica el paquete al que pertenece un fichero.
dpkg-reconfigure paq		Reconfigura un paquete.

- Con **apt** (*Advanced Packaging Tools*), que permite gestionar paquetes descargándolos desde fuentes locales o remotas, además de descargar automáticamente las dependencias necesarias. Las fuentes utilizadas se indican en el archivo `/etc/apt/sources.list`, mientras que la configuración general se puede cambiar en el archivo `/etc/apt/apt.conf` o el directorio `/etc/apt/` `apt.conf.d` (que contiene varios archivos de configuración).

apt-get	<code>update</code>	Actualiza la lista de paquetes (no actualiza los paquetes en sí).
	<code>upgrade</code>	Actualiza los paquetes (debe hacerse después de <code>apt update</code>).
	<code>install [-f] paq</code>	Instala un paquete y sus dependencias (<code>-f</code> para corregir errores).
	<code>remove paq</code>	Elimina un paquete <i>manteniendo</i> su configuración.
	<code>purge paq</code>	Eliminación completa de un paquete, incluido archivos de configuración.
	<code>autoremove</code>	Elimina paquetes que ya no sea necesarios.
	<code>clean</code>	Elimina copias de ficheros descargados.
apt-cache	<code>source paq</code>	Descarga ficheros fuente de un paquete. Compilarlos con <code>--compile</code> .
	<code>build-dep paq</code>	Instala las dependencias necesarias para compilar un paquete.
	<code>search [patrón]</code>	Busca paquetes para instalar de las fuentes (<code>sources.list</code>).
	<code>show paq</code>	Muestra información de cualquier paquete.
	<code>depends paq</code>	Muestra las dependencias de un paquete.
	<code>policy [paq]</code>	Muestra fuentes y prioridades (general o de un paquete).

`/etc/apt/sources.list`

```
# <uri> <dist> <components>
deb ftp://ftp.rediris.es/debian/ stable main contrib non-free

# Distribucion puede ser stable, testing, o unstable.
# Componentes puede ser main (conjunto principal de paquetes), contrib
# (paquetes adicionales) y non-free (no de código abierto).
```

Adicionalmente, **apt-get** utiliza un archivo candado `lock`, de forma que se crea el fichero al inicio de ejecución y se borra al terminar. Si ya existe un archivo `lock`, se asume que ya existe otra instancia de **apt-get** y se aborta la ejecución.

- Mediante la compilación manual del *código fuente del programa*, normalmente distribuidos en *tarballs*:
 - Archivo `.tar`, empaquetado sin comprimir.
 - Archivo `.tar.gz` o `.tgz`, empaquetado comprimido *gzip*.
 - Archivo `.tar.bz2` o `.tbz`, empaquetado comprimido *bzip2*.

El proceso de compilación suele explicarle en el fichero `INSTALL`, que contiene instrucciones de compilación, pero suele comenzar ejecutando el *script* `./configure`, que comprueba el entorno de instalación, bibliotecas necesarias, y genera los `Makefile`'s necesarios para la compilación.

<code>make [all]</code>	Compila el programa (utiliza <code>all</code> si no hay un <i>target</i> por defecto).
<code>make install</code>	Instala el programa una vez compilado
<code>ldd fichero</code>	Lista librerías enlazadas dinámicamente necesarias para un ejecutable.

Este proceso tiene el beneficio de que puede ser realizado por cualquier usuario, y no solo el superusuario (siempre que se el programa instale en un directorio con permisos de escritura).

Automatización de tareas

La realización automática de tareas se realiza fundamentalmente con `at` y `batch` (ejecuta una tarea en un momento dado) y `crontab` (ejecuta tareas periódicas).

<code>at tiempo</code>	Muestra un <i>prompt</i> donde podemos escribir comandos que se ejecutarán a la hora indicada (si el sistema está encendido). La salida estándar se envía como un <i>mail</i> al usuario.
<code>batch</code>	Como <code>at</code> , pero ejecuta trabajos con baja carga del sistema.
<code>atq</code>	<i>At Queue</i> lista los trabajos pendientes del usuario.
<code>atrm job</code>	Elimina un trabajo pendiente.
<code>crontab file</code>	Carga un fichero con comandos a ejecutar periódicamente.

`crontab file`

```
# Especificar shell usado y a quién se envía la salida estándar
SHELL=/bin/bash
MAILTO=pepe

# minuto hora día mes día_semana comando
30      4      *      *      1-5      rm -rf /tmp/* # Días laborales a las 4:30
12      08     10     06     *        /root/backup # 10 de Junio a las 8:12
0       15     *      *      1,3,5,7  rm -r /var/log # Cada dos días a las 15:00
```

Backups

Los *backups* los hay de 3 tipos: *completo* (se guarda todo el sistema de ficheros), *diferencial* (se guardan las diferencias respecto al último backup completo), e *incremental* (se guardan las diferencias respecto al último backup incremental).

<code>dump [-nivel] [ficheros]</code>	Hace copias de un sistema de ficheros (con todos sus ficheros y metadatos). Si el nivel es 0 se hace un <i>backup</i> completo; en caso contrario se salvan sólo ficheros nuevos o modificados desde el anterior backup de nivel inferior.
<code>restore acción [ficheros]</code>	Restaura ficheros guardados por <code>dump</code> .

También se pueden utilizar otros comandos para realizar backups más rudimentarios

<code>tar [opt..] archivo.tar.gz dir</code>	Comprime un directorio en un archivo tar (<i>Tape ARchiver</i>)
<code>dd if=/dev/sdxx of=file</code>	Copia una partición entera a un fichero.

Arranque del sistema

1. **BIOS** ejecuta un pequeño programa que detecta discos y ejecuta el gestor de arranque que se encuentre en el primer sector del disco (el MBR o *Master Boot Record*).

En caso de que el sistema sea compatible con UEFI se puede utilizar el estándar GPT (*GUID Partition Table*), que asigna un identificador único a cada partición y puede crear hasta 128 particiones primarias.

2. **Gestor de arranque**, usualmente GRUB (*G*rand *U*nified *B*ootloader), muestra un menú con todos los SO's del sistema, y carga el kernel del sistema elegido.
3. **initramfs** (Initial RAM File System) es una fase de preparación en el que se carga una imagen de disco a RAM, simulando un sistema de ficheros virtual con la que se carga el verdadero sistema de ficheros raíz.
4. **init** es el proceso de arranque que se ejecuta de `/sbin/init` una vez se carga el sistema de ficheros raíz.

