EL SHELL Y ELEMENTOS DE ADMINISTRACIÓN

CONTENIDO

CONSOLA Y CONSOLAS VIRTUALES

Una de las caracteristicas propias de Linux es que es un S.O. multiusuario. Esto permite que varios usuarios puedan estar usando simultáneamente el sistema a través de lo que se denomina consola virtual o **VCs**. Como ya se dijo, el sistema que sirve a los demás usuarios se llama host.

Una consola se puede abrir para el usuario root o para cualquier otro usuario. Para hacer esto, lo único que se debe hacer es presionar **Alt+F2**, y aparecerá:

login:

Para ingresar, se debe digitar el login o nombre de usuario y posteriormente el password o contraseña.

El límite de consolas que se pueden utilizar lo pone, en un principio, la cantidad de teclas de funcion (F1-F12). Entonces para cambiar entre consolas solo tienes que combinar las teclas **Alt+Fx** *donde **x** es un numero 1-12.*

Puede ser util hacer uso de varias consolas al mismo tiempo, a pesar de ser una sola persona la que utiliza la máquina. Se puede correr una aplicación en una y la misma en otra, o directamente correr dos distintas en diferentes consolas.

EL SHELL

El SHELL es el encargado de establecer una comunicación entre el núcleo (kernel) y el usuario del Sistema Linux ,o sea, que es una interface con nuestro Sistema Operativo. Entonces gracias a él podremos dar las ordenes necesarias para que nuestro sistema informático realice las tareas que necesitamos.

Como hemos mencionado anteriormente en numerosas ocasiones, UNIX es un sistema operativo multitarea y multiusuario. La multitarea es muy útil, y una vez la haya probado, la usará continuamente. Muchas de las características que se

tratarán en esta sección son proporcionadas por el intérprete de comandos. Hay que tener cuidado en no confundir UNIX (el sistema operativo) con el intérprete de comandos. El intérprete de comandos proporciona la funcionalidad sobre el UNIX.

El intérprete de comandos no es sólo un intérprete interactivo de los comandos que tecleamos, es también un potente lenguaje de programación, el cual permite escribir guiones, que permiten juntar varias ordenes en un archivo. El uso de los guiones (scripts) del intérprete de comandos es una herramienta muy potente que le permitirá automatizar e incrementar el uso de UNIX.

TIPOS DE INTÉRPRETES DE COMANDOS

Hay varios tipos de intérpretes de comandos en el mundo UNIX. Los dos más importantes son el **"Bourne shell"** y el **"C shell"**.

El intérprete de comandos Bourne, usa una sintaxis de comandos como la usada en los primeros sistemas UNIX, como el System III. El nombre del interprete Bourne en la mayoría de los UNIX es /bin/sh (donde sh viene de "shell", interprete de comandos en inglés).

El intérprete C usa una sintaxis diferente, a veces parecida a la del lenguaje de programación C, y en la mayoría de los sistemas UNIX se encuentra como /bin/csh.

Bajo Linux hay algunas diferencias en los intérpretes de comandos disponibles. Dos de los más usados son el **"Bourne Again Shell"** o **"Bash"**(/bin/bash) y **Tcsh** (/bin/tcsh). Bash es un equivalente al Bourne con muchas características avanzadas de la C shell. Como Bash es un super-conjunto de la sintaxis del Bourne, cualquier guión escrito para el interprete de comandos Bourne standard funcionará en Bash. Para los que prefieren el uso del interprete de comandos C, Linux tiene el Tcsh, que es una version extendida del C original.

El tipo de intérprete de comandos que decida usar es puramente una cuestion de gustos. Algunas personas prefieren la sintaxis del Bourne con las características avanzadas que proporciona Bash, y otros prefieren el más estructurado intérprete de comandos C. En lo que respecta a los comandos usuales como cp, ls..etc, es indiferente el tipo de intérprete de comandos usado, la sintaxis es la misma. Solo, cuando se escriben scripts para el intérprete de comandos, o se usan características avanzadas aparecen las diferencias entre los diferentes intérpretes de comandos.

CARACTERÍSTICAS DEL SHELL

Flujos

El shell le otorga a cada programa en ejecución 3 flujos: Entrada Estándar (teclado, archivo, dispositivo E/S), Salida Estándar (terminal, archivo, dispositivo E/S) y Error Estándar (terminal, archivo, dispositivo E/S).

