Jose Ignacio López-k idatzia
Asteazkena, 2005(e)ko urria(r)en 19-(e)an 02:10etan

There are no translations available.
Adentrate en el "mundo Linux". Conoce los comandos más básicos de la Shell

Introducción

Este artículo trata de dar un breve repaso a los comandos más básicos del sistema operativo Linux, permitiendo al usuario novel realizar operaciones sencillas que le guíen en sus primeros pasos.

Antes de comenzar, el usuario menos avezado podría preguntarse: pero, ¿qué es Linux? Linux es un sistema operativo multitarea y multiusuario concebido para funcionar en ordenadores personales (PCs). Esta basado en el sistema operativo UNIX, creado en 1969 en los laboratorios Bell de la empresa AT&T, y se desarrolló inicialmente como una versión gratuita de este último. Al tener el código disponible para todo el mundo, y gracias entre otras cosas al auge de la Internet, Linux ha ido evolucionando hasta ser hoy en día una de las opciones preferidas en el mercado no sólo personal sino también profesional.

Debido a su procedencia, originalmente Linux recibía todos las ordenes mediante comandos de texto. Actualmente los usuarios se pueden encontrar con un entono mucho más agradable y más parecido a otros sistemas como Windows. Sin embargo, los comandos de texto siguen existiendo y son muy necesarios en más de una ocasión, sobre todo cuando el usuario va indagando más y más.

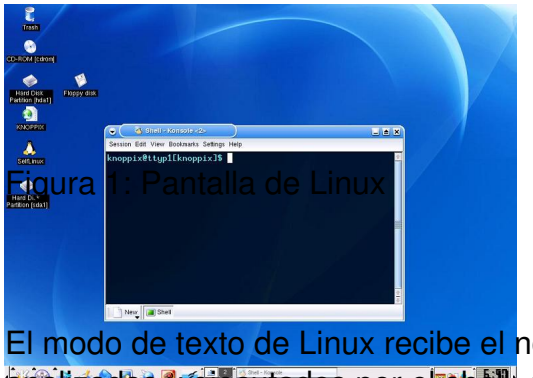


Figura 1: Pantalla de Linux

El modo de texto de Linux recibe el nombre de modo consola. En la consola, los comandos son analizados y ejecutados por el shell, o interprete de comandos. Existen muchos interpretes de comandos distintos. Por ejemplo, en MS-DOS el shell es el command.com. En Linux, los más populares son el *sh*, el *csh*, el *ksh* y el *bash*. Cada uno de ellos se diferencia del anterior en que mejora y complementa las órdenes existentes y añade nuevas posibilidades. Por su popularidad, este artículo se centrará en el bash.

A continuación se expondrán algunos de los comandos de texto más comunes y usados en Linux, así como ejemplos de la salida por pantalla de su ejecución para que el usuario pueda seguir los ejemplos paso a paso.

Primeros pasos en la consola de Linux

Como se comentó en la introducción, Linux es un sistema multiusuario. Cada usuario dispone de su propio espacio personal, conocido como cuenta. Al ser un espacio propio, el acceso a la cuenta está protegido mediante una contraseña. De esta manera, el usuario tendrá un identificador propio (login) y una contraseña (password) asociadas a una cuenta que lo identificarán en el sistema. Cuando alguna persona quiera acceder a su cuenta, lo hará desde una pantalla como la que se muestra en la figura.

Una vez concedido el acceso, el usuario estará dentro de su cuenta, que no es más que un directorio igual que el resto, pero del cual se disponen de todos los permisos. Cada usuario tendrá su propio directorio, del que colgarán más directorios y ficheros, todos con los permisos del usuario. El dueño de cada fichero/directorio es libre para decidir si permite que los otros usuarios accedan a ellos o no, añadiendo o revocando los permisos necesarios. El tema de los permisos se volverá a revisar cuando se hable del sistema de ficheros.