En Debian este proceso `init` es `systemd`, que se encarga de ejecutar en paralelo procesos de creación del sistema (teclado, controladores, sistemas de ficheros, etc), y otros servicios (`networking`, `ssh`, `lvm`, etc).

`systemd` puede omitir el inicio de ciertos servicios en base a su *target*:

- (a) `rescue.target`: se inicia lo mínimo para intentar recuperar un sistema dañado.
- (b) `emergency.target`: un shell monousuario.
- (c) `multi-user.target`: multiusuario no gráfico.
- (d) `graphical.target`: multiusuario gráfico.

Los servicios específicos que se inician en el arranque del sistema se pueden configurar con:

```
systemctl [enable|disable] servicio
```

Además, se pueden tomar otras acciones con `systemctl` como `status`, para saber el estado actual de un servicio, o `start`, `stop` y `restart` para iniciar o detener servicios.

Procesos

Un proceso es una instancia de un programa en ejecución, ya sean procesos lanzados por el usuario, el propio kernel, o servicios del sistema (*daemons*).

top	Lista de procesos actualizada periódicamente, por defecto ordenados por uso de CPU (nota: el 100% de la CPU se mide en base a un sólo núcleo).
uptime	Muestra la carga media del sistema de los últimos 1, 5 y 15 minutos.
w	Como uptime , pero también muestra información de los usuarios y sus procesos.
free	Muestra la cantidad de memoria libre y usada del sistema.
ps [-o col...]	Process Status muestra los procesos lanzados desde la terminal con un determinado formato de columnas (PID, estado, %CPU, etc).
ps -u user	Igual que el anterior, para todos los procesos lanzados por un usuario.
ps -e	Igual que el anterior, pero para todos los procesos.
pstree	Muestra el árbol de procesos.
pgrep nombre	Busca un proceso por nombre (similar a realizar ps grep nombre).
strace cmd	Muestra las llamadas al sistema realizadas por un comando cmd .

\$uptime	Tiempo encendido	Carga media del último minuto	Carga media de los últimos 15 minutos
20:25:03	up 25 days, 11:12	load average: 3.00,	3.07, 3.08
Hora actual	13 users	Carga media de los últimos 5 minutos	
	Usuarios conectados		

```
$ ps -o user,pid,state,comm
USER PID S COMMAND
amo 1357 S bash
amo 1992 S gedit
amo 2279 R ps
```

State	significado
R	<i>Running</i> (ejecutándose o en la cola de ejecución).
S	<i>interruptible Sleep</i> (esperando por un evento).
D	<i>uninterruptible sleep</i> (esperando por IO).
T	<i>sTopped</i> (detenido por una señal STOP o TSTP).
Z	<i>Zombie</i> (terminado, pero no liberado por su padre).

Una terminal puede ejecutar un único comando en primer plano *foreground*, es decir, se espera a que termine el comando antes de pedir otro. Pero se pueden ejecutar multiples procesos en segundo plano (*background*).

jobs	Lista comandos en <i>background</i> .
fg job	Mueve un trabajo al <i>foreground</i>
bg job	Reanuda un trabajo en el <i>background</i>
cmd &	Ejecuta un comando en el <i>background</i>

A los procesos se le pueden enviar *señales*, que pueden ser ignoradas o capturadas (para ejecutar algún código), excepto SIGKILL y SIGSTOP, que incondicionalmente matan y detienen el proceso.

kill -l	Lista de señales.
kill -x PID kill -(SIG)SEÑAL PID	Manda una señal (en formato numérico o simbólico) a un proceso.
kill -x name kill -(SIG)SEÑAL name	Como el anterior, pero envía las señales a procesos identificados por un nombre.
nohup cmd	Crea un proceso que ignora la señal SIGHUP (no se muere el proceso al cerrar la terminal).

SIGTERM	Mata el proceso, permitiéndole terminar correctamente.
SIGKILL	Mata el proceso inmediatamente (<i>no</i> se puede ignorar).
SIGINT	Interrupción del teclado (Ctrl+C).
SIGSTOP	Pausa temporalmente el proceso (<i>no</i> se puede ignorar).
SIGTSTP	Stop del teclado (Ctrl+Z).
SIGCONT	Continúa el proceso pausado.
SIGHUP	Enviada si se cierra la terminal o el proceso controlador.

nice -n x cmd	Lanza un proceso con una cierta prioridad, desde -20 (prioridad más alta), hasta 19 (prioridad más baja). Por defecto es 0.
renice x [-p PID]	Cambia la prioridad de un proceso. Sólo el superusuario puede poner una prioridad más alta; el resto sólo pueden bajarla.
ulimit [limites..]	Limita los recursos a los que tienen acceso los procesos creados por la terminal (tiempo de CPU, número de ficheros, etc).

Ficheros

En Linux se consideran ficheros todo tipo de objetos que pueden ser operados de forma similar a un fichero (ficheros regulares, directorios, ficheros de dispositivo, etc). Todos cumplen que tienen un nombre de hasta 255 caracteres (preferentemente sin `*`, `$`, `?`, `'`, `"`, `/` ni `\.`) y una serie de atributos (accesibles con `ls -l`):

Permisos de usuario	Permisos de otros	Tamaño en bytes				
-	<u>rw-rw-r--</u>	1	usuario_propietario	grupo	731	2005-09-21 20:59 nombre_fichero
	Permisos de grupo	Número de enlaces duros al archivo			Fecha de última modificación	
-	Tipo de fichero					
-	Fichero regular					
d	Directorio					
l	Enlace simbólico (<i>link</i>)					
c	Fichero de dispositivo de caracteres					
b	Fichero de dispositivo de bloques					
p	Tubería (<i>pipe</i>)					
s	Socket de red					

Linux guarda para cada fichero 3 tipos de fechas, que se pueden mostrar con `ls -l` con distintas opciones:

- `mtime`, *modification time*, es la opción por defecto.
- `atime`, *last access*, se muestra con `--time=atime`.
- `ctime`, *status change*, se muestra con `--time=ctime`.