Ejecución del Shell

Una vez que usted se ha registrado en la consola virtual, el sistema carga el intérprete de comandos en memoria. Éste lee primero el archivo /etc/profile que contiene la configuración común del entorno para todo el sistema y sus usuarios, luego lee el archivo .bash_profile ubicado en el directorio home del usuario recien ingresado.

¿Cómo se escriben los comandos?

La sintaxis para escribir un comando en el prompt de bash es:

comando [parámetro1] [parametro2] ...

Capitalización de las Letras

Recordemos que siempre los comandos y parámetros en GNU/Linux son **"case sensitive"**, que quiere decir que "algo" no es igual a "ALGO" u "AlGo".

Historial de Comandos

El archivo .bash_history es un archivo de historial con todos los comandos utilizados por el usuario y como es lógico se encuentra en el directorio home de dicho usuario. Éste archivo aloja una cantidad de comandos definida previamente. Se accede al historial por medio de las teclas **FLECHA ARRIBA** y **FLECHA ABAJO**. Si necesita escribir una cantidad excesivamente grande de parámetros y ha llegado al final de la linea de comandos, puede hacer uso del simbolo "\" seguido de un "enter" para poder continuar con la escritura de dichos parametros en la linea siguiente y también sirve para ver que escribimos anteriormente.

Completado de Linea

Se puede hacer que el shell complete la linea de comandos cuando se introduzcan las primeras letras y se presione la tecla TAB. También se puede utilizar esta propiedad para expandir la ruta a un directorio determinado.
Ejemplo: cd /ho (presionamos TAB) y el shell nos devolverá /home/

Comillas

Las comillas controlan la forma en que el shell expandirá las ordenes que estén encerradas entre ellas.
Existen tres tipos de comillas, las dobles ("), las sencillas (') y las inversas (`).

- Las comillas inversas indican al shell que tendrá que reemplazar lo que está encerrado entre ellas con su resultado.
- Las comillas sencillas (') le dicen al sistema que no hagan ninguna expansión.
- Las comillas dobles tienen casi la misma funcionalidad que las simples pero con la salvedad de que lo que se incluya dentro de estas pasará a ser como una cadena simple de caracteres a excepción de las comillas inversas (`), el signo dólar ($), la diagonal (\) y las mismas comillas dobles (").

OPCIONES DEL SHELL

El shell proporciona varias opciones a la hora de digitar los comandos, tales como:

SIMBOLO	SIGNIFICADO
*	Metacaracteres para usar varios archivos o directorios.
?	Reemplaza un solo caracter
&	Comandos en background, ej: find > archivo &
;	Ejecutar varios comandos, ej: $ ls ; pine
|	Para dirigir salidas standard.
>	Redireccionar la salida estandar a un archivo.
>>	Lo mismo que el anterior pero no sobreescribe.
<	Para tomar los datos de un archivo.

VARIABLES DE ENTORNO

Independientemente del shell que se esté usando, se contará con lo que se denomina **Variable de Entorno**. Podemos decir que estas variables configuran su sistema , para hacerlo un poco más amigable.

Las Variables de Entorno son como las que encontramos en cualquier lenguaje de programación; además pueden ser accedidas desde los scripts que usted cree. Para distinguir las variable de entorno de los comandos, las primeras se ponen en letra mayúscula.

Cuando uno entra al sistema , la mayoría de estas variables ya están asignadas.

Para ver las variables de entorno de tu cuenta, y algunas generales , utiliza el comando : **$env**

El resultado de este comando será algo como esto:

```
USERNAME=
ENV=/home1/temporales/grodaz/.bashrc
HISTSIZE=1000
HOSTNAME=antares
LOGNAME=grodaz
HISTFILESIZE=1000
MAIL=/var/spool/mail/grodaz
WWW_HOME=.bookmarks.html
TERM=vt100
HOSTTYPE=i386
PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home1/
        temporales/grodaz/bin
HOME=/home1/temporales/grodaz
SHELL=/bin/bash
PS1=[\u@\h \W]\$
USER=grodaz
```

OSTYPE=Linux
SHLVL=1
_=/usr/bin/env

Esto lo veremos en el caso de un usuario sin privilegios. Al Usuario Root, le aparecerá mas o menos lo mismo. Todas estas variables de entorno, cumplen una funcion definida en su cuenta, tenga cuidado al modificarlas, y cuando lo haga, con el fin de evitar confusiones, haga una nota del cambio que efectuó.

