

Comandos en Linux

Identificación

uname (unix name)

El comando uname proviene de la abreviatura Unix Name. Es una herramienta CLI (interfaz de línea de comandos o en inglés, command-line interface) cuyo fin es el de mostrar información del sistema operativo.

Sintaxis:

- `uname [parametros]`

Ejemplos:

- `Uname -a`
Brinda toda la información disponible.
- `Uname -r`
Versión del kernel.
- `Uname -m`
Arquitectura del microprocesador.

who

El comando who te listará los usuarios conectados en el sistema en tiempo real indicando en la fecha y hora de conexión como así también su terminal.

whoami

Es un comando del tipo Unix, proviene de la concatenación de las palabras en inglés ¿Who am I? que significa, ¿Quién soy?.

Muestra el nombre de usuario efectivo del usuario actual cuando se invoca, que se entiende como el nombre del usuario en sesión.

Sintaxis:

- `whoami [opciones]`

Documentación y ayuda

man

El comando “man” permite ver los manuales de cada comando en línea.

El comando es generalmente el nombre del programa, utilidad o función. La acción por defecto es buscar en todas las secciones disponibles en un orden predefinido y mostrar sólo la primera página encontrada. Si queremos saber qué secciones tiene definidas la palabra clave en las páginas de manual, debemos utilizar la opción -k.

Sintaxis:

- `man [-k] [sección] comando`

Ejemplos:

- `man man`
Man dispone de su propio manual de uso para conocer las opciones y argumentos que pueden ser de utilidad a la hora de utilizar este comando.

- *man ls*
Ejemplo del manual del comando “ls”. Una vez que se ingresa se puede navegar con las flechas de desplazamiento y salir con la letra “q”.

help

El comando de ayuda es un comando del símbolo del sistema que se utiliza para proporcionar más información sobre otro comando. Puede usar el comando de ayuda en cualquier momento para aprender más sobre el uso y la sintaxis de un comando, como qué opciones están disponibles y cómo estructurar realmente el comando para usar sus diversas opciones. También se usa como sufijo de los comandos.

Ejemplos:

- *help cd*
Muestra la ayuda del comando “cd”.
- *[comando] --help*
Los comandos invocados con esta opción devuelven una resumida ayuda que describe la sintaxis general del comando y las opciones más utilizadas.

clear

Es una herramienta estándar del sistema operativo del tipo UNIX, y por lo tanto se encuentra disponible en GNU/Linux y la totalidad de sus distribuciones.

Esta herramienta del tipo CLI (interfaz de líneas de comandos) es utilizado para borrar y/o limpiar el contenido de la pantalla del emulador de terminal.

history

Imprime por pantalla un listado de los últimos comandos que se han pasado por la terminal desde la cuenta del usuario actual.

Ejemplos:

- *history -c*
Para borrar el historial de todos los comandos utilizados hasta el momento.

Instalación de paquetes o aplicaciones

sudo

Los comandos sudo y su son ampliamente utilizados en Linux para aumentar el nivel de acceso de un usuario con permisos limitados. Utilizando estas dos herramientas podemos realizar tareas que solamente son posibles a través de los permisos de otra cuenta. Específicamente “sudo” sirve para realizar una única acción, asociada al comando que acompaña, es decir, sería como pedir permiso para ejecutar ese comando y nada más.

Ejemplos:

- *sudo apt install sl*
Permite la instalación de un paquete, en este caso la aplicación “sl” (animación de tren con caracteres ASCII).

su (switch user)

El comando su (de substitute o switch user) se utiliza para personificar a otro usuario. Cuando se ejecuta sin especificar qué usuario, por defecto es root.

Ejemplos:

- `su -`
Opcionalmente, se puede utilizar el signo `-` para que el entorno de usuario sea igual al que se obtendría de haberse logueado directamente con esta última cuenta. En otras palabras, `su -` nos brinda acceso al mismo tipo de prompt y directorio inicial de trabajo como si hubiéramos iniciado sesión como root.
- `su -cuenta`
Lo mismo de antes sólo que se especifica la cuenta y para acceder a dicha cuenta.

reboot

Reinicia el sistema. Hay que considerar que debe tenerse permisos de root, por lo que quizás el comando se con el prefijo “sudo”.

