Comandos básicos de Linux

1. comando pwd

Usa el comando **pwd** para encontrar la ruta del directorio (carpeta) de trabajo actual en el que te encuentras. El comando devolverá una ruta absoluta (completa), que es básicamente una ruta de todos los directorios que comienzan con una barra diagonal **(/)** Un ejemplo de una ruta absoluta es **/home/nombredeusuario**.

2. comando cd

Para navegar por los archivos y directorios de Linux, usa el comando **cd**. Te pedirá la ruta completa o el nombre del directorio, dependiendo del directorio de trabajo actual en el que te encuentres.

Supongamos que estás en **/home/nombredeusuario/Documentos** y deseas ir a **Fotos**, un subdirectorio de **Documentos**. Para hacerlo, simplemente escribe el siguiente comando: **cd Fotos**.

Otro escenario es si deseas ir a un directorio completamente nuevo, por ejemplo, **/home/nombredeusuario/Peliculas**. En este caso, debes escribir **cd** seguido de la ruta absoluta del directorio: **cd /home/ nombredeusuario/Peliculas**.

Hay algunos atajos para ayudarte a navegar rápidamente:

- **cd ..** (con dos puntos) para ir un directorio hacia arriba
- **cd** para ir directamente a la carpeta de inicio
- **cd-** (con un guión) para ir al directorio anterior (desde donde venimos)

Como nota al margen, el shell de Linux distingue entre mayúsculas y minúsculas. Por lo tanto, debes escribir el nombre del directorio de forma exacta.

3. comando ls

El comando **ls** se usa para ver el contenido de un directorio. Por defecto, este comando mostrará el contenido de tu directorio de trabajo actual.

Si deseas ver el contenido de otros directorios, escribe **ls** y luego la ruta del directorio. Por ejemplo, ingresa **ls/home/nombredeusuario/Documentos** para ver el contenido de **Documentos**. Hay variaciones que puedes usar con el comando **ls:**

- **ls -R** también listará todos los archivos en los subdirectorios
- **ls -t** ordena los archivos por fecha de modificación
- **ls -a** mostrará los archivos ocultos
- **ls -X** ordena los archivos por extensión
- **ls -S** ordena los resultados por tamaño de archivo
- **ls -l** listará los archivos y directorios con información detallada como los permisos, el tamaño, el propietario, etc.

4. comando cat

cat (abreviatura de concatenate, en inglés) es uno de los comandos más utilizados en Linux. Se utiliza para listar el contenido de un archivo en la salida estándar (sdout). Para ejecutar este comando, escribe **cat** seguido del nombre del archivo y su extensión. Por ejemplo: **cat archivo.txt**.

Aquí hay otras formas de usar el comando **cat**:

- **cat > nombredearchivo** crea un nuevo archivo.
- **cat nombredearchivo1 nombredearchivo2>nombredearchivo3** une dos archivos (1 y 2) y almacena la salida de ellos en un nuevo archivo (3)
- Podemos agregar el contenido de un archivo usando **cat nombredearchivo1 >> nombrearchivo2** agregando el contenido de 1 en 2.
- Podemos convertir un archivo a mayúsculas o minúsculas, **cat nombredearchivo | tr a-zA-Z> salida.txt**
- **Cat -n nombredearchivo** para mostrar además los números de líneas.
- **Cat -s nombredearchivo** para eliminar líneas vacías repetidas (manteniendo una).

5. comando cp

Usa el comando **cp** para copiar archivos del directorio actual a un directorio diferente. Por ejemplo, el comando **cp escenario.jpg /home/nombredeusuario/Imagenes** crearía una copia de **escenario.jpg** (desde tu directorio actual) en el directorio de **Imagenes**.

6. comando mv

El uso principal del comando **mv** es mover archivos, aunque también se puede usar para cambiar el nombre de los archivos.

Los argumentos en mv son similares al comando cp. Debes escribir **mv**, el nombre del archivo y el directorio destino. Por ejemplo: **mv archivo.txt /home/nombredeusuario/Documentos**.

