Implantación de Sistemas Informáticos en Red

UD08

Intérprete de comandos y Administración básica de Directorios y Archivos

Implantación de Sistemas Informáticos en Red

Contenido

1. Definición

Aunque Linux tenga cada vez más programas gráficos, la mayoría no son más que fachadas brillantes para las herramientas en modo texto que hay por debajo. Aprender estas herramientas nos permitirá liberar todo el verdadero potencial de Linux y hacer nuestro trabajo mucho más rápido. Asimismo, seremos capaces de seguir trabajando aun en el caso de que el entorno gráfico falle y deje de funcionar, o bien podremos iniciar sesión y trabajar en remoto.

Las herramientas de la línea de comandos admiten scripts, lo que significa que podremos escribir un pequeño programa que realice una tarea más rápida o fácilmente que usando solo las herramientas estándar.

Opciones de iniciar el intérprete de comandos:

- Iniciar el intérprete de comandos en la interfaz gráfica (GUI) dentro de una ramada Terminal
- Iniciar sesión en el ordenador en una consola en modo texto
- Iniciar sesión en el ordenador en modo remoto mediante un protocolo de solo texto

El intérprete de comandos por defecto de la mayoría de las distribuciones de Linux es h (Boume Again Shell) o bash, que se basa en un shell más antiguo llamado Boume Shell.

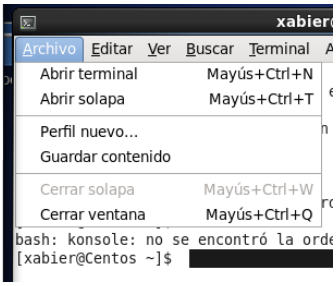
Hay disponibles más intérpretes de comandos, la mayoría muy parecidos a grandes rasgos, si bien difieren en los detalles. Cada cuenta define su intérprete de comandos por defecto, de manera que cada usuario puede cambiarlos si así lo desea.

Otros intérpretes de comandos incluyen tcsh, ksh y zsh. La elección del shell suele ser cuestión de gusto personal.

Terminal

Nuestras opciones se corresponderán con el escritorio que tengamos instalado. Asimismo, es posible ejecutar un programa terminal de un entorno de escritorio dentro de otro, si así lo deseamos. La mayoría de los programas de terminal incluyen la palabra “terminal” sus nombres, otros utilizan palabras como por ejemplo Konsole, así como el genérico xterm.

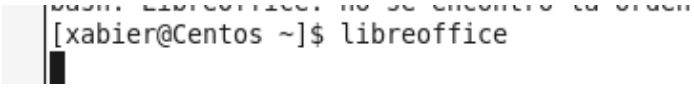
La mayoría de los programas de terminal soportan interfaces con fichas o como se dice pestañas (tabs) similares a las habituales en los navegadores Web. En la mayoría podemos abrir más ficha seleccionando Archivo>Abrir pestaña en la barra de Terminal. Tener abiertas múltiples pestañas resulta muy práctico porque nos permite ejecutar múltiples programas de forma simultánea, trabajar fácilmente con varios directorios.



Para cerrar una ventana pulsaremos sobre cerrar o escribiremos el comando exit.

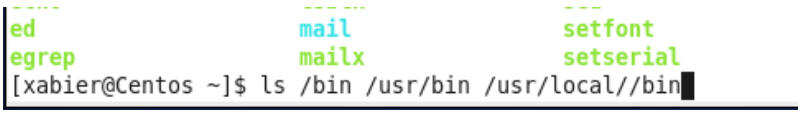
Ejecución de aplicaciones

Desde la terminal podremos ejecutar tanto aplicaciones de terminal como aplicaciones de GUI.

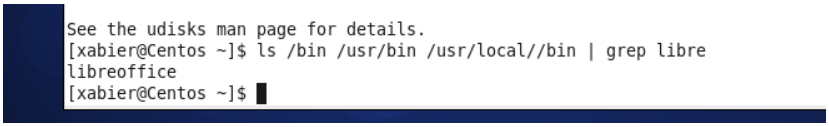


Ejecutar programas en modo texto

Linux guarda programas en muy distintos sitios, entre ellos: /bin, /usr/bin o /usr/local/bin.