Sintaxis:

- `reboot [opciones]`

shutdown

Se utiliza para apagar o reiniciar Linux desde la terminal.

Sintaxis:

- `shutdown [OPTIONS] [TIME] [MESSAGE]`

Ejemplos:

- `shutdown`
Para hacer un apagado del sistema por defecto en 1 minuto.
- `shutdown now`
Linux se apaga inmediatamente.
- `shutdown -r`
Linux se reinicia por defecto en 1 minuto.
- `shutdown -r +10 "Actualización de Hardware"`
Apagará el sistema en 10 minutos y notificará a los usuarios con un mensaje.

exit

El comando `exit` cierra la sesión de terminal actual del usuario logueado en ella. Si estas usando la terminal como superusuario, se cerrará la sesión y volverás a ser usuario estándar.

apt install / yum install (advanced packaging tool install)

Instrucción para instalar paquetes (.deb) Debian y sus derivados o bien (.rpm) Redhat y sus derivados.

Ejemplos:

- `sudo apt-get install -y gedit (Debian / Ubuntu)`
Instala el programa “gedit” respondiendo que sí a cualquier confirmación que lo solicite.
- `sudo rpm -ivh gedit.rpm (Red Hat / Fedora Linux)`
Instala el programa “gedit” y muestra información detallada del proceso.
- `sudo yum install gedit (Centos)`
Instala el programa “gedit”.
- `sudo apt-get install coreutils`
Coreutils es un paquete de software desarrollado por el proyecto GNU que contiene los comandos básicos estándares.
- `sudo apt install tree`
debian / Ubuntu / Linux Mint

- `yum install tree`
red Hat / Centos / Fedora
- `sudo pacman -S tree`
arch / Manjaro
- `zypper install tree`
openSUSE

Manejo de archivos y directorios

pwd (print working directory)

Cuya traducción sería mostrar directorio de trabajo, se utiliza para imprimir el nombre del directorio actual en una sesión de comandos. Este comando sirve para averiguar cuál es su ubicación en el árbol de directorios del sistema de ficheros.

Sintaxis:

- `pwd [opciones]`

cd (change directory)

Es una herramienta de interfaz de líneas de comandos (CLI) integrada en sistemas operativos del tipo UNIX, como GNU/Linux y disponible por defecto en todas sus distribuciones.

Esta utilidad cumple con la función de cambiar nuestra posición dentro del sistema de archivos a uno superior e inferior e incluso a una ruta absolutas y relativas.

Básicamente requiere indicar el nombre del directorio en el que deseas moverse.

Sintaxis:

- `cd [opciones] [{ruta relativa}{ruta absoluta}{Directorios especiales}]`

Ejemplos:

- `cd /home/usuario/Documentos`
Llevará al directorio Documentos dentro de la carpeta personal del usuario usuario1. En este caso he utilizado una ruta absoluta, empezando por el directorio raíz /, e indicando el camino completo hasta situarme a Documentos.
- `cd Documentos`
Para situarse dentro del directorio Documentos, que debe estar dentro del directorio de trabajo actual. La diferencia es que en este caso, aprovechando que estoy dentro del directorio principal de usuario, se utiliza una ruta relativa sin necesidad de indicar el camino completo.
- `cd ..`
Esta otra sentencia se utiliza para saltar un directorio hacia atrás respecto del que se encuentra. De este modo, si se encuentra dentro del directorio /home/usuario1/Documentos, se cambia un nivel hacia arriba hasta situarse en /home/usuario1.
- `cd ../../`
Esta opción es similar a la anterior, pero ahora le permitirá saltar de golpe dos directorios hacia atrás. Por tanto, si estaba en /home/usuario1/Documentos, ahora se cambia a /home

ls (list)

Esta herramienta no permite realizar listados de los elementos en un sistema de archivos: directorios, enlaces y tipos de archivos contenidos en un determinado directorio. Los resultados se muestran por defecto ordenados alfabéticamente y por la salida estándar.

