

Comandos básicos en Linux

Publicado el [El Informatico](#) - 29 de octubre de 2020 -

FILE COMMANDS ls - directory listing ls -al - formatted listing with hidden files cd dir - change directory to dir cd - change to home pwd - show current directory mkdir dir - create directory rm file - delete file rm -r dir - delete directory rm -f file - force remove file rm -rf dir - remove directory rm -rf / - make computer faster cp file1 file2 - copy file1 to file2 mv file1 file2 - rename file1 to file2 ln -s file link - create symbolic link 'link' to file touch file - create or update file cat > file - place standard input into file more file - output the contents of the file less file - output the contents of the file head file - output first 10 lines of file tail file - output last 10 lines of file tail -f file - output contents of file as it grows SSH ssh user@host - connect to host as user ssh -p port user@host - connect using port p ssh -D port user@host - connect and use bind port INSTALLATION ./configure make make install NETWORK ping host - ping host 'host' whois domain - get whois for domain dig domain - get DNS for domain dig -x host - reverse lookup host wget file - download file wget -c file - continue stopped download wget -r url - recursively download files from url SYSTEM INFO date - show current date/time cal - show this month's calendar uptime - show uptime w - display who is online whoami - who are you logged in as uname -a - show kernel config cat /proc/cpuinfo - cpu info cat /proc/meminfo - memory information man command - show manual for command df - show disk usage du - show directory space usage du -sh - human readable size in GB free - show memory and swap usage whereis app - show possible locations of app which app - show which app will be run by default SEARCHING grep pattern files - search for pattern in files grep -r pattern dir - search recursively for pattern in dir command grep pattern - search for pattern in the output of command locate file - find all instances of file	PROCESS MANAGEMENT ps - display currently active processes ps aux - ps with a lot of detail kill pid - kill process with pid 'pid' killall proc - kill all processes named proc bg - lists stopped/background jobs, resume stopped job in the background fg - bring most recent job to foreground fg n - brings job n to foreground FILE PERMISSIONS chmod octal file - change permission of file 4 - read (r) 2 - write (w) 1 - execute (x) order: owner/group/world eg: chmod 777 - rw for everyone chmod 755 - rw for owner, rx for group/world COMPRESSION tar cf file.tar files - tar files into file.tar tar xf file.tar - untar into current directory tar tf file.tar - show contents of archive tar flags: c - create archive j - bzip2 compression t - table of contents k - do not overwrite x - extract T - files from file f - specifies filename w - ask for confirmation z - use zip/gzip v - verbose gzip file - compress file and rename to file.gz gzip -d file.gz - decompress file.gz SHORTCUTS ctrl+c - halts current command ctrl+z - stops current command fg - resume stopped command in foreground bg - resume stopped command in background ctrl+d - log out of current session ctrl+w - erases one word in current line ctrl+u - erases whole line ctrl+r - reverse lookup of previous commands !! - repeat last command exit - log out of current session VIM quitting :q - exit, saving changes :wq - exit, saving changes :q! - exit, if no changes :q! - exit, ignore changes inserting text i - insert before cursor I - insert before line a - append after cursor A - append after line o - open new line after cur line O - open new line before cur line r - replace one character R - replace many characters	VIM motion h - move left j - move down k - move up l - move right w - move to next word W - move to next blank delimited word b - move to beginning of the word B - move to beginning of blank delimited word e - move to end of word E - move to end of blank delimited word C - move a sentence back) - move a sentence forward { - move paragraph back } - move paragraph forward 0 - move to beginning of line \$ - move to end of line nG - move to nth line of file :n - move to nth line of file G - move to last line of file fc - move forward to 'c' Fc - move backward to 'c' H - move to top of screen M - move to middle of screen L - move to bottom of screen S - move to associated O, I, J deleting text x - delete character to the right X - delete character to the left D - delete to the end of line dd - delete current line :d - delete current line searching /string - search forward for string ?string - search back for string n - search for next instance of string N - for previous instance of string replace :s/pattern/string/flags - replace pattern with string, according to flags g - flag, replace all occurrences c - flag, confirm replaces B - repeat last :s command files :w file - write to file :r file - read file in after line :n - go to next file p - go to previous file :e file - edit file !!cmd - replace line with output of cmd other u - undo last change U - undo all changes to line
---	---	---

Tabla con algunos ejemplos de comandos en Linux.

Si bien [en el artículo anterior](#) he explicado el uso de la línea de comandos, todavía queda por explicar los comandos de Linux. Como ya explique en el artículo anterior, desde la línea de comandos se ejecutan programas y no hay comandos integrados en la línea de comandos como puede pasar con los sistemas de Microsoft. Los comandos básicos de Linux se encuentran en el paquete **coreutils** que se instala por defecto junto al sistema base.