Cada fichero tiene un triplete de permisos, unos que se aplican al usuario dueño del archivo (aunque sean los más restrictivo), otros a los usuarios del grupo, y otros para los demás usuarios. Cada grupo de permisos se puede representar simbólicamente (`rw`) o numéricamente (con un dígito octal, suma de los permisos individuales).

Permiso	Oct.	Fichero	Directorio
Ejecución	x	1	Ejecutar
Escritura	w	2	Modificar
Lectura	r	4	Abrir/leer

-	<u>rw</u> x	<u>rw</u> -	<u>r</u> --	-	<u>r</u> -x	<u>r</u> --	--
	7	6	4		5	4	0
	764				540		

Existen, además, otros 3 permisos especiales que modifican ligeramente la representación simbólica, y que se representan numéricamente con un nuevo dígito octal a la izquierda.

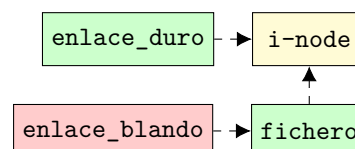
Permiso	Oct.	Fichero/Directorio
<i>Sticky bit</i>	T	1
<code>setgid</code>	S	2
<code>setuid</code>	S	4

d	<u>rw</u> x	<u>rw</u> x	-- <u>T</u>	d	<u>rw</u> x	<u>rw</u> x	<u>rw</u> T	-	<u>r</u> --	<u>r</u> - <u>S</u>	<u>r</u> --	-	<u>r</u> - <u>S</u>	<u>r</u> --	<u>r</u> --	-	<u>r</u> - <u>S</u>	<u>r</u> -x	<u>r</u> -x
	7	7	0		7	7	7		4	4	4		4	4	4		5	5	5
	1770				1777				2444				4444				4555		

chmod [-R]	<i>wxyz file</i>	Establece los permisos en formato numérico de un archivo.
	[+ =] <i>permisos file</i>	Añade (+), quita (-) o establece (=) <i>permisos</i> en formato simbólico.
	<i>u</i> [+ =] <i>permisos file</i>	Igual que lo anterior para permisos del usuario propietario.
	<i>g</i> [+ =] <i>permisos file</i>	Igual que lo anterior para los permisos del grupo.
	<i>o</i> [+ =] <i>permisos file</i>	Igual que lo anterior para permisos de otros usuarios.

Un tipo especial de fichero son los *enlaces*, que permiten referirse a un fichero o directorio desde otro lugar. Los hay de dos tipos:

- **Duros**, que hacen referencia al mismo *nodo-i* que el archivo original. Esto hace que borrar el archivo original no afecte al enlace, pero solo funciona para enlaces a archivos (no directorios) de una misma partición.
- **Blandos** o *simbólicos*, que simplemente guardan el *path* del fichero enlazado. Borrar el archivo original implica invalidar el enlace.



<code>touch file</code>	Crea un nuevo fichero vacío o actualiza el <code>mtime</code> y <code>atime</code> al tiempo actual.
<code>mkdir dir</code>	Crea un directorio vacío <code>dir</code> .
<code>mknod ...</code>	Crea ficheros de dispositivo o tuberías con nombre.
<code>ln [-s] src link</code>	Crea un enlace duro (o simbólico, con la opción <code>-s</code>) al fichero <code>src</code> .
<code>mv src dest</code>	Mueve un fichero o directorio <code>src</code> a <code>dest</code> .
<code>cp [-r] src dest</code>	Copia un fichero a otro lugar (<code>-r</code> para copiar directorios recursivamente).
<code>rm [-r] file</code>	Elimina un fichero (<code>-r</code> para eliminar directorios recursivamente).
<code>ls -l [dir]</code>	Muestra los atributos de los archivos de un directorio.
<code>file file</code>	Muestra el tipo de un fichero (si es un directorio, enlace, PDF, PNG, etc).
<code>chown own file</code>	Cambia el propietario de un fichero (sólo para superusuarios).
<code>chgrp grp file</code>	Cambia el grupo del fichero a otro al que pertenezca el propietario.

find [dir]	<code>expr1 expr2</code> <code>expr1 -o expr2</code> <code>! expr</code> <code>\(expr \)</code>	Busca archivos que cumplan ambas expresiones. Busca archivos que cumplan algunas de las expresiones. Busca todos los archivos que no cumplan la expresión. Busca todos los archivos que cumplan la expresión agrupada.
	<code>-name patron</code> <code>-iname patron</code> <code>-regex patron</code> <code>-type tipo</code> <code>-user usuario</code> <code>-group grupo</code> <code>-size n [ckMG]</code> <code>-size -n [ckMG]</code> <code>-size +n [ckMG]</code> <code>-perm permisos</code> <code>-perm -permisos</code>	Ficheros que coincidan con el patrón (se puede usar <code>"*"</code>). Como <code>-name</code> , pero sin distinguir mayúsculas y minúsculas. Como <code>-name</code> , pero con expresiones regulares. Ficheros de un tipo dado (<code>-</code> , <code>d</code> , <code>l</code> , <code>c</code> , <code>b</code> , <code>p</code> , <code>s</code>). Ficheros con un propietario dado. Ficheros de un grupo dado. Ficheros con un tamaño exacto (en bytes, kbytes, etc). Ficheros con un tamaño menor que <code>n</code> . Ficheros con un tamaño mayor que <code>n</code> . Ficheros con unos permisos exactos. Ficheros con unos permisos iguales o menos restrictivos.
<code>which [opt..] cmd</code>		Muestra la localización de un comando buscando en <code>\$PATH</code> .
<code>whereis [opt..] cmd</code>		Localización de un comando, fuente, y página del manual.
<code>locate fichero</code>		Localiza rápidamente cualquier tipo de fichero.

De forma general a los comandos de manipulación de ficheros se le puede especificar como argumento el nombre del archivo que se quiere manipular, aunque todos permiten utilizar la entrada estándar como contenido. Por ejemplo, `sort file.txt | uniq | wc` cuenta el número de líneas únicas de un fichero.