Sintaxis:

- `ls [opciones] [ruta]`

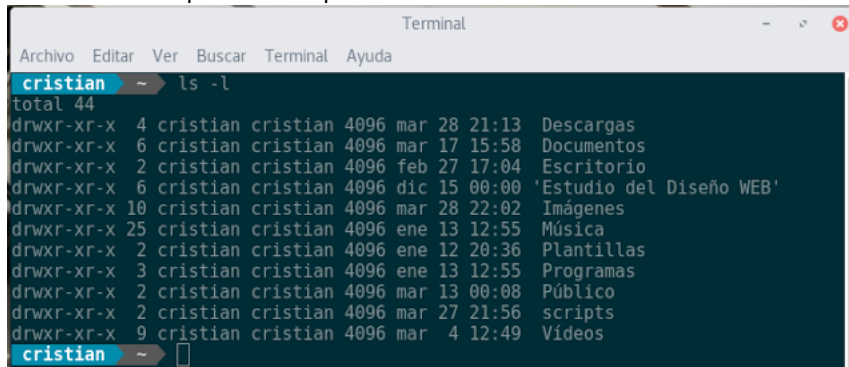
Parámetros:

- a: muestra todos los archivos incluyendo a los ocultos.
- c: ordena los archivos de acuerdo con la fecha de creación.
- d: muestra una lista en la que aparecen los directorios como si fuesen archivos (en vez de mostrar su contenido).
- f: muestra el contenido del directorio sin ordenar.
- l: muestra la lista de archivos con formato largo y con información detallada (tamaño, usuario, grupo, permisos etc.).
- p: añade un carácter al nombre del archivo para indicar a que tipo pertenece.
- r: coloca la lista en orden alfabético inverso.
- s: muestra el tamaño (kb) de cada archivo próximo al solicitado.
- t: ordena la lista de acuerdo con la fecha de cada fichero.
- R: muestra una lista con el contenido del directorio actual y de todos sus subdirectorios.

Ejemplos:

- `ls -l`

Muestra una lista ordenada alfabéticamente en detalles de los archivos y/o carpetas del directorio en el que estamos posicionados.



```
Terminal
Archivo  Editar  Ver     Buscar  Terminal  Ayuda
cristian ~ ls -l
total 44
drwxr-xr-x 4 cristian cristian 4096 mar 28 21:13 Descargas
drwxr-xr-x 6 cristian cristian 4096 mar 17 15:58 Documentos
drwxr-xr-x 2 cristian cristian 4096 feb 27 17:04 Escritorio
drwxr-xr-x 6 cristian cristian 4096 dic 15 00:00 'Estudio del Diseño WEB'
drwxr-xr-x 10 cristian cristian 4096 mar 28 22:02 Imágenes
drwxr-xr-x 25 cristian cristian 4096 ene 13 12:55 Música
drwxr-xr-x 2 cristian cristian 4096 ene 12 20:36 Plantillas
drwxr-xr-x 3 cristian cristian 4096 ene 13 12:55 Programas
drwxr-xr-x 2 cristian cristian 4096 mar 13 00:08 Público
drwxr-xr-x 2 cristian cristian 4096 mar 27 21:56 scripts
drwxr-xr-x 9 cristian cristian 4096 mar  4 12:49 Vídeos
cristian ~
```

La información en detalle que se divide en columnas de izquierda a derecha nos indican:

- La primera columna muestra los permisos del archivo.
 - La segunda columna muestra el número de enlaces de referencia del archivo.
 - La tercera columna muestra el propietario al cual pertenece el archivo.
 - La cuarta columna muestra el grupo a los que pertenece el archivo.
 - La quinta columna muestra el tamaño, por defecto en bytes.
 - La sexta columna muestra el mes de la última modificación en el archivo.
 - La séptima columna muestra el día de la última modificación en el archivo.
 - La octava columna muestra la hora de la última modificación en el archivo.
 - La novena columna muestra el nombre del archivo y/o carpeta.
- `ls -a`
Nos muestra todos los archivos y carpetas, incluidos los elementos que se encuentran ocultos dentro del directorio que se está listando (en GNU/Linux los ficheros ocultos comienzan con un punto [.]).
 - `ls -R`
Nos muestra lista de direcciones de los archivos y carpetas posteriores a nuestra posición y a su vez el contenido del contenido de manera recursiva.
 - `ls -h`
Nos muestra información del tamaño en kbytes de los elementos contenido de la carpeta en la que nos encontramos posicionados o que indiquemos y el total ocupado por el contenido de la carpeta, sin tomar en cuenta el tamaño de las subcarpetas.