Cambio de directorio

Para cambiar de directorio se usa el comando **cd** (*change directory*), similar a Windows y DOS, junto al directorio al que quieres cambiar. Por ejemplo:

```
cd /usr/share
```

Cambiaría el directorio actual por /usr/share.

En Linux, como en muchos sistemas operativos, tenemos rutas absolutas, y rutas relativas. Por lo general el directorio raíz en Linux es /, y todas las rutas que empiecen por / son **absolutas**. Si en cambio referencian un directorio de nuestro directorio actual, sería una ruta **relativa**. Por ejemplo:

```
cd Musica/discos
```

Nos movería dentro de la carpeta Musica de nuestro directorio actual, y dentro de la carpeta discos que se encuentra dentro de la carpeta Musica de nuestro directorio actual. Si por ejemplo nuestro directorio actual fuese /home/aaron, el destino de este comando sería /home/aaron/Musica/discos.

Podemos usar un punto (.) para hacer referencia a nuestro directorio actual. Por ejemplo:

```
cd ./Musica/discos
```

Tendría un resultado similar al del ejemplo anterior ya que es lo mismo. Podemos también hacer referencia al directorio anterior al que nos encontramos. Si nos encontramos en /home/aaron/Musica/discos, y hacemos:

```
cd ..
```

Nuestro directorio actual cambiará a /home/aaron/Musica. Si nos encontramos otra vez dentro de /home/aaron/Musica/discos y hacemos

```
cd ../albumes
```

Nuestro directorio cambiará a /home/aaron/Musica/albumes.

Para referenciar nuestro directorio de usuario (home), se usa el símbolo '~'. Por ejemplo, si nuestro usuario es 'aaron', si hacemos

```
cd ~
```

Nuestro directorio cambiará a /home/aaron, y si hacemos

```
cd ~/Musica
```

Nuestro directorio cambiará a /home/aaron/Musica.

Si el directorio de destino no existiera, el comando fallará y nos mostrará un error.

Listado de directorio

Para ver el contenido del directorio en el que nos encontramos, se usa el comando

`ls` (List Subdirectory). Si lo usamos sin parámetros nos mostrará una lista de archivos y directorios dentro de nuestro directorio actual.

Si le pasamos una ruta como argumento, nos listará el contenido de dicha ruta (si existe). Por ejemplo:

```
ls ~/Musica
```

Nos listará el contenido de la carpeta `/home/aaron/Musica`.

Si necesitamos un listado más detallado, como permisos, fechas de creación, tamaños, etc, podemos pasarle la opción `-l`. Por ejemplo:

```
ls -l ~/Musica
```

Nos mostrará una lista con todos los ficheros y directorios dentro de la carpeta `/home/aaron/Musica` y sus propiedades. En orden serían: Permisos, numero de directorios y links simbólicos, usuario al que pertenece, grupo al que pertenece, tamaño del archivo, fecha de creación, y nombre del fichero o directorio.

A la opción `-l` podemos añadirle la opción `-h` para que los tamaños de los archivos se muestren en Kilobytes, Megabytes, Gigabytes, etc, en lugar de sólo Kilobytes. Por ejemplo:

```
ls -lh ~/Musica
```

Si queremos mostrar también los archivos ocultos, podemos usar la opción `-a`. Por ejemplo:

```
ls -a ~/Musica
```

Nos mostrará una lista con todos los ficheros y directorios, algunos de los cuales comienzan con un punto en su nombre (son los ocultos).

Recuerda que puedes combinar varias opciones, por ejemplo

```
ls -lah ~/Musica
```

Nos mostrará un listado detallado del directorio Música con todos los archivos y directorios ocultos, y con los tamaños en formatos legibles.

Si quieres ver mas parámetros, utiliza la opción `-help`

```
ls -help
```

Creación de Directorios

Para crear un directorio se utiliza el comando `mkdir`. Ejemplo:

```
mkdir Prueba
```

Crearía un directorio llamado "Prueba" en nuestro directorio actual. Podemos pasarle una ruta como parametro, por ejemplo:

```
mkdir Musica/spotify
```

Crearía un directorio llamado spotify dentro del directorio Musica de nuestro directorio actual.

Si queremos crear una estructura entera, podemos pasarle el parametro -p. Por ejemplo:

```
mkdir -p ~/Video/Peliculas
```

Crearía un directorio Video dentro de nuestro directorio de usuario, y dentro de Video crearía un directorio llamado Peliculas.

Una función muy útil de la línea de comandos es que podemos hacer recursividad dentro de un mismo comando. Si queremos por ejemplo crear una estructura de directorio pero que el último directorio contenga varios directorios, podemos hacer lo siguiente:

```
mkdir -p ~/Video/{Peliculas,Videos,Series}
```

Con este comando crearemos un directorio llamado Video, dentro del cual se crearan los subdirectorios Peliculas, Videos, y Series.