Filtrar la salida:



Los programas que suelen estar reservados para el root suelen aparecer en /sbin, /usr/Sbin y /usr/local/sbin. Si un programa ejecutable aparece en uno de estos directorios (que conforman el path del sistema) lo podemos ejecutar simplemente escribiendo su nombre en el prompt:
$ free
Este comando de ejemplo muestra información sobre el uso de la memoria del ordenador.

Para saber qué directorios forman parte del path, escribiremos el comando siguiente:
echo $PATH

El resultado será un conjunto de nombres de directorios separados por dos puntos entre los que el intérprete de comandos busca en secuencia el comando introducido para tratar de ejecutarlo cuando no puede hacerlo directamente.

Muchos comandos aceptan “argumentos” que no son otra cosa que subcomandos o códigos que siguen al nombre del programa. Por ejemplos el comando cat, abreviatura de concatenate, puede mostrar rápidamente un texto breve en pantalla:

$ cat uno.txt

Ejecutar programas en entornos gráficos (GUI)

Podemos ejecutar programas GUI desde un terminal al igual que podemos hacerlo con programas basados en texto. Sin embargo, esto no es posible si hemos iniciado sesión con una consola VT en modo texto, así como tampoco podría funcionar con un inicio de sesión remoto aun en el caso de que estemos ejecutando X de forma local. Asimismo, debemos conocer también el nombre de fichero del programa para ejecutarlo. Este suele estar relacionado con el nombre que usamos para abrir el programa desde un menú del escritorio, pero no siempre es

idéntico. Por ejemplo, el nombre de fichero de GNOME Terminal es gnome-terminal, así que eso es lo que debemos escribir para abrir otro GNOME Terminal de ese modo.

Algunos programas GUI producen salidas en modo texto que pueden resultar útiles a la hora de registrar el origen de los problemas. Así pues, abrir un programa desde una ventana de terminal puede ser un buen primer paso para solucionar dichos problemas.

Asimismo, podríamos querer abrir programas de este modo porque resulta más rápido que buscarlos entre los menús de los entornos de escritorio, o bien porque el programa buscado directamente no aparece en dichos menús.

Ejecutar programas en segundo plano

Cuando abrimos un programa GUI desde una ventana de terminal, este se abre en su propia ventana o ventanas. La ventana de terminal sigue abierta, no obstante, pero deja de responder por lo general. Si tuviéramos que escribir más comandos en esta ventana, lo podríamos hacer seleccionándola y pulsando Control-Z. De este modo suspendemos el programa, es decir, entra en hibernación. En este estado, el programa GUI no responderá a entradas ni hará ningún trabajo.

Si desea utilizar tanto el programa GUI como el terminal del que se abrió, puede escribir bg (abreviación de background, segundo plano) en el terminal y ambos programas quedarán activos.

Escribir fg devuelve el programa suspendido al primer piano y le permite seguir ejecutándose, pero deja el shell incapaz de responder a algún comando.

Si, antes de abrir un programa ya sabe que desea ejecutarlo en segundo plano podrá hacerlo directamente añadiendo la letra ampersand & al final de la línea de comando:
$ gedit uno.txt &

O

Libreoffice &

Este comando abre el editor GUI gedit con el archivo uno.txt en segundo plano lo que nos permite editar el archivo y continuar usando el intérprete de comandos.
Esta característica es más útil para ejecutar programas GUI, pero en ocasiones se utiliza con programas basados en modo texto. Un programa complejo de cálculo matemático, por ejemplo podría estar diseñado para ejecutarse unos minutos o varias horas sin producir salida alguna. Así pues podría ser conveniente ejecutarlo en segundo plano y mantener el control del intérprete de comandos. Tenga presente sin embargo, que si abrimos un programa en segundo plano y genera una salida, dicha salida se mostrará en el shell, posiblemente interrumpiendo cualquier otra cosa que estuviéramos haciendo.

Cómo obtener listados de archivos

Para manipular archivos, resulta útil saber cuáles son. El comando ls (list) ofrece una lista con dicha información. El comando ls muestra los nombres de los archivos en el directorio. Si no le pasamos opciones mostrará los archivos en el directorio actual.