Para cambiar el nombre de los archivos, el comando de Linux es **mv nombreviejo.ext nombrenuevo.ext**

7. comando mkdir

Usa el comando **mkdir** para crear un nuevo directorio: si escribes **mkdir Musica**, creará un directorio llamado **Musica**. También hay comandos adicionales de **mkdir**:

- Para generar un nuevo directorio dentro de otro directorio, usa este comando básico de Linux **mkdir Musica/Nuevoarchivo.**
- Se puede establecer el modo del archivo en su creación con el parámetro -m. Por ejemplo **mkdir -ma = rwx nombrearchivo** de modo que establecemos el modo de todos los permisos (-m junto con rwx) y a todos los usuarios (-a).
- Usa la opción **p** (padres) para crear un directorio entre dos directorios existentes. Por ejemplo, **mkdir -p Musica/2020/Nuevoarchivo** creará el nuevo archivo «2020».

8. comando rmdir

Si necesitas eliminar un directorio, usa el comando **rmdir**. Sin embargo, rmdir solo te permite eliminar directorios vacíos.

9. comando rm

El comando **rm** se usa para eliminar directorios y el contenido dentro de ellos. Si deseas eliminar el directorio con todo su contenid, como alternativa a rmdir, usa **rm -r**. Por si solo, el comando solo elimina ficheros sueltos. Como alternativa directa a rmdir se puede usar **rm -d.**

Especificando el parámetro -f **rm -f** elimina el elemento/os seleccionados sin confirmaciones, incluso si el fichero esta protegido contra escritura. Si puede eliminarlo, lo hará.

Nota: Ten mucho cuidado con este comando y verifica en qué directorio te encuentras. Este comando elimina todo y no se puede deshacer.

10. comando touch

El comando **touch** te permite crear un nuevo archivo en blanco a través de la línea de comando de Linux. Como ejemplo, ingresa touch **/home/nombredeusuario/Documentos/Web.html** para crear un archivo HTML titulado **Web** en el directorio **Documentos**.

11. comando find

Similar al comando **locate**(herramienta para la búsqueda de archivos y directorios basada en patrones de nombres, utiliza una base de datos indexada que hay que configurar), usando **find** también buscas archivos y directorios. La diferencia es que usas el comando **find** para ubicar archivos dentro de un directorio dado.

Como ejemplo, el comando find **/home/ -name notas.txt** buscará un archivo llamado **notas.txt** dentro del directorio de inicio y sus subdirectorios.

Otras variaciones al usar **find** son:

- Para buscar archivos en el directorio actual, **find . -name notas.txt**
- Para buscar directorios, **/ -type d -name notes.txt.**
- Para eliminar el archivo o directorio buscado **find -name nombre -delete**

12. comando sudo

Abreviatura de «**SuperUser Do**» (SuperUsuario hace), este comando te permite realizar tareas que requieren permisos administrativos o raíz. Sin embargo, no es aconsejable usar este comando para el uso diario, ya que podría ser fácil que ocurra un error si haces algo mal. (sudo)

Si queremos entrar en el directorio raíz del superusuario y convertirnos en él, usaremos el comando **su**. Para añadir un usuario en el sudoers file editaemos el fichero /etc/sudoers y añadiremos al usuario (**nombreUsuario ALL=(ALL:ALL) ALL**).

13. comando df

Usa el comando **df** para obtener un informe sobre el uso del espacio en disco del sistema, que se muestra en porcentaje y KB. Si deseas ver el informe en megabytes, escribe **df -m**.

14. comando chmod

chmod es otro comando de Linux, utilizado para cambiar los permisos de lectura, escritura y ejecución de archivos y directorios. Como este comando es bastante complicado, puedes leer el **tutorial completo** (en inglés) para ejecutarlo correctamente.

15. comando chown

En Linux, todos los archivos son propiedad de un usuario específico. El comando **chown** te permite cambiar o transferir la propiedad de un archivo al nombre de usuario especificado. Por ejemplo, **chown usuariolinux2 archivo.ext** hará que **usuariolinux2** sea el propietario del **archivo.ext**.

16. comando jobs

El comando **jobs** mostrará todos los trabajos actuales junto con sus estados. Un trabajo es básicamente un proceso iniciado por el shell. Si no muestra resultado es que no hay procesos presentes.

17. comando kill

Si tienes un programa que no responde, puedes cerrarlo manualmente utilizando el comando **kill**. Enviará una cierta señal al programa que se está ejecutando mal y le indica a la aplicación que finalice.