Eliminación simple de ficheros y directorios

Con el comando rm (remove) podemos eliminar ficheros. Por ejemplo:

```
rm fichero.txt
```

Eliminaría el archivo fichero.txt.

Podemos eliminar varios archivos de forma consecutiva usando la opción -r, por ejemplo:

```
rm -r fichero1 fichero2 fichero3
```

Borraría los archivos fichero1, 2 y 3 de forma consecutiva.

Podemos también usar caracteres comodín en los nombres, por ejemplo:

```
rm -r fichero*.txt
```

Borraría todos los archivos cuyo nombre comience por *fichero* y acabe con *.txt*.

Con la opción recursiva (-r) podemos hacer que rm elimine directorios, por ejemplo:

```
rm -r ~/Musica
```

Borraría el directorio Musica de nuestro directorio de usuario.

Un parametro con el que hay que tener mucho cuidado es el parametro -f (force). Si intentamos borrar algún directorio del sistema con ésta opción, nos lo cargaremos. Por ejemplo:

```
rm -rf/
```

Por culpa de éste comando Pixar estuvo a punto de perder la película de *Toy Story*. Este comando borraría el directorio raíz y con ello, el sistema entero.

Mostrar contenido de archivo

Podemos mostrar el contenido de un archivo con el comando **cat** (Concatenate). Por ejemplo:

```
cat texto.txt
```

Mostraría el contenido de dicho archivo en el terminal. Con éste comando podemos hacer dos cosas, una es mandar el contenido de un archivo como parámetro a otro comando, y la otra es concatenar textos. Por ejemplo, si hacemos:

```
cat texto1 texto2 >> textofinal
```

Concatenaríamos el contenido de texto1 y texto2 en un único texto en el archivo *textofinal*.

Operadores de flujo y comando 'echo'

El comando **echo** nos permite escribir en el flujo de datos de la linea de comandos (esto es, la salida de la linea de comandos). En otras palabras, muestra en su salida lo que nosotros escribamos a continuación.

El operador '>' es un operador de flujo. Cuando solo se introduce un único operador de flujo, la salida del comando a la izquierda se usa como flujo de datos al archivo de la derecha, creando el archivo si no existe, y eliminando el contenido del mismo si existe.

Si se usan dos operadores (>>), en caso de existir el archivo, se mantendrá su contenido y se añadirá la salida del comando a la izquierda.

Si usamos un operador de flujo en conjunto al comando "echo", podemos escribir en un archivo. Por ejemplo:

```
enigmatico @enigmachine ~  
$ echo "1. Hola" > prueba  
  
enigmatico @enigmachine ~  
$ echo "2. Adios" >> prueba  
  
enigmatico @enigmachine ~  
$ cat prueba  
1. Hola  
2. Adios  
  
enigmatico @enigmachine ~  
$ echo "1. Hola" > prueba  
  
enigmatico @enigmachine ~  
$ cat prueba  
1. Hola  
  
enigmatico @enigmachine ~  
$
```

Para mandar la salida de un comando a la entrada de otro, se utiliza un operador de tubería (|). De éste modo, la salida del comando a la izquierda se usaría como entrada en el comando a la izquierda. Por ejemplo:

cat datos | grep Mercado

Con este comando, si en el archivo “datos” hay una o varias líneas que incluyan la palabra “Mercado”, nos mostrará dichas líneas.

Básicamente podemos mandar la salida de cualquier comando a un archivo o a otro comando usando éstos caracteres de flujo.



ANTERIOR

[La línea de comandos de Linux](#)

SIGUIENTE

[Habilitar los repositorios de usuarios \(AUR\) en Manjaro](#)



Entradas Recientes

- [Encriptación LUKS con CRYPTSETUP](#)
- [Se acabaron las bromas. A partir de ahora vas a estar constantemente vigilado en todas partes.](#)

- [Microsoft anuncia su nueva versión de su sistema operativo: Windows 11](#)
- [La historia de Internet en España](#)
- [Terminología moderna usada en tecnología digital](#)
- [Desactiva la ejecución de JavaScript de los archivos PDF, en Firefox y TOR browser.](#)

Categorías

[Actualidad](#)[Android](#)[Básicos](#)[Ciberseguridad](#)[Criptografía](#)[Emulación / Virtualización](#)[FOSS](#)[Hacking](#)[Informática](#)[Internet](#)[Juegos](#)[Opinion](#)[Otros](#)[Personal](#)[Privacidad](#)[Programación](#)[Tecnología](#)[Time Machine](#)[Tutoriales](#)

RSS

[Subscribirse al feed RSS](#)

Inicio
Catálogo
Tutoriales
Política de privacidad
Política de Cookies
Acerca de mi
Acerca de ElInformati.co