También se pueden definir nuevas variables de entorno, de la siguiente forma:

export NOMBRE=valor

Algunas de las variables de entorno más utilizadas son:

- **HOME** Directorio casa.
- **HOSTNAME** Nombre del Servidor.
- **MAIL** Archivo de correo.
- **PATH** Lista de directorios donde busca los programas.
- **PS1** Prompt.

La variable de entorno PATH

También llamada **"ruta de búsqueda"**. Esta variable contiene un grupo de directorios predefinidos en los cuales el shell buscará el comando o programa a ejecutar. Esto ahorra tiempo ya que el sistema no tendrá que buscar en todos los directorios el programa a ejecutar. Por esto el sistema, en caso de que el directorio no figure en el PATH no podrá ejecutar el programa hasta que le demos la ruta exacta en donde se encuentre.

EL ARCHIVO .bash_profile

Este archivo es el equivalente al autoexec.bat MS-DOS. Cuando se carga la cuenta, automáticamente se ejecutan todos los comandos que se hallan en él. Precisamente por eso, este archivo es usado para configurar las variables de entorno a nivel de usuario (incluido el root) .

El archivo .bash_profile puede ser modificado únicamente por el usuario dueño del archivo y por root**.**

EL COMANDO ALIAS

Por medio de este comando podemos referenciar a un comando o grupo de comandos con un nombre corto y fácil de recordar. Los alias pertenecen a cada usuario, pudiéndose configurar de la forma que más convenga.

Veamos un Ejemplo: Al escribir el comando "***date***" sin parámetros obtenemos una sálida similar a esta:

lun jun 4 22:37:27 ART 2001

Si sólo queremos ver la hora, podemos crear un **alias** al comando **date** con el parámetro correspondiente.

[max@máquina /max]$alias hora='date +%r'

Así, tendríamos un alias *hora*, para el comando *date*. Así, cuando digitemos hora en el prompt, aparecerá así:

[max@máquina /max]$hora

22:38:27

Ahora bien, esto funcionará sólo mientras permanezcamos en el sistema, ya que estos datos son cargados en memoria, y al salir del sistema se perderían. Si agregamos el alias en el archivo .bash_profile del usuario que lo necesite, estará disponible cada vez que ingrese al mismo.

Si queremos ver un listado de los alias para el usuario digitamos ***alias*** sin parámetros.

COMANDOS BÁSICOS

Antes de conocer algunos de los comandos disponibles en el Shell, se debe tener muy claro que tareas se van a realizar, es decir, se debe diferenciar entre tareas de administración y uso habitual del sistema.

Los comandos se invocan por su nombre y se le pasan opciones o parámetros por medio del signo menos **"-"** seguido de una o mas letras. Por ejemplo **ls -la**.

Además de estas opciones muchas veces fijándonos en las páginas del manual de un determinado comando nos encontramos con opciones de palabra completa.Dichas opciones se indican con doble signo "-" y generalmente existe su correspondencia con la forma de pasar parámetros a un comando, como vimos anteriormente.

Ejemplo: "cat --number" equivale a "cat -n".

También debes dejar un espacio entre el comando y el primer parámetro.

COMANDOS BÁSICOS DE USUARIO

Normalmente, como usuario, se creará el hábito de crear, leer y borrar archivos en el directorio home/usuario, ya que fuera de este las posibilidades se ven reducidas. A continuación se describen algunos comandos sencillos que pueden ser útiles para familiarizarse con los comandos del sistema.

COMANDO	FUNCIONALIDAD
date	Muestra por pantalla el día y la hora.
cal 2000	Muestra el calendario del año 2000.
cal 05 2003	Muestra el calendario de mayo de 2003.
who	Indica qué usuarios tiene el ordenador en ese momento, en qué terminal están y desde qué hora.
whoami	Indica cuál es la terminal y la sesión en la que se está trabajando.
man comando	Todos los manuales de Linux están dentro del propio sistema operativo, y este comando permite acceder a la información correspondiente al comando que se digite a su lado.
clear	Este comando limpia la consola

A continuación se describe la forma de uso y funcionamiento de los comandos de usuario más importantes en el shell de Linux.