Resume de las opciones más importantes de ls:

F.L.	F.C	Descripción
--all	-a	Normalmente, ls omite aquellos archivos cuyos nombres comienzan con un punto (.). Estos dot files (archivos de punto) también conocidos como hidden files (archivos ocultos) suelen ser archivos de configuración que carecen de interés para el usuario común. Añadiendo este parámetro se mostrarán junto con los demás.
--Color	(no hay)	Esta opción produce una lista codificada por colores de los distintos directorios y de otros tipos de archivos especiales mostrándolos con diferentes colores. Algunas distribuciones de Linux configuran sus shell para utilizar esta opción por defecto.
--directory	-d	Normalmente, si escribimos el nombre de un directorio como opción, ls muestra los contenidos de dicho directorio. Lo mismo sucede si un nombre de directorio se corresponde con un comodín. Añadir este parámetro modifica este comportamiento para listar solo el nombre del directorio, lo que es a menudo preferible.
(no hay)	-l	Esta opción añade un código indicador al final de cada nombre de manera que se sepa de qué tipo de archivo se trata. Además produce un listado con información como tamaño, fecha, permisos, propietario y grupo de los archivos.
--file-type	-p	Esta opción añade un código indicador al final de cada nombre de manera que se sepa de qué tipo de archivo se trata.
--recursive	-R	La opción –R o –recursive hace que ls muestre los contenidos de los directorios de forma recursiva.

Opcionalmente se puede facilitar a ls uno o varios nombres de archivos y directorios, en cuyo caso ls mostrará información acerca de dichos archivos o directorios.

Ls –p /usr /bin/ls

Esta salida muestra tanto el archivo de programa /bin/ls como los contenidos del directorio /usr. Este último consiste básicamente en subdirectorios, indicados por la barra Inclinada “/” cuando se utiliza la opción —p. Por defecto, ls crea un listado organizado por nombres de archivo.

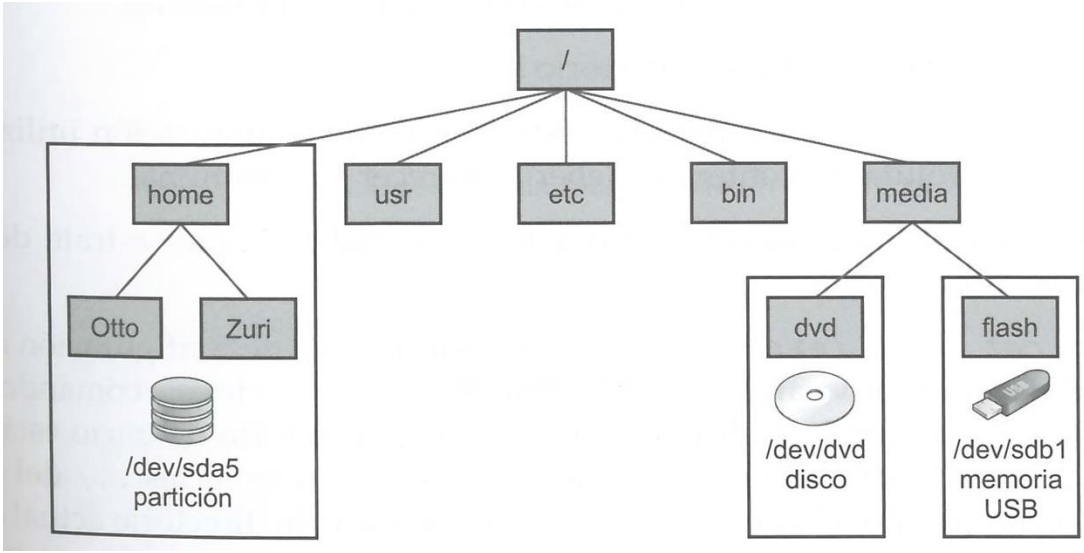
Un signo @ al final de un nombre denota un vínculo simbólico que es un archivo que apunta a otro.

Si necesitamos conocer la ruta completa de la ubicación actual, podemos utilizar el comando pwd:
$pwd

Linux utiliza una barra inclinada (slash) “/“ como separador de directorios. Los usuarios familiarizados con Windows, sabrán que este sistema utiliza la barra invertida (backslash) “\“para el mismo propósito. Conviene no confundir ambas. En Linux, una backslash tiene la función de carácter de escape para introducir caracteres que de otro modo se confundirían con

comandos o partes de comandos, espacios, partes de nombres de archivo, etc. Asimismo, tenga en cuenta que la barra inclinada, slash, no es un carácter válido para nombres de archivo en Linux por esa razón.