Hay un total de sesenta y cuatro señales que puedes usar, pero las personas generalmente solo usan dos señales:

- **SIGTERM (15)**: solicita que un programa deje de ejecutarse y te da algo de tiempo para guardar todo tu progreso. Si no especificas la señal al ingresar el comando kill, se utilizará esta señal.
- **SIGKILL (9)**: obliga a los programas a detenerse inmediatamente. El progreso no guardado se perderá.

Además de conocer las señales, también debes conocer el número de identificación del proceso (PID) del programa que deseas detener (**kill**). Si no conoces el PID, simplemente ejecute el comando **ps ux**.

Después de saber qué señal deseas usar y el PID del programa, ingresa la siguiente sintaxis:

kill [opción de señal] PID.

18. comando ping

Usa el comando **ping** para verificar tu estado de conectividad a un servidor. Por ejemplo, simplemente ingresando **ping google.com**, el comando verificará si puedes conectarte a Google y también medirá el tiempo de respuesta.

19. comando wget

La línea de comandos de Linux es muy útil: incluso puedes descargar archivos de Internet con la ayuda del comando **wget**. Para hacerlo, simplemente escribe **wget** seguido del enlace de descarga.

20. comando uname

El comando **uname**, abreviatura de Nombre de Unix, imprimirá información detallada sobre tu sistema Linux, como el nombre de la máquina, el sistema operativo, el núcleo, etc.

21. comando top

Como un terminal equivalente al Administrador de tareas en Windows, el comando **top** mostrará una lista de los procesos en ejecución y la cantidad de CPU que utiliza cada proceso. Es muy útil monitorear el uso de los recursos del sistema, especialmente para saber qué proceso debe terminarse porque consume demasiados recursos.

22. comando man

¿Confundido sobre la función de ciertos comandos de Linux? No te preocupes, puedes aprender fácilmente cómo usarlos directamente desde el shell de Linux mediante el comando **man**. Por ejemplo, al ingresar **man tail** se mostrarán las instrucciones manuales del comando tail.

23. comando echo

Este comando se usa para mover algunos datos a un archivo. Por ejemplo, si deseas agregar el texto «Hola, mi nombre es John» en un archivo llamado nombre.txt, debes escribir **echo Hola, mi nombre es John >> nombre.txt**

24. comando hostname

Si deseas conocer el nombre de tu host/red, simplemente escribe **hostname**. Agregar un **-I** al final mostrará la dirección IP de tu red.

25. comando useradd, groupadd, userdel, groupdel

Dado que Linux es un sistema multiusuario, esto significa que más de una persona puede interactuar con el mismo sistema al mismo tiempo. **useradd** se usa para crear un nuevo usuario, mientras que **passwd** agrega una contraseña a la cuenta de ese usuario. Para agregar una nueva persona llamada John, escribe **useradd John** y luego para agregar su contraseña, escribe **passwd 123456789**.

Podemos parametrizar la creación de usuario (**useradd**) para que cree automáticamente la carpeta del usuario en /home/nombreUsuario **useradd -m**. SI le agregamos el parámetro -g especificamos al grupo principal al que va a pertenecer el