Lo primero que verá el usuario será el *prompt*. El *prompt* es el indicador que utiliza el shell para avisar de que está a la espera de comandos. Para el lector familiarizado con MS-DOS, el prompt que se utiliza en este ultimo sistema operativo suele ser algo como C:>. En Linux este indicador es totalmente configurable, aunque generalmente suele mostrar el directorio actual y el nombre de la maquina, o bien nombre de máquina y nombre de usuario. Un aspecto muy común puede ser el siguiente: [CIBELES@nacho] Tras el
prompt
, se introducen las ordenes por el usuario.

Comandos básicos

La mayor parte de los comandos que se utilizan en un sistema operativo sirven para recorrer el árbol de directorios y para la manipulación de ficheros. Linux no es una excepción al respecto, por lo que la mayor parte de los comandos simples son similares en cuanto a funcionamiento a los de otros sistemas, aunque varíen en sintaxis. Para comprender mejor los comandos que se van a explicar a continuación, conviene tener una consola abierta para poder practicar los comandos y ver los resultados por uno mismo. Aunque existen muchos más comandos (y posiblemente mucho más complejos) que los expuestos aquí, se pretende proporcionar al lector con la cantidad suficiente como para que sus primeros paseos en Linux le resulten de

utilidad. Antes de comenzar, es importante precisar que Linux distingue entre mayúsculas y minúsculas, por lo que hay que ser cuidadoso y respetar la sintaxis de los comandos.

pwd: imprime el directorio actual

El comando pwd nos muestra la ruta de directorios en la que estamos situados en este momento. Podemos hacer uso de este comando siempre que no sepamos exactamente el lugar en el que estamos.

ls: lista archivos

Este comando lista los archivos (incluyendo directorios) que hay dentro del directorio actual. El equivalente en DOS sería dir. El comando ls tiene bastantes parámetros que nos permitirán cambiar su comprotamiento, aunque los más usados son los siguientes:

- -l : no sólo muestra los archivos, sino que para cada uno de ellos indica usuario, grupo, tamaño, permisos, etc.

- -a: muestra todos los archivos, incluyendo los ocultos.

- -t: ordena los archivos por fecha de modificación.

Estas opciones pueden combinarse usando un solo signo -, por ejemplo, -lat muestra información larga sobre todos los archivos incluidos los ocultos y ordenados por fecha de modificación.

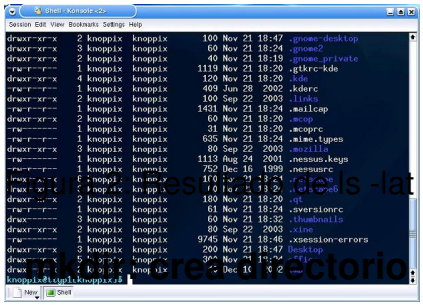
Un tema importante para comprender la salida del comando ls es entender el tema de los permisos en Linux. Cuando se ejecuta ls l, aparece una primera columa con diez caracteres que indican el tipo de fichero y los permisos, de la siguiente manera:

- El primer carácter indica qué tipo de archivo es. Una d indica un directorio, un l es un

enlace y un - representa un archivo normal.

· Los siguientes tres caracteres indican los permisos que tiene el archivo para el dueño. La primera columna indica lectura, y puede tener una r si está habilitado en permiso o - en caso contrario. La segunda indica permiso de escritura, y puede tener una w si está habilitado o un - si no lo está. La tercera indica permiso de ejecución, y puede tener una x si está habilitado o un - si no lo está.

· Los otros tres caracteres son los permisos para el grupo, y los últimos tres son los permisos para el resto de los usuarios.