Referencia absoluta y relativa



Lo más común es que los medios removibles aparezcan como subdirectorios del directorio /media. La mayoría de subdirectorios se pueden dividir como particiones separadas e incluso situarse en discos diferentes. En la figura, el directorio /home reside en su propia partición, pero se accede a él exactamente de la misma forma que si formara parte de la partición raíz (/).

Hay tres formas de referirse a archivos y directorios:

Referencias absolutas: Son referencias relativas al directorio raíz (/), como en /home/otto/uno. txt para referirnos al archivo uno.txt en el directorio de inicio de Otto. Tales referencias empiezan siempre con el carácter slash (/).

Referencias al directorio de inicio (home) del usuario: El carácter tilde (~) hace referencia al directorio de inicio del usuario. Si un nombre de archivo comienza con dicho carácter, es como si sustituyera la ruta al directorio home del usuario. Así, para Otto, ~/uno.txt es equivalente a escribir /home/otto/uno.txt.

Referencias relativas: Estas referencias a archivos son relativas al directorio actual. Las referencias relativas pueden incluir subdirectorios, como en undirectorio/otroarchivo.txt.

En Linux, cada directorio inccluye un directorio especial oculto, . . , que hace referencia a su directorio padre. Así, si Zuri está trabajando en /home/zuri, se puede referir al archivo uno.txt de Otto, como . . /otto/uno.txt.

Funcionalidades del intérprete de comandos (shell)

Bash incluye varias características que facilitan mucho su uso. command completion (finalización de comandos) y command history (histórico de comandos).

Cómo utilizar la finalización de comandos

Escribimos parte de un comando o nombre de archivo y luego pulsamos la tecla Tab. Si solo hay un comando en la ruta que complete dicha entrada, rellenará el resto; lo mismo vale para la finalización de nombres de archivo.

Algunos de los detalles de cómo funciona la finalización de comandos varían entre las distintas distribuciones de Linux.

Histórico de comandos

Bash recuerda los comandos recientemente introducidos en el símbolo del sistema, que nos permite ahorrarnos cierto esfuerzo cuando tenemos que repetir un comando que se ya introducido recientemente. En su forma más básica, podemos utilizar la flecha hacia arriba para introducir el comando anterior.

Flecha arriba	Recupera la entrada anterior del histórico de comandos.
Flecha izquierda	Mueve el cursor a la izquierda un carácter.
Flecha derecha	Mueve el cursor a la derecha un carácter.
Control A	Mueve el cursor al principio de la línea.
Supr	Elimina el carácter situado bajo el cursor.
Retroceso	Elimina el carácter situado a la izquierda del cursor.
Control-X seguida de Control-E	Abre un editor completo en la línea de comandos actual.
Control-R	Busca un comando. Escriba unos pocos caracteres y el intérprete de comandos localizará el último comando que los incluya en su nombre. Se puede buscar el siguiente comando más reciente que incluya dichos caracteres pulsando de nuevo Control-R.

Administración de archivos y Directorios

Archivos

Crear archivos

Un comando que merece atención especial como medio de crear archivos es touch. Para ejecutarlo, escribimos su nombre seguido del nombre del archivo que deseemos crear:

touch newfile.txt

De este modo crearemos un archivo de texto llamado newfile.txt. De ordinario, no es necesario hacer esto para crear un archivo de tipo particular, dado que utilizaríamos un programa especializado para realizar el trabajo.

Si pasamos al comando touch el nombre de un archivo existente, éste actualizará las fechas y horas de acceso y modificación al archivo con las más recientes. Esto puede resultar práctico si utilizamos un comando que funcione con archivos basados en sus tiempos de acceso y deseamos que el programa trate un archivo antiguo como si fuera nuevo.

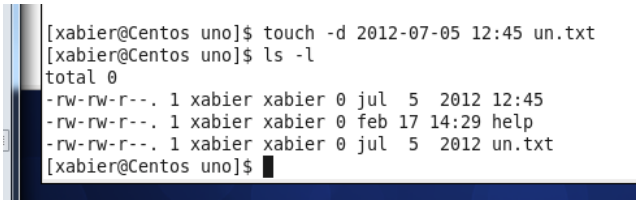
Asimismo, podríamos desear hacerlo si planeamos distribuir una colección de archivos y deseamos que todos tengan idénticas time stamps, o marcas de tiempo.