Listado del Contenido de Directorios: Comando ls

Una de las acciones más habituales a la hora de trabajar es mostrar el contenido de un directorio, el shell incluye un programa con este fin: ls.

ls	Muestra los nombres de los ficheros y subdirectorios contenidos en el directorio en el que se está. Sólo se obtienen los nombres de los ficheros, sin ninguna otra información.
ls -a	Muestra todos los ficheros incluyendo algunos que ordinariamente están ocultos para el usuario (aquellos que comienzan por un punto).
ls -l	Esta es la opción de lista larga: muestra toda la información de cada fichero incluyendo: protecciones, tamaño y fecha de creación o del último cambio introducido.
ls -c	Muestra ordenando por día y hora de creación.
ls -t	Muestra ordenando por día y hora de modificación.
ls -r	Muestra el directorio y lo ordena en orden inverso.
ls subdir	Muestra el contenido del subdirectorio subdir.
ls -l filename	Muestra toda la información sobre el fichero.
ls --color	Muestra el contenido del directorio coloreado.

Las opciones anteriores pueden combinarse. Por ejemplo: ls -cr Muestra el directorio ordenando inversamente por fechas.

El comando ls admite los caracteres de sustitución o metacarácteres (*) y (?). El carácter * representa cualquier conjunto o secuencia de caracteres. El carácter ? representa cualquier carácter, pero sólo uno. Por ejemplo, si digitamos ***ls *.gif***, el resultado mostrado serán todos los archivos cuya extensión sea .gif, por ejemplo dibujo.gif, cuadro.gif, etc.
De manera análoga si digitamos ***ls file?***, el shell mostrará todos los archivos cuyos nombres empiecen por file y tengan un nombre de cinco caracteres, por ejemplo: file1, file2, filea, etc.

Creación de Subdirectorios: Comando mkdir

El comando **mkdir** (*make directory*) permite a cada usuario crear un nuevo subdirectorio, la forma de utilizar este comando sería: *mkdir subdir1*, donde subdir sería el nombre del directorio que se va a crear.

Borrado de Subdirectorios: Comando rmdir

Este comando borra uno o más directorios del sistema (*remove directory*), siempre que estos subdirectorios estén vacíos. Por ejemplo: *rmdir subdir1* donde subdir es el nombre del directorio que se va a eliminar.

Cambio de directorio: Comando cd

Este comando permite cambiar de directorio a partir del directorio actual de trabajo. Por ejemplo, ***cd /home/Pedro***, en este ejemplo pasamos del directorio actual de trabajo al nuevo directorio /home/Pedro, que será desde ahora nuestro nuevo directorio.
cd modelo nos traslada al subdirectorio modelo. Dicho subdirectorio deberá existir como subdirectorio en el directorio actual.

Otra forma de utilizar el comando cd es: ***cd ..*** Retrocedemos un nivel en la jerarquía de directorios. Por ejemplo, si estamos en /home/Pedro y usamos este comando, pasaremos al escalafón inmediatamente superior de la jerarquía de directorios, en este caso a /home.

Situación actual: Comando pwd

El comando pwd (*print working directory*) visualiza o imprime la ruta del directorio en el que nos encontramos en este momento. Este comando es uno de los pocos que no tiene opciones y se utiliza escribiendo simplemente **pwd**.

Acceso a Unidades de Disco: Comando mount

Linux a diferencia de Windows no utiliza letras ("a:", "c:", "d:", ...) para acceder a las distintas unidades de disco de un ordenador. En Linux para acceder al contenido de una unidad de disco o de un CD-ROM este tiene que haber sido previamente ***"montado"***. El montaje se realiza mediante el **comando mount**, con

lo que el contenido de la unidad se pone a disposición del usuario en el directorio de Linux que se elija.

Por ejemplo para acceder al CD-ROM se teclearía el siguiente comando:

mount -t iso9660 /dev/cdrom /mnt/cdrom

donde -t iso9660 indica el tipo de sistema que usa la unidad de disco para guardar los archivos (las más usuales son: iso9660 en el caso de un CD-ROM, vfat en el caso de Windows, y ext2 en el caso de Linux), /dev/cdrom indica el dispositivo que se va a montar. Todos los dispositivos están representados por un archivo del directorio /dev, por ejemplo en el caso de un disquete será /dev/fd0, por último /mnt/cdrom es el directorio en el que se pondrá a disposición del usuario el contenido del CD-ROM. *Para montar disquetes se suele utilizar el directorio /mnt/floppy.*
De todas formas usted puede crear un directorio vacío con el nombre que elija para montar las unidades de disco que desee donde desee.