El comando mkdir nos permite crear un directorio, igual que en MS-DOS. La manera correcta de usarlo es la siguiente:

```
mkdir [-p] directorio
```

Podemos usar rutas relativas o absolutas. Cuando se usan rutas relativas, se usa el directorio actual como raíz. Por ejemplo, supongamos que queremos saber el directorio en el que estamos, crear otro dentro de él y consultar el resultado. La secuencia completa sería la siguiente:

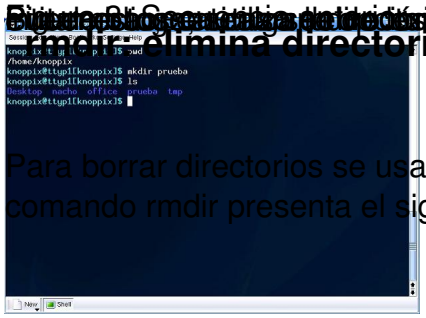


Figura 8: Se utiliza el comando mkdir para crear un directorio llamado prueba, que aparecerá en el directorio personal del usuario knoppix. Para

rmdir: elimina directorios

Para borrar directorios se usará el comando rmdir, equivalente al rmdir de MS-DOS. El comando rmdir presenta el siguiente formato:

rmdir [-ri] directorio

Una condición para que el comando funcione correctamente es que los directorios a eliminar estén vacíos. Si no lo están, habrá que borrar los ficheros que contiene antes de borrar el directorio.

Para borrar todos los directorios (vacíos) que cuelgan de uno dado se utiliza la opción r. Con la opción i entramos en el modo interactivo, el que se nos pregunta antes de eliminar cada directorio.

cd: cambiar de directorio

Para movernos entre directorio podemos utilizar el comando cd, igual que en MSDOS. Por ejemplo, cd / nos lleva al directorio raíz

Para cambiar a un directorio dentro del actual, podemos hacer cd directorio, aunque tambíen podemos emplear rutas absolutas como cd /home/ilopez.

Si utilizamos cd sin parámetros, accederemos al directorio personal del usuario (home). Esto puede ser muy útil cuando queramos regresar a nuestro directorio personal después de haber estado en cualquier otro sitio.

Para cambiar al directorio padre del actual, se hará cd .. (con espacio entre cd y ..). Para volver al directorio en el que estaba el usuario antes de ejecutar el último cd, se puede ejecutar cd -.

cat, more, less: examinar el contenido de un fichero.

Una vez que el usuario ya conoce como listar los archivos de un directorio, como moverse entre directorios y como crear y borrar directorios, lo siguiente que surge es la necesidad de examinar el contenido de los ficheros. Los comandos cat, more y less permiten hojear el contenido de un fichero, teniendo cada uno sus particularidades que se comentarán a continuación. El formato de los tres comandos es el siguiente:

cat fichero

more fichero

less fichero

cat es el comando más simple, pues muestra el contenido de un fichero mostrándolo por pantalla y sin ningún tipo de pausa. Un caso especial se produce cuando se ejecuta el comando cat sin parámetros. Entonces el comando se queda esperando a que se introduzcan caracteres por pantalla, mostrándolos línea a línea hasta que pulsa Ctrl-D.

more y less sí permiten hacer pausas durante la visualización de los datos. Para avanzar entre pantallas se pulsa cualquier tecla. La diferencia entre ambos consiste en que el comando less utliza las teclas de flechas para poder avanzar y retroceder por el fichero, cualidad de la que carece el comando more.

cp: copiar ficheros

El comando cp es el que se utiliza para copiar archivos (equivalente a copy de MS-DOS). El formato del comando cp es el siguiente:

cp [-frivu] origen destino

Para poder copiar un fichero es imprescindible que se tenga permiso de lectura del fichero origen y permiso de escritura en el directorio destino.

Cuando el fichero destino ya existe, el sistema actuará con una acción por defecto diferente según el sistema, pero que el usuario puede modificar con los parámetros de entrada:

- -f: sobreescribe el fichero destino.
- -i: pregunta al usuario sobre si debe sobreescribir o no.
- -u: sólo sobreescribe si el fichero destino es más antiguo que el origen.

Otros parámetros de interés son:

- -r: copia recursivamente directorios y subdirectorios.
- -v: muestra por pantalla las operaciones que realiza el comando.

mv: mover ficheros

El comando mv mueve ficheros de un lugar para otro. También sirve para renombrar un fichero. Si se ejecuta mv viejo nuevo, el archivo viejo pasará a llamarse nuevo. Por lo demás, su comportamiento es similar al del comando cp.

rm: borra archivos

Por medio del comando rm se pueden eliminar archivos. Hay que tener cuidado, aquí no existe

una papelera de reciclaje. Lo que se borra se pierde, y no se puede recuperar de ninguna forma.