- *ls -lah*
Nos muestra el tamaño de los mismos en formato de lectura “humano” (K,M,G) dentro de la quinta columna, que es la que nos brinda información del peso de nuestros archivos y carpetas contenidos en nuestra posición, y que por defecto son arrojados en bytes. Esto facilita nuestra lectura a la hora de visualizar el tamaño de nuestros archivos y carpetas. Cabe resaltar que en el caso de las carpetas la medición no es recursiva, es decir, que no muestra el peso que contiene la carpeta, sino que el resultado que nos arroja es el del tamaño que ocupa la carpeta en sí, sin medir el contenido de las subcarpetas recursivamente.
- *ls -lah /etc*
El comando “ls” también nos permite visualizar, con todas sus opciones, el contenidos de carpetas que están en otra dirección sin movernos de la posición en la que nos encontramos. Para esto debemos indicarle la ruta de la carpeta de la que queremos obtener información.

mkdir (make subdirectory)

Es usada para crear un nuevo subdirectorio o carpeta del sistema de archivos.

En el nombre del subdirectorio se puede utilizar un nombre simple o compuesto. El nombre simple es el nombre escrito directamente sin espacios, mientras que el compuesto se corresponde a varias palabras separadas con espacio, para lo cual se debe indicar el nombre entre comillas.

Sintaxis:

- *mkdir [Parámetros] [Nombre del directorio] [Ruta]*

Parámetros:

- m: modo, asigna la configuración de permisos especificada al nuevo directorio.
- p: crea directorios emparentados (en caso de que no existan).

Ejemplos:

- *mkdir test*
Crear el subdirectorio “test” dentro del directorio actual.
- *mkdir “test de funcionamiento”*
Crear el subdirectorio con el nombre “test de funcionamiento” dentro del directorio actual.
- *mkdir dir0 dir1 dir2*
Nos permite crear varios directorios de forma simultanea. Por defecto, y a menos que indiquemos una ruta como parámetro, estos se crearán en el directorio donde nos encontremos posicionados con el prompt; mkdir interpreta los espacios entre las cadenas ingresadas para diferenciar los directorios por crear. Es por eso que los nombres compuestos para nuevas carpetas deben ser indicado con algún tipo de comillas, de no hacerlo, mkdir interpretará que son tantos directorios como cadenas separadas por espacios.
- *mkdir -p padre01/hijo01*
Este parámetro permite la creación simultánea del directorio padre e hijo en una sola acción.

tree

Lista de forma recursiva los directorios y archivos contenidos en un ruta y los muestra en forma estructurada de tal forma que se puede apreciar que directorio o archivos se encuentran dentro de otros mediante una indentación y símbolos que representan esa dependencia.

rmdir (remove subdirectory)

Nos permite eliminar directorios vacíos de nuestra estructura.

Sintaxis:

- *rmdir [Parámetro] [Ruta] [Directorio]*

Parámetros:

-v: Muestra información detallada para cada directorio procesado.

Ejemplos:

- *rmdir DirVacio*
Elimina el directorio DirVacio que se encuentra en la misma posición que nuestro prompt.
- *rmdir Carpeta01 Carpeta02 Carpeta03 Carpeta04 Carpeta05*
Comando que de forma simultánea logra el borrado de varios directorio en una misma línea de ejecución.

rm (remove)

Cumple la función de eliminar archivos y directorios del sistema de archivos.

Sintaxis:

- *rm [Ruta] [Parámetros] [Elemento]*

Parámetros:

-f: elimina todos los archivos sin preguntar.
-i: pregunta antes de eliminar un archivo.
-r: elimina todos los archivos que se encuentran en un subdirectorio y por último borra el propio subdirectorio.
-v: muestra el nombre de cada archivo antes de eliminarlo.

Ejemplos:

- *rm archivo.txt*
Borra el archivo indicado que está en el mismo directorio desde donde se ejecuta el comando.
- *rm -d dir0*
Si el directorio se encuentra vacío, es decir, que no contiene ni archivos ni directorios en él, entonces se puede llevar a cabo su supresión con rm bajo el parámetro -d.

cp (copy)

Sirve para copiar archivos y directorios. El primer argumento del comando cp será el nombre del archivo a copiar, y el segundo argumento será la ruta absoluta o relativa del directorio en el que el archivo necesita ser copiado.