El comando touch admite una serie de opciones que modifican su comportamiento. Las más importantes son las siguientes:

No crear un archivo: Las opciones -c o --no-create indican a touch que no genere un nuevo archivo si no hay ninguno existente. Utilice esta opción si desea actualizar las marcas de tiempo, pero sin crear archivos vacíos por accidente en caso de escribir mal en nombre.

Establecer la hora en un valor específico: Podemos utilizar las opciones —d string o -- date = string para fijar la fecha de un archivo; esta fecha será la representada por la cadena de texto string, que puede tomar muchas formas.

Por ejemplo, si escribimos touch -d “July 4 2012” afile.txt haremos que la marca de tiempo del archivo, quede establecida el 4 de julio de 2012. Se puede lograr idéntico efecto con -t [[CC] YY]MMDDhhmm[. Ss], donde [[CC]YY]MMDDhhmm[.SS] es una fecha y una hora en un formato numérico específico, como 201207041223, para las 12:23 de la tarde del 4 de julio de 2012.



Copiar archivos

Si estamos trabajando en una shell en modo texto, el comando cp copia un archivo. Su uso básico consiste en pasarle el nombre de fichero de un archivo de origen, el nombre de fichero de un archivo de destino, un directorio de destino, o ambos.

El comando cp ofrece numerosas opciones que modifican su comportamiento. Algunas de las más útiles nos permiten modificar la propia operativa del comando de formas muy provechosas:

Sobre escritura forzosa (force overwrite): La opción - f o —-force obliga al sistema a que sobrescriba cualquier archivo existente sin pedir autorización.

Utilizar el modo interactivo: La opción -i o - -interactive hace que el comando cp pida autorización antes de sobrescribir cualquier archivo existente.

Conservar propiedad y permisos: Por lo general un archivo copiado es propiedad del usuario que ejecuta el comando cp y hace uso de los permisos por defecto dicha cuenta. La opción -p o - -preserve permite conservar la propiedad y permisos siempre que sea posible.

Realizar una copia recursiva: Si utilizamos la opción -R o --recursive y especificamos un directorio como origen, se copiará el directorio completos incluyendo todos sus subdirectorios. Aunque la opción — r (nótese la "r" minúscula) también suele realizar copias recursivas, su comportamiento con cualquier tipo de archivo que no sea un fichero ordinario o un directorio estándar, no está claro entre las distintas distribuciones Linux. La mayoría de las implementaciones de cp utilizan -r como sinónimo de -R, pero esto no está garantizado.

Realizar una copia para archivar: La opción -a o --archive es parecida a –R, pero conserva también la propiedad y copia los vínculos simbólicos tal cual son. La opción —R copia los archivos a los cuales apuntan los vínculos simbólicos (symbolic links) en lugar de los propios vínculos.

Realizar una copia de actualización: La opción -u o --update indica al comando cp que copie el archivo solo si el original, es más reciente que el que ya existe en el destino, o bien si en el destino no hay tal archivo.

Mover y cambiar de nombre los archivos

En un intérprete de comandos de modo texto, el mismo comando, mv, se utiliza tanto para mover como para cambiar el nombre de archivos y directorios. Su uso es muy parecido al de cp. Por ejemplo, si queremos mover outline . pdf a /publication, escribiríamos:

$ mv outline.pdf —/publication

Si especificamos un nombre de archivo con el destino, el archivo cambiará de nombre cuando se mueva al nuevo destino. Si especificamos un nombre de archivo y el directorio de destino es el mismo que el directorio de origen, es decir, el archivo no se mueve, el archivo cambiará de nombre. En otras palabras, los efectos de mv son muy similares a los de cp, salvo porque el nuevo archivo reemplaza, en vez de suplementar, al original.

Linux utiliza mv para cambiar el nombre a los archivos porque ambas operaciones son idénticas cuando el directorio origen y destino son el mismo.

mv hace lo siguiente:
Cuando el origen y el destino se hallan en el mismo sistema de archivo, mv rescribe las entradas en los directorios sin llegar a mover los datos físicamente.

Cuando movemos un archivo de un sistema de archivo a otro diferente, mv copia el archivo físicamente y borra los datos del archivo original.

El comando mv acepta muchas de las mismas opciones que cp sin embargo, --preserve, --recursive y --archive no se aplican a mv.