El formato del comando rm es el siguiente:

rm [-friv] nombre

Las opciones funcionan de manera similar a cómo lo hacen en el comando cp, pero en este caso hay que tener más cuidado con el uso de las opciones -f y r. Como ya se vió, la opción r actúa recursivamente en directorios y subidrectorios, mientras que la opción f fuerza la ejecución del comando sin ningún tipo de pregunta. Imagínese lo que podría provocar un comando rm rf en el directorio raíz. ¡Se perdería toda la información del sistema sin posibilidad de recuperarla!

find: encuentra archivos

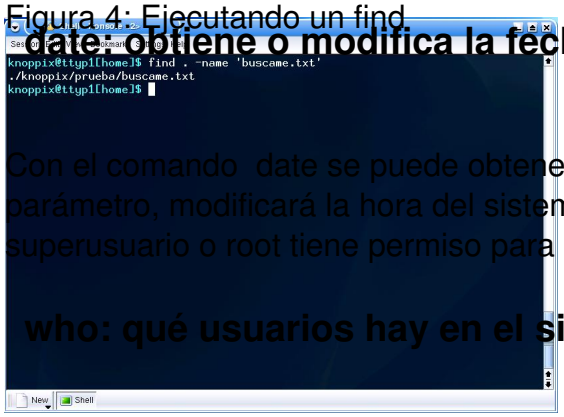
El comando find localiza los archivos pasados como parámetros. Resulta de mucha utlidad en el caso de querer saber en qué directorio se encuentro un archivo determinado. Es importante destacar que el comando sólo buscará en los directorios en los que el usuario tenga permiso de lectura.. Existen diversas maneras de utilizar el comando find, que se entenderán mejor usando ejemplos:

· find . name main.cpp encuentra todos los archivos llamados main.cpp a partir del directorio actual.

· find . size 60k busca los archivos que ocupan 60 kilobytes a partir del directorio actual. Si se utiliza find . size 30c, buscará los archivos que ocupen 30 bytes.

Se puede observar en la figura el resultado de una búsqueda en el sistema.

Figura 4: Ejecutando un find



date: obtiene o modifica la fecha actual del sistema

Con el comando date se puede obtener la fecha actual. Si se le pasa una hora como parámetro, modificará la hora del sistema (sólo un usuario especial conocido como superusuario o root tiene permiso para cambiar la hora del sistema).

who: qué usuarios hay en el sistema

El comando who muestra por pantalla los usuarios que están conectados en el sistema. Ejecutando el comando de la forma who am i, devuelve el login del usuario que lo ha ejecutado.

Metacaracteres

En todos los comandos de gestión de ficheros sería muy interesante trabajar con grupos de archivos. Por ejemplo, puede ser muy interesante borrar todos los ficheros temporales con un solo comando, o mover todos los ficheros de texto a un directorio Documentos, etc. Para trabajar con grupos de fichero, el shell permite el uso de caracteres especiales llamados comodines que permitirán la creación de patrones o plantillas para ajustar nombres de ficheros. Por ejemplo, el patrón mon* identifica todos los archivos que empiezan por mon.

Los comodines que podemos usar son:

*

Puede ser sustituido por cualquier cadena de caracteres. De este modo, *pan significa

cualquier cadena de caracteres que termine en pan.

?

Se sustituye por cualquier carácter. Es decir, la expresión se reemplaza por cualquier nombre que en esa posición tengan cualquier carácter y el resto coincidan con los que hemos escrito.

[]

Encerrado entre corchetes podremos seleccionar un conjunto de caracteres que deben concordar con el pedido. Podemos especificar además un rango usando el guión -.