Cuando se haya dejado de usar ese disco, deberá ***"desmontarlo"*** mediante el **comando umount** antes de sacar el disquete o el CD-ROM. En este último caso se debe escribir:

umount /mnt/cdrom

Para utilizar el comando mount de la forma anterior hace falta ser administrador o root. Para que un usuario común pueda utilizar disquetes, CD-ROM, etc. hay que editar el archivo /etc/fstab, por ejemplo para que cualquier usuario pueda acceder a un disquete habrá que indicar la siguiente línea:

/dev/fd0 /mnt/floppy vfat user,noauto 0 0

También habrá que asegurarse de que el directorio /mnt/floppy sea accesible por todos los usuarios. Una vez seguidos los pasos anteriores cualquier usuario podrá "montar" un disquete escribiendo la siguiente línea en el prompt del shell:

mount /mnt/floppy

Al igual que antes el usuario deberá ejecutar el comando umount /mnt/floppy antes de sacar el disquete.

En la actualidad existen distribuciones (p. ej. Linux Mandrake) que realizan este proceso de forma automática por lo que las unidades de disquete y CD-ROM quedan accesibles a todos los usuarios de una forma sencilla, empleando los comandos:
mount /mnt/floppy, umount /mnt/floppy, siempre que /mnt/floppy sea la ruta adecuada.

Copia de Archivos: Comando cp

Este comando hace una copia de file1 con el nombre file2, tiene la siguiente forma:

cp file1 file2

Si file2 no existía, lo crea con los mismos atributos de file1. Si file2 existía antes, su contenido queda destruido y es sustituido por el de file1. El archivo file2 estará en el mismo directorio que file1. Tanto file1 como file2 indican el nombre de un archivo, que puede incluir la ruta al mismo si alguno de ellos no se encuentra en el directorio
actual.

Traslado y cambio de nombre de Archivos: Comando mv

El comando mv realiza la misma función que el comando cp, pero además destruye el archivo original. Este comando tiene la siguiente sintaxis:

mv file1 file2

Este comando traslada el contenido de file1 a file2; esto quiere decir que se le ha cambiado el nombre al archivo file1 y se le ha llamado file2.De igual forma, la instrucción:

mv file1 file2 namedir

Traslada uno o más archivos (file1, file2,...) al directorio namedir conservándoles el nombre.

Borrado de Archivos: Comando rm

Este comando tiene las siguientes formas,

rm file1 file2

Este comando elimina uno o más archivos de un directorio en el cual tengamos permiso de escritura.
Con este comando resulta muy fácil borrar archivos inútiles, y desgraciadamente, también los útiles.

Es conveniente y casi imprescindible emplear lo opción -i, de la forma siguiente:

rm -i file1 file2

Con esta opción, Linux pedirá confirmación para borrar cada archivo de la lista, de si realmente se desea su destrucción o no. Se recomienda usar siempre este comando con esta opción para evitar el borrado de archivos útiles.

En este comando se pueden utilizar los caracteres de sustitución (* y ?), como por ejemplo:

*rm arch**

que borraría todos los archivos del directorio actual que comiencen por arch. Así, el comando *rm* * borrará todos los archivos del directorio actual, mientras que *rm -i* *
realiza una labor análoga, pero con previa confirmación.

Características de un archivo: Comando file

Este comando realiza una serie de comprobaciones en un fichero para tratar de clasificarlo. Su formato es:

file archivo

Luego de su ejecución este comando muestra el tipo del archivo e información al respecto del mismo.

Escribir primeras lineas de un archivo: Comando head

La forma de usar este comando es como sigue:

head -7 filename

Lo que indicamos al shell con esta instrucción, es que escriba las 7 primeras líneas del archivo filename.

Visualización y concatenación de un archivo: Comando cat

Este comando permite visualizar el contenido de uno o más archivos de forma no formateada.
También permite copiar uno o más ficheros como apéndice de otro ya existente, es dicir, permite realizar la concatenación entre archivos. Algunas formas de utilizar este comando son las siguientes:

cat filename

Saca por pantalla el contenido del fichero filename.

cat file1 file2...