Utilización de vínculos (links)

En ocasiones resulta práctico referirse a un solo archivo por medio de múltiples nombres. Así en lugar de crear múltiples copias del mismo archivo con nombres distintos, podemos crear varios links a dicho archivo.

Linux soporta dos tipos de links, y crea ambos con el comando ln:

Hard link (vInculo fuerte): Un hard link es una entrada de directorio duplicada. Ambas entradas apuntan al mismo archivo. Como su funcionamiento se basa en unir estructuras de datos de bajo nivel del propio sistema de archivo, los hard links solo pueden existir en un único sistema de archivo. Cuando se usan hard links ninguno de los nombres de archivo ostenta prioridad alguna sobre el otro. Ambos están fijados directamente a las estructuras de datos del archivo y a los datos que contenga.

ln nombreOriginal nombreLink, donde nombreOriginal es el nombre original del archivo y nombreLink es el nombre del hard link.

Symbolic link o vínculo simbólico (también soft link o vínculo débil): Un symbolic link es un archivo que se refiere a otro por su nombre. Es decir, el symbolic link no es más que un archivo que contiene el nombre de otro archivo y cuando pedimos a un programa que lea o escriba desde uno de estos symbolic links, Linux redirige el acceso al archivo original. Como los symbolic links funcionan por referencias a nombre de archivo, pueden atravesar los límites de los sistemas de archivo.

ln – s nombreOriginal nombreLink para crear un symbolic link.

```
$ ln report.odt hardlink.odt
$ ln -s report.odt softlink.odt
$ ls -l
total 192
-rw-r--r--   2 rod users 94720 Jan 10 11:53 hardlink.odt
-rw-r--r--   2 rod users 94720 Jan 10 11:53 report.odt
lrwxrwxrwx   1 rod users    10 Jan 10 11:54 softlink.odt -> report.odt
```

Este ejemplo comenzó con un solo archivo, report . odt. Los dos primeros comandos crearon dos links, uno duro (hardlink. cdt) y otro simbólico (softllnk. odt).

ls -l muestra los tres archivos.

El archivo original y el hard link se pueden localizar por la presencia del valor 2 en la segunda columna de la salida del comando ls —l. Esta columna identifica el número de entradas de los

nombres de los archivos en el sistema de archivos que apuntan al mismo archivo; así pues, un número superior a 1 indica la existencia de hard links. El link simbólico se denota con una ‘1” (ele minúscula, no el número uno) como primer carácter de la cadena de caracteres de los permisos (lrwxrwxrWX) para soft link.

Ambos tipos de vínculos resultan útiles para referirse a archivos utilizando múltiples nombres o en múltiples directorios.

Por ejemplo, si escribimos una carta que enviamos a múltiples destinatarios, nos gustaría guardar una copia en cada directorio dedicado a dicho destinatario. En tal situación, cualquier tipo de link resultaría adecuado, pero cada uno tiene sus propias implicaciones. Sobre todo, si utilizamos links simbólicos y borramos el archivo original deja el archivo completamente inaccesible. El link permanece pero no apunta a ningún archivo. Si, por el contrario, usamos hard links, es preciso borrar “todas” las copias del archivo para eliminarlo. Esto es debido a que los hard links son duplicados de las entradas de directorio que apuntan al mismo archivo, mientras que los links simbólicos son archivos separados que se refieren al original por su nombre.

Si modificamos un archivo accediendo a su link simbólico, o por cualquier nombre vinculado como hard link, debemos asegurarnos de que el programa que usemos modificará el archivo original. Algunos programas crean una copia de seguridad del archivo original, que podemos utilizar para recuperarlo en caso de error. La mayoría de los editores hacen esto de una manera tal que la copia es un nuevo archivo y escriben los cambios en el archivo original, y de este modo afectan tanto al archivo original como al link. Otros programas, por su parte, cambian el nombre del archivo original y luego escriben un archivo nuevo con los cambios. Si un programa actúa así, y hemos accedido al archivo a través de un link, el archivo vinculado no se verá afectado por los cambios. En caso de duda, lo mejor es comprobar qué es lo que hace el programa en efecto, para aseguramos de que es lo esperado.