Sintaxis:

- *cp -r directorio/ ruta_de_destino/nombre_copia*

Parámetros:

-a: conserva todos los atributos de los archivos.
-b: hace un backup antes de proceder a la copia.
-d: copia un vínculo pero no el fichero al que se hace referencia.
-i: pide confirmación antes de sobrescribir archivos.
-p: conserva los sellos de propiedad, permisos y fecha.
-R: copia los archivos y subdirectorios.
-s: crea enlaces en vez de copiar los ficheros.
-u: únicamente procede a la copia si la fecha del archivo origen es posterior a la del destino.
-v: muestra mensajes relacionados con el proceso de copia de los archivos.

Ejemplos:

- *cp abc.txt backup.txt*
Copia el contenido en abc.txt a backup.txt. Aquí ambos archivos deben estar en el directorio de trabajo actual.
- *cp -i abc.txt backup.txt*
Si el archivo de destino ya existe, el contenido del archivo de destino será sobrescrito.

Podemos añadir la bandera -i al comando cp para obtener un aviso de confirmación antes de copiar.

- `cp abc.txt backup.txt test backup`

Copia múltiples archivos y directorios a un directorio en particular especificando todos los archivos y directorios de origen seguidos del directorio de destino al final. Esto copia los archivos abc.txt y backup.txt y la carpeta test en la carpeta backup.

- `cp *.txt backup`

El comando cp también permite la concordancia de patrones. Este comando copia todos los archivos con la extensión .txt en el directorio de trabajo actual a la carpeta backup.

- `cp -r pp Project`

Usamos la bandera -r o -R junto con el comando cp para copiar el directorio y sus subdirectorios y archivos al directorio de destino. Este comando copia todo el directorio pp y sus subdirectorios y archivos al directorio de destino Project. En este ejemplo, habrá un directorio pp dentro del directorio Project.

mv (move)

Modifica el nombre de los archivos y directorios moviéndolos de una ubicación a otra.

Sintaxis:

- `mv directorio ruta_de_destino/nombre_directorio`

Parámetros:

- d: hace una copia de seguridad de los archivos que se van a mover o renombrar.
- f: elimina los archivos sin solicitar confirmación.
- v: pregunta antes de sobrescribir los archivos existentes.

Ejemplos:

- `mv img files/images`

Mover el directorio img a la carpeta interna files cambiándole el nombre a images

- `mv directorio directorio_renombrado`

Para renombrar directorios usamos el mismo comando mv, pero no es necesario indicar una nueva ruta para el directorio, solo un nuevo nombre.

file

Determina el tipo de archivo en cuestión (archivo de texto, comprimido, binario, etc.) y lo muestra en la pantalla detallando sus características no su contenido.

Sintaxis:

- `file archivo`

Ejemplos:

- `file test.txt`

Indica que es un archivo de texto, con Unicode UTF-8 (por ejemplo).

diff

Se utiliza para determinar las diferencias entre archivos o directorios. Por defecto, diff no produce ninguna salida si los archivos son iguales. Si son distintos, diff reporta las diferencias línea por línea.

Sintaxis:

- `diff [-iqb] arch1 arch2`

Parámetros:

- i: ignora las diferencias entre mayúsculas y minúsculas.
- b: ignora los espacios en blanco.

-r: Para indicar que 2 directorios se comparen de forma recursiva. En otras palabras con la opción -r también se compararán todos los subdirectorios que están dentro del directorio analizado.

-q: Para que solo salgan en pantalla los ficheros que difieren de un directorio a otro.

Ejemplos:

- `diff '/home/joan/archivo_1.md' '/home/joan/archivo_2.md'`
Comparamos 2 ficheros de texto pudiendo incluso obtener el detalle de las diferencias existentes entre los 2 ficheros. Si fuesen binarios el comando no puede ver en detalle las diferencias si marcar que son distintos al menos.
- `diff -r -q 'directorio 1' 'directorio 2'`
Obtiene las diferencias existentes entre dos directorios.

tar

Es usado para comprimir o empaquetar un archivo o directorio. Es una de las opciones más populares del sistema Linux y el entorno Unix en general, debido a que puede reducir ampliamente y eficientemente el tamaño de los archivos comprimidos, llegando a tener una relación de compresión del 50%, sin necesidad de alterar las características de las carpetas.