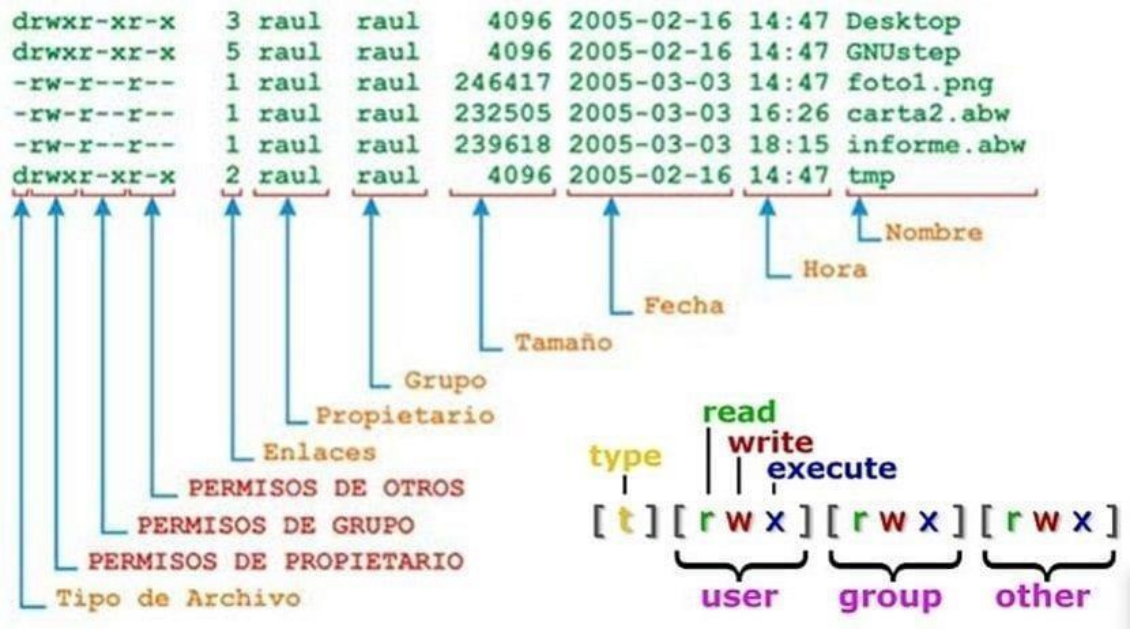
usuario (se puede especificar el grupo secundario con -G) **useradd -m -g Usuarios.**

Para añadir a un usuario a un grupo primero debe existir ese grupo, el cual podemos crear con el comando **groupadd NombreGrupo**. También podemos añadir un usuario a un grupo de usuarios con **adduser NombreUsuario NombreGrupo**.

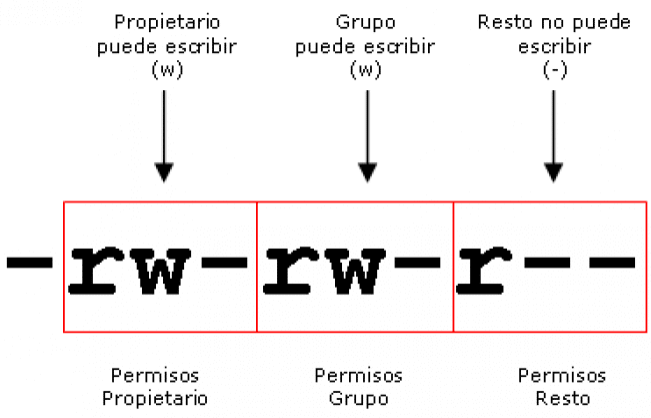
Si el usuario ya pertenece a un grupo, para cambiarlo a otro usaremos **usermod -g NombreGrupoDestino NombreUsuario.**

Eliminar un usuario es muy similar a agregar un nuevo usuario. Para eliminar la cuenta de usuario, escribe **userdel NombredeUsuario.** Y para borrar un grupo utilizaremos su análogo **groupdel NombreGrupo**.

Permisos Linux



Como se definen los permisos



Número	Binario	Lectura (*r*)	Escritura (*w*)	Ejecución (*x*)
0	000	✗	✗	✗
1	001	✗	✗	✓
2	010	✗	✓	✗
3	011	✗	✓	✓
4	100	✓	✗	✗
5	101	✓	✗	✓
6	110	✓	✓	✗
7	111	✓	✓	✓

Ejemplos:

- Permisos 777
  - Permisos de lectura escritura y ejecución para propietario, grupo y públicos(resto).
- Permisos 444
  - Permisos de lectura para propietario, grupo y públicos(resto).
- Permisos 421
  - Permisos de lectura para propietario, escritura grupo y ejecución públicos(resto).

Ejemplo para cambiar permisos

Chmod xxx nombre de archivo o carpeta

Chmod 777 ejemplo.txt

Para este tipo de operaciones debemos utilizar (sudo).

Tipos de sistemas de ficheros para particiones Linux

Sabremos ya que en Windows son de tipo ntfs.
Además del propio sistema de ficheros que utiliza Linux para instalarse, **también es compatible con el de otros sistemas como Windows o MAC** con el objetivo de facilitar el intercambio de información entre ellos. Será necesario instalar algún que otro paquete extra para añadir esta funcionalidad, concretamente **ntfs-3g** aunque, en el caso de Ubuntu, **ya está disponible de forma nativa** ésta.