Si hay que crear un link a un directorio, hay que tener en cuenta que esto solo se puede hacer normalmente mediante links simbólicos. Los hard links entre directorios son potencialmente peligrosos en lo relativo a las estructuras de datos de bajo nivel en el sistema de archivo. Por lo tanto, el comando in solo permite al superusuario crear ese tipo de links.

Aun así, la mayoría de los sistemas de archivo no permiten hard links entre directorios, así que ni siquiera el root podría crearlos. Por contra, cualquier usuario puede crear links simbólicos a un directorio.

Las instalaciones Linux hacen uso de links, sobre todo simbólicos, en varios lugares. Por ejemplo, Linux llama a los scripts de inicio de sistema a través de links simbólicos ubicados en directorios dedicados a condiciones específicas de arranque, conocidos como “niveles ejecución” (runlevels).

Eliminar archivos

El comando rm borra archivos desde los intérpretes de comandos en modo texto. Como la de esperar, pasamos los nombres de uno o más archivos al comando como parámetros:

rm Outline.pdf outline.txt

Este ejemplo elimina dos archivos, outline.pdf y outline.txt. Si desea eliminar un árbol de directorios completo, pasaríamos los parámetros -r, -R o --recursive al Comando rm junto con el nombre del directorio:

Rm -r viejo/

La opción i hace que el comando rm pida confirmación antes de borrar cada archivo. Esta es una medida de seguridad útil; no obstante, si estamos seguros de lo que queremos borrar, podemos utilizar la opción - t o - - force para que todo se borre.

Una vez borrado con rm lo borrado no pasa a la papelera.

Comodines

? (Interrogación de cierre): Una interrogación de cierre ? ocupa el lugar de un solo carácter. Por ejemplo b???n se puede corresponder con: balon, bacon, buzon o cualquier otro nombre de cinco caracteres que comience con una b y acabe con una n.

* (Asterisco): Un asterisco equivale a cualquier carácter o conjunto de caracteres incluida la ausencia de caracteres. Por ejemplo b*n equivale igualmente a balón y a bn.

Valores entre corchetes: Los caracteres incluidos entre corchetes normalmente equivalen a cualquier carácter del conjunto. Por ejemplo b[al] [co] n equivale a balon y bacon, pero no a buzon. También es posible especificar Un rango de valores, por ejemplo: p [a-Z] ne s, que incluiría panes o pones.

Directorios

Crear directorios

En la mayoría de los casos, utilizaremos mkdi r sin opciones adicionales al nombre del directorio, pero es posible modificar su comportamiento en ciertas formas:

Establecer el modo (mode): La opción -m modo o --mode=modo indica al directorio nuevo que adopte el permiso especificado en modo, que se expresa en forma de número octal.

• Crear directorios padre: Normalmente, si especificamos la creación de un directorio dentro de otro directorio que no existe, mkdir responde con el mensaje de error No such file or directory (No existe el archivo o directorio) y no crea el directorio solicitado. Ahora, si incluimos el parámetro -p o --parents, mkdir crea los directorios padres que sean necesarios.

Por ejemplo, si escribimos mkdir first/second, obtendremos un mensaje de error y creando tanto first como second.

Borrar directorios

Debemos entender que rmdir solo puede borrar directorios “vacíos”. Si un directorio contiene archivos de cualquier tipo, no funcionará. No obstante, podrá utilizar la opción -p para eliminar una serie de directorios anidados, siempre y cuando ninguno de ellos contenga archivos.

Podemos utilizar el comando rm, junto con la opción -r O –recursive para eliminar directorios no vacíos.

Administración de directorios

Los directorios en Linux no son más que un tipo especial de archivo, ya archivos que contienen ficheros y otros archivos. Así pues, cabe la posibilidad de utilizar la mayor parte de las herramientas de manipulación de archivos descritas.

Pero es preciso tener en cuenta algunas precauciones:

. Es posible utilizar el comando touch para actualizar las marcas de tiempo de un directorio, pero no para crear un directorio.

• Se puede usar cp para copiar un directorio. Sin embargo, hay que utilizar la opción - r, -R, - - recursive, -a o - -archive para copiar el directorio y todo su contenido.

• Se puede emplear el comando mv para mover o cambiar el nombre de un directorio.

• Podemos emplear ln con su opción -s para crear un link simbólico a un directorio.

Sin embargo, ningún sistema de archivo de Linux soporta hard links a directorios.