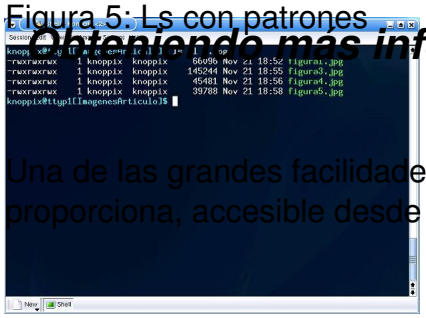
Por ejemplo, [mb]* representa todos los archivos cuyo nombre comience por m o por b.

[^]

Representa todos los caracteres excepto el indicado.

Por ejemplo, [^0-9] representa todos los archivos cuyo nombre no empiece con un numero.

Figura 5: Ls con patrones



Obteniendo más información

Una de las grandes facilidades que proporciona Linux a sus usuarios es el manual en línea que proporciona, accesible desde el comando man. El formato del comando man es el siguiente:

man [sección] [-aK] nombre

El comando man muestra en la consola una ayuda sobre el comando pedido, permitiendo avanzar y retroceder con la flechas para repasar o adelantar a la zona de interés.

La ayuda está dividida en secciones. Cada una de las secciones está dedicada a un grupo de utilidades del sistema. Las diferentes secciones se muestran en la siguiente tabla.

Sección

Tabla

1

Comandos del usuario

2

Llamadas al sistema

3

Bibliotecas

5

Formatos de ficheros

6

Juegos

7

Miscelánea

8

Administración del sistema

N

Nuevos elementos

Cuando se ejecuta el comando man usando tan sólo la información a buscar, sin parámetros, se muestra el contenido de la primera sección en la que aparezca la ayuda. Si se quiere especificar una sección en concreto, se deberá especificar su número.

La opción K permite especificar una cadena de búsqueda que el comando man buscará por todas las páginas de ayuda hasta encontrarla. De esta manera, se puede hacer una consulta cuando se sabe qué se quiere hacer, pero no se sabe el cómo. Por ello, el comando man se convierte en un aliado perfecto para el usuario de Linux.

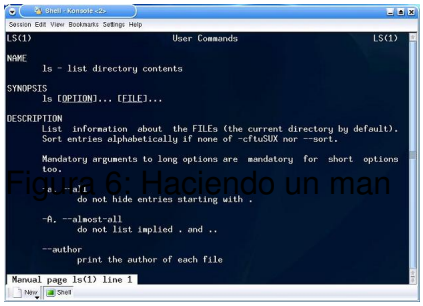


Figura 6: Haciendo un man

Otra fuente de información fundamental a la hora de trabajar con Linux son los documentos de ayuda que se incluyen con el propio sistema operativo. Casi todas las aplicaciones tendrán documentos de información que podremos localizar en el directorio /usr/share/doc.

Además de estos documentos de ayuda, existen otros con información precisa sobre cómo hacer determinadas cosas en el sistema, como configurar la tarjeta de sonido o la tarjeta de red. Estos documentos, conocidos como HOWTO, se encuentran localizados en el directorio /usr/share/doc/HOWTO.

Entendiendo el sistema de archivos

Si en cualquier sistema operativo los archivos son importantes, en Linux se convierten en esenciales. Linux trata absolutamente todo como si fueran archivos, no sólo los documentos de texto, sino los directorios, dispositivos, red, etc.

Todos los archivos están organizados en directorios ubicados a partir del directorio raíz /. Gracias a esto, el usuario no necesita saber dónde está localizado físicamente un archivo. Cuando se ejecuta ls / se mostrarán todos los archivos, algunos de los cuales estarán localizados en el servidor, otros en una máquina de nuestra red, e incluso otros en un servidor remoto, siendo esto transparente para el usuario.

Cualquier estructura jerárquica de directorios y archivos recibe el nombre de sistema de archivos (*file system*). En un sistema Linux pueden coexistir varios sistemas de archivos. Estos se *montan* en el directorio raíz del sistema y adquieren la apariencia de subdirectorios como pueden ser /usr o /home. Cada uno de estos sistema de archivos pueden estar montados en diferentes dispositivos, y de hecho es bastante común tener los sistemas de archivos importantes localizados en diferentes dispositivos, para evitar la perdida de datos importantes.