Saca por pantalla, secuencialmente y según el orden especificado, el contenido de los ficheros indicados.

cat file1 file2 >file3

El contenido de los ficheros file1 y file2 es almacenado en file3.

cat file1 file2 >>file3

El contenido de file1 y file2 es añadido al final del archivo file3.

cat >file1

Acepta lo que se introduce por el teclado y lo almacena en file1 (se crea file1). Para terminar se emplea <ctrl>d.

Visualización de archivos pantalla a pantalla: Comandos more y less

Estos comandos permiten visualizar un archivo pantalla a pantalla. El número de líneas por pantalla es de 23 líneas de texto y una última línea de mensajes, donde aparecerá la palabra more. Cuando se pulsa la barra espaciadora (el espacio en blanco), se visualizará la siguiente pantalla. Para salir de este comando (terminar la visualización) se pulsa *<ctrl>d o q*. Por ejemplo: more file

El comando less es muy similar al anterior pero permite el desplazamiento a lo largo del texto empleando las teclas de cursores pudiendo desplazarse hacia arriba o abajo de un archivo.

Búsqueda dentro de archivos: Comandos grep, fgrep y egrep

El **comando grep** localiza una palabra, clave o frase en un conjunto de directorios, indicando en cuáles de ellos la ha encontrado. Este comando busca archivo por archivo, uno por uno, imprimiendo aquellas líneas que contienen el conjunto de caracteres buscado. Si el conjunto de caracteres a buscar está compuesto por dos o más palabras separadas por un espacio, se colocará el conjunto de caracteres entre apóstrofes (').

El formato de este comando es el siguiente:

grep 'muchaspalabras' file1 file2 file3

siendo 'muchaspalabras' la secuencia de caracteres a buscar, y file1, file2, y file3 los archivos donde se debe buscar.

Este comando permite seleccionar, entre todas las líneas de uno o más archivos, aquellas que contienen un motivo que satisface una expresión regular determinada.
grep [-opcion] expresión_regular [referencia...]

Las opciones principales son:

c	Lo único que se hace es escribir el número de las líneas que satisfacen la condición.
i	No se distinguen mayúsculas y minúsculas.
l	Se escriben los nombres de los ficheros que contienen líneas buscadas.
n	Cada línea es precedida por su número en el fichero.

s	No se vuelcan los mensajes que indican que un fichero no se puede abrir.
v	Se muestran sólo las líneas que no satisfacen el criterio de selección.

Difusión de programas: Comandos tar y gzip

Tanto el comando **tar** como **gzip** son ampliamente empleados para la difusión de programas y archivos en Linux. El primero de ellos agrupa varios archivos en uno solo, mientras que el segundo los comprime. En conjunto estos dos programas actúan de forma muy similar a programas como Winzip. Para crear un nuevo archivo se emplea:

tar –cvf nombre_archivo.tar fichero1 fichero2 …

donde fichero1, fichero2 etc. son los ficheros que se van a añadir al archivo tar. Si se desea extraer los ficheros se emplea:

tar –xpvf nombre_archivo.tar fichero1 …

Al contrario que tar, que agrupa varios ficheros en uno, **gzip** comprime un único archivo con lo que la información se mantiene pero se reduce el tamaño del mismo. El uso de gzip es como sigue:

gzip filename

con lo que se comprime filename (que es borrado) y se crea un archivo con nombre filename.gz.

Si lo que se desea es descomprimir un archivo se emplea entonces:

gzip –d filename.gz

recuperando el archivo inicial. Como se mencionó al principio es típico emplear tar y gzip de forma consecutiva, para obtener archivos con extensión tar.gz o tgz que contienen varios archivos de forma comprimida (similar a un arhcivo .zip). El comando tar incluye la opción z para estos archivos, de forma que para extraer los archivos que contiene:

tar –zxf fichero.tar.gz

Para terminar con los comandos más utilizados por los usuarios, a continuación se presenta una tabla con las funciones de otros comandos adicionales.