- **ext**: del inglés (extended file system) es el sistema de ficheros transaccional nativo para sistemas operativos Linux, sea cual sea su distribución. Dispone de distintas evoluciones que hace de este sistema fichero, uno de los mejores jamás creados. **ext2**: con este sistema se mejoran las prestaciones de la primera versión, soportando ficheros de hasta 2 TB de tamaño y particiones de hasta 4 TB. El tamaño máximo del volumen puede ser de 16 TB.
- **ext3**: en este caso tendremos soporte para ficheros de 2 TB y volúmenes de hasta 32 TB.
- **ext4**: es la versión actual de este sistema, en el que se soportan ficheros de hasta 16 TB y volúmenes de hasta 1000 PB. Además, la gestión de energía y optimización de acceso y escritura también está mejorado de forma considerable.

Montaje y desmontaje de una partición

En Linux, **todos los elementos se tratan como si de ficheros y directorios se tratasen**, y **esto también incluye los volúmenes y particiones**. Para poder utilizar un **sistema de ficheros** que no sea el propio en donde está instalado el sistema, **necesitaremos montarlo previamente**. Para ello, cada partición se enlaza mediante la operación de montaje con otra estructura de directorios de la que cuelga, como si de una carpeta se tratara. Actualmente, **no es necesario utilizar el terminar para montar particiones típicas como las de Windows o las unidades de almacenamiento extraíble**. Eso sí**, si tenemos una versión server de Linux, sí que necesitaremos hacer esto de forma manual**. Además, forma parte del funcionamiento más básico del sistema y debemos de conocerlo.

Organización de directorios en Linux

La organización de directorios de Linux se lleva a cabo mediante una jerarquía establecida mediante el **FHS (Filesystem Hierarchy Estándar)** en el que se definen la distribución y nombres de ficheros y directorios de Linux.

La estructura de directorios y ficheros de Linux se organiza **en forma de árbol invertido,** es decir, desde un directorio principal denominado **"/" o directorio raíz**, cuelgan absolutamente todos los demás, incluso las particiones cuando son montadas. Los principales directorios que cuelgan de raíz son:

- **/bin:** contiene los comandos básicos del sistema y que tienen permisos de ejecución para todos los usuarios del sistema (la mayoría de ellos).
- **/boot:** Contiene los comandos de arranque del sistema como el kernel. En nuestra máquina, **podremos tener instalados varios kernels** y arrancar del que nosotros queramos, algo impensable en un sistema Windows. En este directorio también se encuentra el **cargador de arranque grub.** Cuando actualizamos un sistema Linux a un nuevo kernel, se creará una nueva entrada en el grub para arrancar desde el nuevo o desde el antiguo.
- **/dev:** contiene los archivos especiales para los dispositivos hardware de nuestro equipo. Además de las unidades de almacenamiento.
- **/etc:** directorio que almacena todos los ficheros de configuración del sistema. Todos los usuarios pueden ver los ficheros, pero **solo el root puede modificarlos**.
- **/home:** en este directorio se almacenan todos los directorios de trabajo de los usuarios creados en el sistema. **El funcionamiento es similar a "mis documentos"** en Windows.
- **/lib:** directorio en donde se almacenan las librerías básicas para trabajar en el sistema.
- **/media:** este directorio está destinado al montaje y desmontaje de las unidades de almacenamiento como particiones, USB y CD-ROM. Es opcional, y además en según qué distribución tendrá un nombre distinto, por ejemplo **/mnt.**
- **/sbin:** comandos que **solamente puede usar el root**, para la administración del sistema.
- **/proc:** lugar en donde se almacenan de forma dinámica las listas de procesos que se ejecutan en el sistema y las estructuras de datos del sistema.
- **/root:** directorio especial del súper usuario de Linux, por supuesto, solamente el root podrá escribir dentro de este directorio.
- **/tmp:** directorio exclusivamente destinado a la información temporal. **Podremos incluso crear una partición exclusiva para /tmp.**
- **/usr:** directorio en donde se almacenan los programas que nosotros instalamos, y **que no forman parte del sistema operativo básico.**
- **/var:** directorio que almacena ficheros de datos como log del sistema, datos administrativos, etc.

- Sistema de ficheros en Linux: Todo sobre su estructura (ayudalinux.com)