<code>wc [opt..] [file]</code>	Cuenta el número de líneas, palabras, y bytes de un fichero.
<code>cat [file]</code>	Simplemente imprime el fichero (su uso suele ser innecesario).
<code>tac [file]</code>	Imprime las líneas del fichero en orden inverso.
<code>rev [file]</code>	Imprime todo el fichero en orden inverso.
<code>head [-n l] [file]</code>	Muestra las primeras líneas de un fichero.
<code>tail [-n l] [file]</code>	Muestra las últimas líneas de un fichero.
<code>sort [file]</code>	Ordena en orden alfabético las líneas de un fichero.
<code>uniq [file]</code>	Elimina líneas duplicadas sucesivas (se suele usar con <code>sort</code>).
<code>grep cadena [file]</code>	Muestra las líneas que contienen una cierta cadena.
<code>egrep exp [file]</code>	Muestra las líneas que contienen una cierta expresión regular.
<code>sed -r "s/X/Y/g" [file]</code>	Busca y reemplaza expresiones regulares en un archivo.

<code>tr [opt..] set1 set2 [file]</code> <i>Translate</i> sustituye caracteres de un conjunto a otro. Por ejemplo, <code>tr 'a-z' 'A-Z'</code> cambia de minúsculas a mayúsculas.
<code>cut [opt..] [file]</code> Extrae una parte de cada línea del archivo. Por ejemplo, <code>cut -c 1-7</code> obtiene los 7 primeros caracteres de cada línea.
<code>paste [opt..] file [file..]</code> Une las líneas de cada uno de los ficheros.
<code>join [opt..] file1 file2</code> Une líneas de dos ficheros en base a campos en común.
<code>split [opt..] [file] prefix</code> Separa un fichero en varios (que comienzan por <code>prefix</code>).

Expresiones Regulares Extendidas

.	Cualquier carácter.
[<i>abc</i>]	Cualquiera de los caracteres entre corchetes.
[<i>a-z</i>]	Cualquiera de los caracteres en el rango.
[^ <i>abc</i>]	Cualquier carácter que no esté entre corchetes.
^	El principio de la línea.
\$	El final de la línea.
<i>exp</i> *	0 o más ocurrencias de la expresión regular <i>exp</i> .
<i>exp</i> +	1 o más ocurrencias de la expresión regular <i>exp</i> .
<i>exp</i> ?	0 o 1 ocurrencia de la expresión regular <i>exp</i> .
<i>exp</i> { <i>n</i> }	Exactamente <i>n</i> ocurrencias de la expresión regular <i>exp</i> .
<i>exp</i> { <i>n</i> ,}	Al menos <i>n</i> ocurrencias de la expresión regular <i>exp</i> .
<i>exp</i> { <i>n</i> , <i>m</i> }	Entre <i>n</i> y <i>m</i> ocurrencias de la expresión regular <i>exp</i> .
<i>exp</i> <i>exp</i>	Una expresión regular o la otra.
(<i>exp</i>)	Captura una expresión que puede ser referenciada después.
\1, \2..	La primera, segunda, etc expresión capturada anteriormente con (<i>exp</i>).
\., \\. .	Escapa un metacaracter (., \$, \, [,], (,), etc).

Las expresiones regulares se pueden utilizar con multitud de comandos (**grep -E**, **egrep**, **find -regex**, etc), pero notablemente **sed -r**, que permite buscar y reemplazar coincidencias en un archivo o en entrada estándar:

```
$ echo "Hola x!" | sed -r "s/Hola/Adiós/g"
Adiós x!
$ echo "test.mp3" | sed -r "s/(.*)\.mp3/\1.mp4/g"
test.mp4
```

Al usar comillas dobles " (y no comillas simples ') se pueden usar comandos y variables en las expresiones:

```
$ echo "fecha" | sed -r "s/fecha/`date`/g"
mar 03 ene 2023 10:41:16 CET
```

Siempre se reconoce la cadena más larga posible:

```
$ echo "holaaaaa" | sed -r "s/a+/a/g"
hola
```

Adicionalmente, se puede usar & para hacer referencia a toda la cadena reconocida:

```
$ echo "x=5" | sed -r "s/[0-9]+/[Número &]/g"
x=[Número 5]
```

AWK

AWK es un programa avanzado de procesamiento de textos que permite ejecutar código arbitrario por cada línea de un archivo. También nos deja acceder fácilmente a distintos *campos* de cada línea (por defecto separados por espacios en blanco).

BEGIN { [op;]* }	Ejecuta varias operaciones en el inicio del programa.
{ [op;]* }	Ejecuta varias operaciones en cada línea del programa.
/exp/ { [op;]* }	Ejecuta varias operaciones en las líneas que verifiquen la expresión regular.
!/exp/ { [op;]* }	Ejecuta varias operaciones en las líneas que <i>no</i> verifiquen la expresión regular.
END { [op;]* }	Ejecuta varias operaciones al final del programa.
print \$1, \$2...	Imprime el primer, segundo, etc campo de una línea.

```
$ls -ldh * | awk -f programa
Ocupación:
fich1 ocupa 36 bytes
fich2 ocupa 9,1K bytes
fich3 ocupa 3,7M bytes
Final
```

programa

```
BEGIN {print "Ocupación:"}
{print $9, "ocupa ", $5, "bytes"}
END {print "Final"}
```

Particiones y Sistemas de Ficheros

Los discos duros pueden ser leídos, escritos e identificados mediante su pseudofichero de dispositivo del directorio `/dev`, o mediante su *identificador único universal* (UUID), con la ventaja de que el UUID no depende del orden en el que se inicializan los discos (el primer disco es `/dev/sda`, el segundo `/dev/sdb`, etc).

Los hay de distintos tipos:

- Serial ATA (o SATA), de los más comunes, identificados por `/dev/sda`, `/dev/sdb`, etc.
- SCSI, formado por distintos dispositivos (cintas, CD-ROMS, escáneres, etc) conectados en cadena. Son usados por servidores de altas prestaciones y se identifican de forma similar a los SATA.
- IDE o Parallel ATA, poco usados en la actualidad, e identificado con `/dev/hda` y `/dev/hdb` para el controlador primario (maestro y esclavo, respectivamente) y `/dev/hdc` y `/dev/hdd` para el controlador secundario (maestro y esclavo).