Cuando queremos añadir un dispositivo nuevo al sistema, debemos montar este dispositivo. Existe un comando que nos facilita el manejo de los distintos sistemas de archivos que tenemos en el sistema: el comando mount. La salida de este comando nos lista la identificación del sistema de archivos, el lugar del árbol de directorios en el que está montado y el tipo de sistema de archivos que constituye, además de los flags con los que se ha montado.

Por ejemplo, para poder utilizar un disquete se debe montar antes. Ejecutando

mount /floppy

se montará en el directorio /floppy el disquete y se podrá acceder a él. En este caso, el sistema operativo conoce el tipo concreto de sistema de ficheros que debe montar, y no hace falta

añadir más opciones al comando.

Una vez utilizado el disquete, y cuando ya no queramos usarlo, habrá que desmontarlo. Es muy importante hacer esto por la forma en la que trabaja un sistema operativo. Cuando se están haciendo modificaciones en un archivo, éstas no se salvan inmediatamente, porque el acceso a disco es lento, y por eso el sistema operativo realiza los cambios en los momentos libres de los que dispone. Por ello si no se desmonta un dispositivo, puede ocurrir que se pierdan algunos de los cambios que se han realizado. Al desmontar un sistema de archivos se fuerza la escritura de datos en el dispositivo, permitiendo retirar el dispositivo con total seguridad. Para desmontar la unidad de disquetes, se ejecutará el comando

umount /floppy

Otro comando muy útil a la hora de trabajar con sistemas de archivos es df. Este comando muestra los sistemas de ficheros que hay instalados en el sistema. Además indica el espacio que está siendo utilizado y el que queda aún libre para cada uno de los sistemas de ficheros.

El árbol de directorios

Los sistemas Linux, y en general todos los sistemas Unix, siguen un orden básico a la hora de ordenar la información dentro de los directorios. El árbol de directorios que casi todas las distribuciones utilizan, colgando del raíz /, es el siguiente:

/bin

Contiene las utilidades básicas del sistema, por ejemplo los comandos comentados en la

primera parte del artículo.

/dev

Contiene los archivos que representan a

dispositivos. Son archivos necesarios para el funcionamiento de Linux, y están relacionados

con periféricos de la máquina. Como nota curiosa,

el dispositivo reprentado por /dev/null es una

especie de agujero negro: todo lo que se envíe

hacia él desaparece.

/etc

En este directorio se almacenan los archivos de configuración del sistema y de las aplicaciones instalac

inicio que se ejecutan cuando arranca la máquina.

En principio no debería haber ningún fichero de configuración fuera de este directorio. Al ser un director

superusuario tiene permiso de escritura en este directorio.

/home

En este directorio se encuentran los directorios principales de los usuarios. Generalmente cada usuario

/home . Por ejemplo el usuario nacho tendría su

cuenta localizada en /home/nacho . Para evitar pe

suele montar en un dispositivo aparte y se le suele hacer un backup periódico.

/lib

Contiene las librerías necesarias para que se

puedan ejecutar los comandos que se encuentran

en /bin , así como para

las funciones de librería del lenguaje C se

encuentran en este directorio.

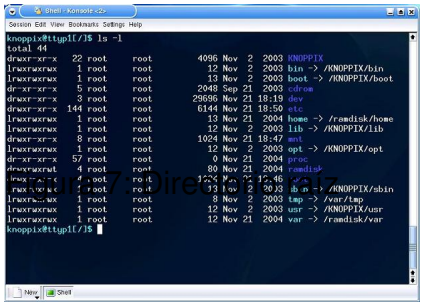
/usr

Tras el sistema raíz, este es el sistema de ficheros más importante. Contiene todos los datos y programas que se utilizan en una distribución Linux. Por ello, este directorio se divide a su vez en una jerarquía muy parecida a la del directorio raíz.

/var

Aquí se almacenan todos aquellos ficheros que se consideran variables, como algunos ficheros de regis

/tmp

Ficheros temporales.

Jose Ignacio López-k idatzia
Asteazkena, 2005(e)ko urria(r)en 19-(e)an 02:10etan