COMANDO	FUNCIONALIDAD
passwd	Cambiar password.
diff	Encuentra la diferencia entre archivos.
tail	Muestra las ultimas lineas de un archivo.
echo	Escribe mensajes en la salida standard.
sort	Ordena las lineas de un texto.

cut	Corta secciones de una linea.
od	Convierte los archivos a forma octal u otras.
paste	Une lineas de diferentes archivos.
uniq	Remueve lineas repetidas.
wc	Cuenta lineas, palabras y caracteres.
finger	mirar los usuarios
w	Que están haciendo los usuarios
write	Escribirle un mensaje a un usuario
wall	Ponerle un mensaje a todos los usuarios conectados
talk	Conversación iteractiva con otro usuario
mesg	Habilita que le puedan o no escribir mensajes.
pico	Editor de texto
mdir	Muestra el contenido de un directorio de un MSDOS
mcopy	Copia de una o hacia un MSDOS.
mformat	Formate en MSDOS
mcd	Cambia de directorio en MSDOS
mdel	Borrar un archivo en MSDOS
mail	Manejador de correo electrónico
pine	Manejador de correo electrónico
elm	Manejador de correo electrónico
telnet	Programa para conexiones telnet
ftp	Programa para conexiones ftp

COMANDOS DE ADMINISTRACIÓN

El usuario root del sistema operativo Linux, tiene acceso a todos los comandos anteriormente mencionados, y además tiene acceso a otros comandos adicionales para fines administrativos.

A continuación se mencionarán los más importantes.

Cambio de modo de los archivos: comandos chmod, chown y chgrp

Los permisos de cada archivo se pueden ver con el comando ls -l. Para cambiar los permisos de un archivo se emplea el **comando chmod**, que tiene el formato siguiente:

chmod [quien] oper permiso files

[quien] --> Indica a quien afecta el permiso que se desea cambiar. Es una combinación cualquiera de las letras **u** para el usuario, **g** para el grupo del usuario, o para los otros usuarios, y **a** para todos los anteriores. Si no se da el *quien*, el sistema supone *a*.

oper --> Indica la operación que se desea hacer con el permiso. Para dar un permiso se pondrá un **+**, y para quitarlo se pondrá un **-**.

permiso --> Indica el permiso que se quiere dar o quitar. Será una combinación cualquiera de las letras: r,w,x,s.

files --> Nombres de los archivos cuyos modos de acceso se quieren cambiar.

Por ejemplo, para quitar el permiso de lectura a los usuarios de un archivo la forma correcta de usar el comando sería:

chmod a -r filename.txt

Por otra parte, el **comando chown** se emplea para cambiar de propietario *("change owner")* a un determinado conjunto de archivos. Este comando sólo lo puede emplear el actual propietario de los mismos. Los nombres de propietario que admite Linux son los nombres de usuario, que están almacenados en el archivo */etc/passwd.*

La forma general del comando chown es la siguiente:

chown newowner file1 file2 ...

Análogamente, el grupo al que pertenece un archivo puede ser cambiado con el **comando chgrp**, que tiene una forma general similar a la de chown:

chgrp newgroup file1 file2...

Los grupos de usuarios están almacenados en el fichero */etc/group.*

Espacio ocupado en el disco: Comandos du y df

El **comando du** permite conocer el espacio ocupado en el disco por un determinado directorio y todos los subdirectorios que cuelgan de él. Para usarlo basta simplemente colocarse en el directorio adecuado y teclear, du, éste comando da el espacio de disco utilizado en bloques. Para obtener la información en bytes se debe emplear el comando con la opción –h: *du -h*

El **comando df** por el contrario informa del espacio usado por las particiones del sistema que se encuentren montadas.

A continuación citamos una serie de comandos de administración con su respectiva funcionalidad:

COMANDO	FUNCIÓN
passwd	Cambiar los passwords a los usuarios
date	Cambiar la fecha y hora al sistema
chfn	Cambiar shell.
su	Adquirir permisos de superusuario
top	Mira el comportamiento de la CPU

kill	Mata o envía señales a los procesos.
shutdown	Sirve para bootear o apagar la máquina.
reboot	Bootea la máquina
halt	Apaga la máquina
adduser	Adiciona un usuario
deluser	Borra un usuario
free	Muestra el espacio libre de disco duro
setup	Programa para reconfigurar elementos de la instalacion.
linuxconf	Programa para el manejo de la configuracion de todo el sistema.
at	Sirve para programar tareas una vez.
ping	Mira el estado de conexión de la red
traceroute	Describe la ruta que sigue un paquete en la red.