Cada disco puede dividirse en unidades lógicas llamadas *particiones*, cuyo nombre de pseudofichero se consigue añadiendo un número al disco al que pertenecen (ej, un disco `/dev/sda` tendrá particiones `/dev/sda1`, `/dev/sda2`, etc).

<code>fdisk dispositivo</code>	Fixed Disk crea, modifica o lista particiones de un dispositivo físico.
<code>blkid dispositivo</code>	Block Identification muestra el UUID de un dispositivo físico o partición.

Sobre una una partición se puede montar un *sistema de ficheros*, que permite guardar información estructurada en forma de archivos y directorios. Los hay de distintos tipos:

- `ext4`, *Fourth Extended filesystem*, es el sistema de ficheros estándar de Linux, permitiendo trabajar con discos y ficheros de gran tamaño, y con *journaling* (se registran los cambios realizados en un historial para así evitar la corrupción de datos).
- `ext3` y `ext2`, los predecesores de `ext4`.
- `fat16` y `fat32`, usados en MS-DOS y versiones antiguas de Windows.
- `ntfs` y `ReFS`, usados en versiones modernas de Windows.
- `swap`, sistema de ficheros especial que actúa como RAM adicional y permite guardar la RAM al hibernar.

Cada sistema de ficheros se puede asociar o *montar* sobre un directorio, ya sea mediante el comando `mount`, cuyos efectos se pierden al reiniciar, o modificando el archivo `/etc/fstab`, que contiene líneas indicando qué sistemas de archivos se montan en qué directorios en el arranque. Los sistemas de ficheros montados actualmente se encuentran en el archivo `/etc/mtab`.

<code>mkfs.tipo part</code>	Make File System crea un sistema de ficheros de un <i>tipo</i> dado (<code>ext2</code> , <code>ext3</code> , <code>ext4</code> , <code>msdos</code> , <code>ntfs</code> , etc) en la partición indicada.
<code>mkswap part</code>	Crea un sistema de ficheros de intercambio o <i>swap</i> en la partición.
<code>fsck.tipo part</code>	File System Consistency Check testea y repara sistemas de ficheros.
<code>mount dir</code>	Monta sobre un directorio dado la partición indicada por <code>/fstab</code> .
<code>mount part dir</code>	Monta sobre un directorio una partición explícita.
<code>umount part/dir</code>	Desmonta una partición explícita o la partición asociada a un directorio dado.
<code>swapon part</code>	Activa una partición <i>swap</i> .
<code>du -hs dir</code>	Disk Usage muestra el espacio de disco utilizado por un directorio.
<code>df -h</code>	Disk Free muestra los sistemas de ficheros montados y su espacio disponible.

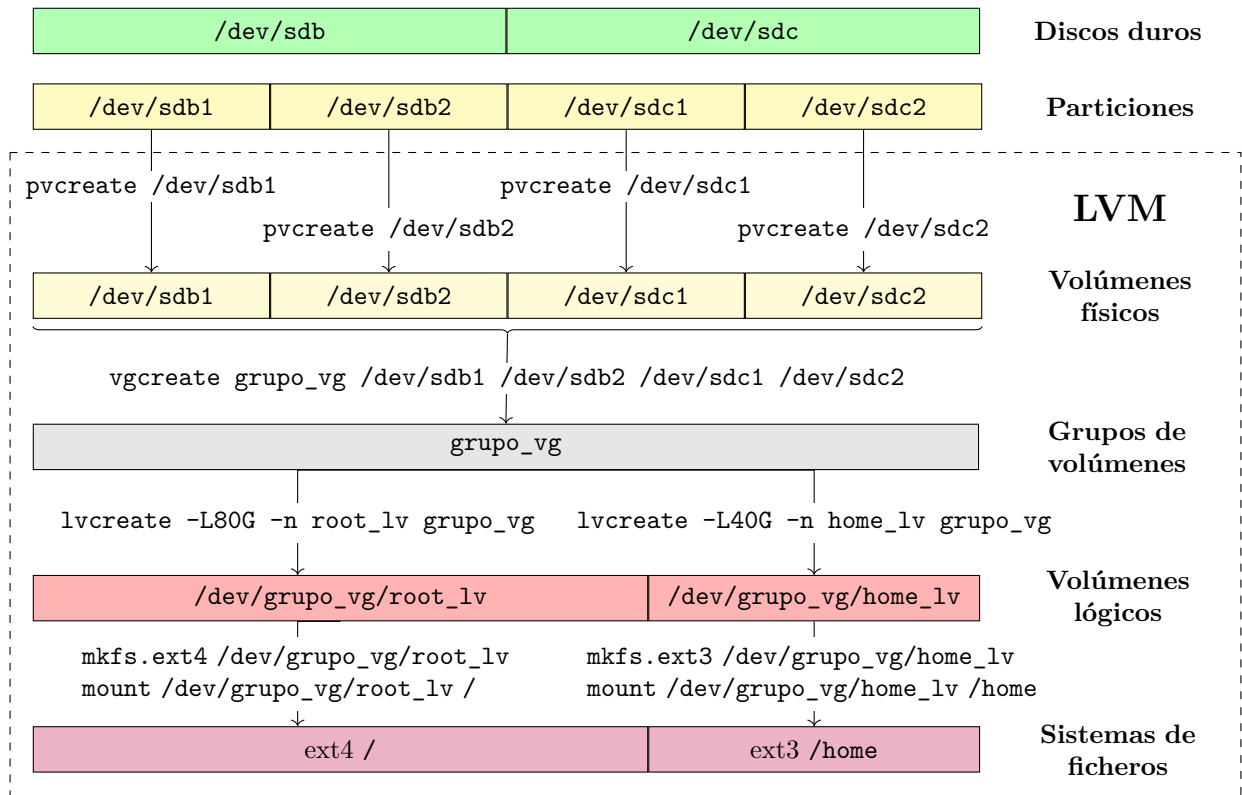
/etc/fstab

```
# <file system> <mount point> <type> <options> <dump> <pass>
/dev/sda1      /                auto          defaults      50           1
/dev/sda9      /home           ext4          auto,user,rw  0            2
/dev/sdb2      /tmp            ext3          defaults      0            3
UUID=b0f7f038-c762-40f4-aa9b-c718193e1db0 none swap sw 0 0

# Opciones:
# * ro/rw:      read-only o read-write.
# * auto/noauto: montar o no montar automáticamente en el arranque
# * user/nouser: puede ser montado por cualquier usuario o sólo por
#                  el superusuario.
# * defaults:   opciones por defecto (rw, auto, nouser).
```