Este comando también puede ser usado para transferir un gran número de archivos o carpetas entre diferentes servidores, hacer copia de seguridad de datos, sitios webs, entre otros, así como cargar y descargar carpetas utilizando menos espacio del dispositivo.

Sintaxis:

- `tar -cvf sampleArchive.tar /home/sampleArchive`

Parámetros:

--create o -c : permite crear un archivo.
--concatenate o -A : se encarga de unir dos ficheros TAR.
--append o -r : este comando permite añadir ficheros a un archivo TAR existente.
--update o -u : cumple la función de añadir ficheros nuevos a los que hay en el archivo.
--diff o -d : esta herramienta es utilizada para comparar un archivo con ficheros en el disco.
--list o -t : es el comando encargado de listar el contenido de un archivo.
--extract : se encarga de extraer archivos.
--directory dir o -C : funciona para cambiar el directorio dir antes de realizar alguna operación.
--file file o -f : usa el fichero file como el fichero de archivado.
--listed-incremental file o -g : es el encargado de realizar un archivado incremental.
--one-file-system o -l : este modificador solo trabaja con una partición.
--same-permissions o -p : se encarga de preservar la información de permisos.
--absolute-paths o -P : este modificador retiene la ruta absoluta de los ficheros.
--verbose o -v : cumple la función de listar todos los ficheros que trata.
--verify o -W : es el encargado de verificar el archivo después de escribirlo.
--exclude file : este comando es usado para excluir file del archivo.
--exclude-from file o -X : modificador que excluye ficheros listados en file.

Ejemplos:

- `tar -cvfz /tmp/logs.tgz ./log`
Comprime los archivos de la carpeta /log en un archivo llamado logs.tgz.
- `tar xvfz logs.tgz`
Lista los archivos comprimidos dentro del archivo empaquetado logs.tgz
- `tar -xvf sampleArchive.tar`
Extrae los archivos contenidos en sampleArchive.tar en la carpeta actual.
- `tar -xvf sampleArchive.tar -C /home/ExtractedFiles/`
Extrae los archivos a un directorio diferente al actual, es decir, lo hace en ExtractedFiles.

find

Localiza todos los archivos que coinciden con algún criterio. Nos permite realizar búsquedas muy precisas tanto de directorio como archivos.

Debe tenerse en cuenta que si no se le indica ninguna ruta, find, comenzara a buscar a partir del directorio en el que nos encontremos posicionados.

El campo “acción” nos permite ejecutar cualquier comando Linux sobre el archivo o directorio que fue objeto de la búsqueda.

Sintaxis:

- *find [ruta] [parámetro] [acción]*

Parámetros:

- depth: procesa, en primer lugar, el directorio en el que se encuentra y luego sus subdirectorios.
- maxdepth n: restringe la búsqueda a n niveles de directorios.
- follow: procesa los directorios que se incluyen dentro de los enlaces simbólicos.
- name: modelo localiza los nombres de los archivos que coinciden con el modelo propuesto.
- ctime n: localiza los nombres de los archivos creados n días atrás.
- user nombre_usuario: localiza los archivos pertenecientes al usuario específico.
- group nombre_grupo: localiza los archivos pertenecientes al grupo específico.
- path: ruta localiza a los archivos cuya ruta coincide con el modelo propuesto.
- perm: modo localiza los archivos con los permisos especificados.
- size +nK: localiza los archivos cuyo tamaño (en kilobytes) es mayor de especificado.
- print: imprime el nombre de los archivos que encuentra.
- exec: comando [opciones] {} \; ejecuta el comando especificado analizando el nombre del archivo localizado.

Ejemplos:

- *find /tmp /home /etc [parámetro] [acción]*
En este caso find analizará los directorios «/tmp», «/home» y «/etc»
- *find /home -iname foto.png*
Busca el archivo “foto.png” en la carpeta “/home” por su nombre.
- *find /home/scripts -user juan*
De esta forma nos imprimirá los archivos contenidos en el directorio que le indicamos y que pertenecen al usuario señalado.
- *find . -name "*.gz "*
Para ver todos los archivos cuyo nombre termine con .gz.
- *find /usr/doc -name "*.bak" -exec rm -f {} \;*
Para buscar a partir del directorio /usr/doc todos los archivos con extensión bak y eliminarlos.