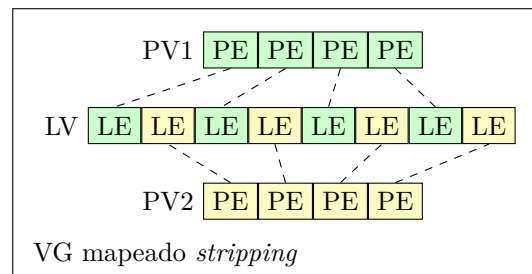

Gestor de volúmenes LVM

Con LVM, *Logical Volume Manager*, en vez de trabajar sobre particiones, los sistemas de ficheros se montan sobre *volúmenes lógicos*, que pueden ser movidos, cambiados de tamaño fácilmente (incluso *en caliente*, esto es, según el espacio que sea necesario), y *encriptados*. Esta flexibilidad también permite distribuir el espacio adicional de nuevos dispositivos de almacenamiento entre varios volúmenes según se necesite.



Cada volumen físico se divide en unidades básicas denominadas *extensiones físicas* (*Physical Extensions*), que se asignan a *extensiones lógicas* (*Logical Extensions*) de distintos volúmenes lógicos de su grupo de volúmenes.

La forma en la que se hace esta asignación depende del mapeado usado (*stripping*, *lineal*, *mirroring*).



<code>pvcreeate pv</code>	Physical Volume muestra información de un volumen físico.
<code>vgdisplay vg</code>	Volume Group muestra información de un grupo de volúmenes.
<code>lvdisplay lv</code>	Logical Volume muestra información de un volumen lógico.
<code>pvcreeate part.</code>	Crea un volumen físico sobre una partición dada.
<code>vgcreate vg pv...</code>	Crea un grupo de volúmenes a partir de varios volúmenes físicos.
<code>vgextend vg pv</code>	Añade un nuevo volumen físico a un grupo de volúmenes.
<code>vgreduce vg pv</code>	Elimina un volumen físico de un grupo de volúmenes.
<code>vgremove vg</code>	Elimina un grupo de volúmenes.
<code>lvcreate -n lv vg</code>	Crea un volumen lógico (especificando el tamaño en bytes con <code>-Lx</code> o en extensiones lógicas con <code>-lx</code>) a partir de un grupo de volúmenes.
<code>lvextend -L+x lv</code>	Amplía ² un volumen lógico en x bytes (o en extensiones lógicas con <code>-l+x</code>).
<code>lvreduce -L-x lv</code>	Reduce el tamaño de un volumen lógico en x bytes.
<code>lvremove lv</code>	Elimina un volumen lógico (que no puede estar montado en ese momento).
<code>fsadm resize lv</code>	File System Administrator redimensiona el sistema de ficheros para que sea tan grande como la partición a la que pertenece.

¹Nótese que, realmente, tras ampliar el tamaño de un volumen lógico o partición, mientras no se amplíe el tamaño del sistema de ficheros con `fsadm` comandos como `df` y demás programas seguirán viendo un sistema de ficheros con el tamaño original.

Usuarios

Todo usuario de un sistema UNIX debe tener una cuenta para poder acceder al sistema, que define quién es el usuario y qué puede hacer en el sistema. Todas las propiedades de las cuentas de usuario se encuentran en 4 ficheros:

/etc/passwd

```
root:x:0:0:root:/root:/bin/bash
pepe:x:1002:1002:Pepe Perez,dept.EC,desp.1:/home/pepe:/bin/bash
| | | | |
| | | | GECOS (Información varia) | Shell
| | | | GID (Código del grupo principal) | interactivo
| | | | UID (Identificador de usuario) |
| Contraseña encriptada | Directorio
Usuario | personal
```

Si en `/etc/passwd` la contraseña se indica como `x`, entonces la verdadera contraseña encriptada y otras propiedades (último cambio de contraseña, tiempo de validez, etc) se encuentran en el archivo protegido:

/etc/shadow

```
pepe:QKDPc5E$SWlkjRWexrXYgc98F:12825:0:90:5:30:13096:
| ~~~~~
Usuario Contraseña
```

Existe también un equivalente de los anteriores archivos para grupos de usuarios, aunque a los grupos no se le suele poner contraseña, así que `/etc/gshadow` normalmente sólo tiene una copia de los miembros del grupo.

/etc/group

```
users:x:100:pepe,elena
| | ~~~~~
Nombre GID Miembros
```

/etc/gshadow

```
users:::pepe,elena
|
Sin contraseña (lo común)
```

<code>passwd [opt..] [user]</code>	Cambia la contraseña de un usuario y sus propiedades
<code>gpasswd [opt..] group</code>	Gestiona miembros y contraseñas de un grupo.
<code>chage [opt..] [user]</code>	Change age permite ver y cambiar información de expiración de la contraseña.
<code>su [opt..] [-] [user]</code>	Permite cambiar a otro usuario (o si no se especifica <code>user</code> , al administrador).
<code>newgrp [-] [group]</code>	Permite cambiar a otro grupo del que se sea miembro.
<code>sudo [-u user] cmd</code>	Permite ejecutar un comando <code>cmd</code> con la identidad de otro usuario. Si no se especifica <code>user</code> , se ejecuta como <code>root</code> sólo si el usuario pertenece al grupo <code>sudo</code> o si en el fichero <code>/etc/sudoers</code> se le da permiso al usuario para ejecutar el comando.

```
$ chage -l debian
Último cambio de contraseña : ago 09, 2016
La contraseña caduca : nunca
Contraseña inactiva : nunca
La cuenta caduca : nunca
Número de días mínimo entre cambio de contraseña : 0
Número de días máximo entre cambio de contraseña : 99999
Número de días de aviso antes de que caduque contraseña : 7
```

Ejemplos de algunos grupos de usuarios que suelen estar presentes en un sistema:

<code>users</code>	Los usuarios normales.
<code>sudo</code>	Usuarios que pueden ejecutar cualquier comando.
<code>http</code>	Pueden gestionar los servicios web.
<code>games</code>	Pueden gestionar los juegos.
...	

Adicionalmente, cuando se crea un nuevo usuario por defecto se le da su propio grupo.

Creación de Usuarios

- Creación *manual* de las cuentas de usuario, siguiendo los siguientes pasos:
 1. Insertar las líneas pertinentes en `/etc/passwd` y `/etc/group` utilizando el comando `vipw`, que permite guardar los ficheros sólo si tienen el formato correcto (evita su corrupción accidental).
 2. Con `mkpasswd` generar contraseñas encriptadas, e introducirlas en `/etc/shadow` y `/etc/gshadow` (si fuese necesario una contraseña de grupo) utilizando `vipw -s`.
 3. Crear el directorio personal de usuario, copiando sus contenidos de `etc/skel`.
 4. Cambiar los permisos del directorio personal con `chown`, `chgrp` y `chmod`.

Una vez hecho esto, el usuario debería cambiar su contraseña lo antes posible.

<code>vipw [-s]</code>	Edita de forma segura <code>passwd</code> y <code>group</code> (y <code>shadow</code> y <code>gshadow</code> con <code>-s</code>).
<code>mkpasswd password</code>	Obtiene la versión cifrada de una contraseña.

- Usar *comandos de manejo de cuentas*, que facilitan la modificación de los archivos `/etc/passwd`, `/etc/shadow`, `/etc/group`, y `/etc/gshadow` de forma segura.

<code>useradd</code> , <code>groupadd</code> <code>userdel</code> , <code>groupdel</code> <code>usermod</code> , <code>groupmod</code>	Comandos de bajo nivel para añadir, modificar y eliminar usuarios y grupos. Por defecto no se crean los directorios de usuario ni contraseñas (cuentas deshabilitadas).
<code>adduser</code> <code>addgroup</code>	<i>Frontend</i> de los comandos de bajo nivel, que crean usuarios y grupos según la configuración de <code>/etc/adduser.conf</code> .
<code>newusers</code> <code>chpasswd</code>	Permite crear varias cuentas simultáneamente. Permite actualizar varias contraseñas simultáneamente

Módulos de autenticación

PAM (*Pluggable Authentication Module*) es una biblioteca de autenticación que cualquier aplicación puede utilizar para validar usuarios, utilizando por debajo múltiples esquemas de autenticación alternativos (ficheros locales, claves de un solo uso, DNI electrónico, Kerberos, LDAP, etc.). PAM utiliza módulos de varios tipos:

- Módulos de autenticación (**auth**): para la identificación del usuario (por ejemplo, contraseña, tarjeta de identificación, características biométricas, etc).
- Módulos de cuentas (**account**): controlan las condiciones para que la autenticación sea permitida (por ejemplo, que la cuenta no haya caducado, que el usuario tenga permiso para iniciar sesiones a esa hora del día, etc.).
- Módulos de contraseña (**password**): condiciones y procedimientos para el cambio de contraseñas.
- Módulos de sesión (**session**): configuran y administran sesiones de usuarios (tareas adicionales que son necesitadas para permitir acceso, como el montaje de directorios, etc.)

Cuotas de usuario

Editando el archivo `/etc/fstab` se pueden imponer límites al uso del disco de los usuarios y grupos a algunos *filesystem*'s, evitando así que unos usuarios monopolicen el disco.

<code>quotacheck</code>	Construye el índice y testea la integridad.
<code>quotaon</code>	Activa las cuotas.
<code>quotaoff</code>	Desactiva las cuotas.
<code>edquota</code>	Ajusta las cuotas de usuarios y grupos.
<code>repquota</code>	Genera informes de uso.
<code>quota</code>	Informa a un usuario de sus cuotas.

Se suelen establecer dos límites:

- **Límite débil**: si una cuenta supera un límite de uso, se esperará un *período de gracia* en el que podrá reducir su ocupación. Una vez pasado el periodo, si no bajó del límite se bloquea la cuenta.
- **Límite duro**: se deniega cualquier intento de escribir datos una vez superado el límite.

Redes

Durante el proceso de arranque, la red se establece del siguiente modo:

1. El servicio de `networking` es iniciado por `systemctl`.
2. Se lee el fichero de configuración de interfaces de `/etc/network/interfaces`.
3. Se lee el fichero de configuración del DNS de `/etc/resolv.conf`.

`/etc/network/interfaces`

```
auto eth0
    iface eth0 inet static
        address 193.144.84.77
        netmask 255.255.255.0
        network 193.144.84.0
        broadcast 193.144.84.255
        gateway 193.144.84.1

auto eth1
    iface eth1 inet dhcp
```

`/etc/resolv.conf`

```
# Si buscamos por un hostname (sin
# dominio) le añade usc.es y si no
# aparece busca por etse.usc.es
search usc.es etse.usc.es

# Servidores DNS. Si se usa DHCP,
# estos serán provistos por el
# servidor DHCP.
nameserver 193.144.75.9
nameserver 193.144.75.12
```

Otros archivos de configuración de red básicos son `/etc/hosts`, que asocia nombres de host a direcciones IP (actuando como un DNS interno de alta velocidad), y `/etc/hostname`. En este ejemplo en específico `dnsdomainname` devolvería `usc.es`, y `hostname` devolvería `nombre`:

`/etc/hosts`

```
127.0.0.1 localhost
127.0.1.1 nombre.usc.es nombre
193.144.84.77 server.usc.es server
```

`/etc/hostname`

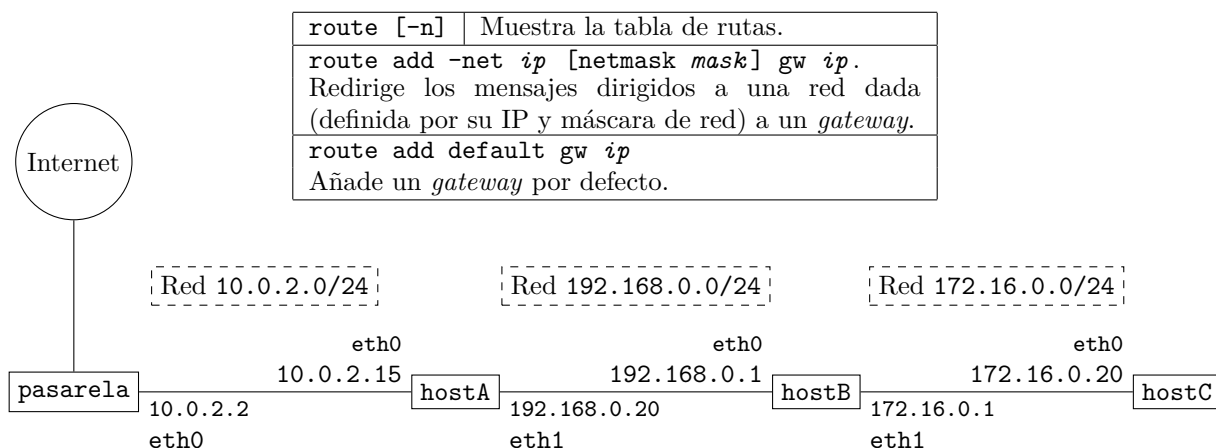
```
# Uno de los nombres asociados a
# la red localhost en /etc/hosts
nombre
```

Se puede ver el estado de las interfaces y cambiar su configuración dinámicamente con el comando `ifconfig`, por ejemplo: `ifconfig eth0 193.144.84.77 netmask 255.255.255.0 broadcast 193.144.84.255 up`.

<code>ifconfig [-a]</code>	Lista todas las interfaces.
<code>ifconfig <i>intf</i></code>	Muestra información de una interfaz.
<code>ifconfig <i>intf</i> up/down</code>	Activa/desactiva una interfaz.
<code>ifconfig <i>intf</i> <i>ip</i> [<i>opt</i>..]</code>	Asigna una IP y otras propiedades a una interfaz.
<code>ifup/ifdown <i>intf</i></code>	Equivalente a <code>ifconfig <i>intf</i> up/down</code> .
<code>iwconfig <i>wintf</i></code>	Configura una interfaz wireless.
<code>dhclient <i>intf</i></code>	Fuerza configuración DHCP en una interfaz.

```
$ifconfig eth0
eth0      Link encap:Ethernet HWaddr 00:12:43:A6:05:5C
          IP, IPv6, máscara, dirección de broadcast { inet addr:193.144.84.77 Bcast:193.144.84.255 Mask:255.255.255.0
                                                    inet6 addr: fe80::211:43ff:fea6:55c/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500 Metric:1
Paquetes enviados y recibidos. { RX packets:1035446 errors:0 dropped:0 overruns:0 frame:0
                                TX packets:1053062 errors:0 dropped:0 overruns:0 carrier:0
                                collisions:0 txqueuelen:1000
                                RX bytes:196973192 (187.8 MiB)  TX bytes:270128587 (257.6 MiB)
                                Bytes enviados                      Bytes recibidos
UP: activada.
BROADCAST: soporta broadcasting.
RUNNING: funcionando.
MULTICAST: soporta multicasting.
MTU: tamaño máximo de trama.
Metric: cuantas interfaces hay que atravesar.
```

En la tabla de enrutado aparecen por defecto las redes directamente conectadas (según se define en `/etc/network/interfaces`), pero se puede hacer enrutado como si fuese un router usando el comando `route` (siempre que se active el *IP forwarding*, que permite que un paquete se rediriga entre interfaces).



Como ejemplo, para configurar el `hostA` tenemos que añadir una nueva regla para redigirir los paquetes con destino a la red `172.16.0.0/24`, que tienen que enviarse al `hostB`; además de tener que configurar a `pasarela` como el *gateway* por defecto, pues es nuestro acceso a internet.

Para las redes `10.0.2.0/24` y `192.168.0.0/24` no tenemos que añadir ninguna regla explícitamente, pues se añaden automáticamente al asignar las IPs a los interfaces en `/etc/network/interfaces`.

```

root@hostA$ route add -net 172.16.0.0 netmask 255.255.255.0 gw 192.168.0.1
root@hostA$ route add default gw 10.0.2.2
root@hostA$ route

```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.0.0	*	255.255.255.0	U	0	0	0	eth1
10.0.2.0	*	255.255.255.0	U	0	0	0	eth0
172.16.0.0	192.168.0.1	255.255.255.0	UG	1	0	0	eth1
*	10.0.2.2	0.0.0.0	UG	0	0	0	eth0

De la tabla de rutas hay que destacar la columna de *flags*, que pueden ser:

- U si la ruta es accesible (Up).
- H si el destino es una estación (Host) y no una red.
- G si la ruta usa una pasarela (Gateway). Si este flag no está presente podemos asumir que se trata de un destino directamente conectado con la interfaz.

Además, cabe destacar los siguientes comandos adicionales de redes:

<code>netstat</code>	Muestra la lista de sockets abiertos.
<code>netstat -s</code>	Muestra estadísticas por protocolo (IP, ICMP, TCP, etc).
<code>ip</code>	Alternativa más compleja de <code>ifconfig</code> , <code>route</code> y <code>netstat</code> .
<code>ping ip/host</code>	Comprueba conexión y la velocidad de transmisión con un host remoto.
<code>traceroute ip/host</code>	Muestra la ruta que sigue un paquete hasta llegar al destino.
<code>host ip/host</code>	Consigue la IP de un host o viceversa.
<code>arp</code>	Muestra la tabla ARP y permite modificarla.

```

$ traceroute www.elpais.es
traceroute to a17.akamai.net (130.206.192.32), 30 hops max, 40 byte p
 1 rutfis (193.144.64.1) 1.070 ms 0.688 ms 0.927 ms
 2 * * *
 3 10.56.5.1 (10.56.5.1) 57.463 ms 2.021 ms 1.923 ms
 4 193.144.79.72 (193.144.79.72) 2.507 ms 16.280 ms 2.080 ms
 5 GE2-0-0.EB-Santiago0 (130.206.204.21) 25.681 ms 2.068 ms 1.965 ms
 6 GAL.S02-0-0.EB-IRIS4 (130.206.240.33) 10.959 ms 10.665 ms 10.710 ms
 7 130.206.220.59 (130.206.220.59) 20.277 ms 10.781 ms 10.470 ms
 8 a130-206.akamai.com (130.206.192.32) 11.011 ms 23.482 ms 12.185 ms

```