Comandos Unix

Índice:

Pidiendo disculpas a los puristas, esta lista está estructurada de la siguiente forma:

A la izquierda figuran los comandos DOS, y a la derecha su equivalente Unix.

A continuación hay otra lista con los comandos sin equivalente DOS.

Comandos comunes:

DOS	**Unix**
cd <Directorio>	cd(nota: cd.. no sirve. Usar cd ..(Con espacio).
md <Directorio>	mkdir <Directorio>
rd <Directorio>	rmdir <Directorio>
del	rm
deltree <Directorio>	rm −R <Directorio>
dir	ls (dir funciona como ls −l −color)
copy, xcopy	cp
move, rename	mv
COMMAND.COM	bash, tcsh, zsh, etc. (En Unix hay varios shells)
win	startx
help	man <Comando>
cls	clear
pkzip/pkunzip	gunzip
doskey	Automático en Linux.
Comando /?	Comando −−help. Si no, mirar el manual.
type <Archivo>	cat <Archivo>
edit	pico, vi, emacs, etc. (El pico es el más fácil de usar)
time, date	date (Imprime la fecha y hora). date −−set la define.
cmp	cmp,diff(Imprime las diferencias entre dos archivos)

echo Cadena de caracteres	echo Cadena de caracteres (Entrecomillado)
ping	ping
tracert	traceroute
set	set

Comandos propios de Unix:

chmod <permisos> <archivo>: Cambia los permisos de acceso a un fichero. Los permisos se especifican mediante tres números. El primero simboliza los permisos del propietario del archivo, el segundo los permisos del grupo al que pertenece el archivo, y los tres últimos son los permisos para todo el mundo. Si pasamos cada uno de estos números a binario, ocuparán tres bits. Los permisos se estructuran en tres tipos: Lectura(r), Escritura(w), y ejecución(x). Por ejemplo:

chmod 764 archivo.txt (Nota: 7=111, 6=110, 4=100)

Permisos 764: en binario los tres números son (111 110 100). Esto es, da permisos rwx(111) rw−(110) r− −(100); Es decir, todos los permisos para el propietario(rwx), lectura−escritura para todos los usuarios

pertenecientes al grupo dueño del archivo(rw−) y sólo de lectura para el resto del mundo(r− −).

chown <Nombre de Usuario> <Archivo>: Cambia el usuario propietario de un archivo. Sólo root puede

ejecutar este comando.

chgrp <Nombre de Grupo> <Archivo>: Cambia el grupo efectivo de un archivo.

su: Cambia a modo administrador desde cualquier usuario. Por supuesto, pide el password.

ps: Muestra una lista de los procesos en curso, con su PID.

kill <PID>: Mata el proceso especificado por PID. Si falla, kill −9 <PID> lo mata incondicionalmente.

reboot: Reinicia el sistema.

halt: Apaga el ordenador(Si la BIOS lo soporta). Equivale a Inicio−Apagar sistema−Apagar el equipo.

lsmod: Muestra los módulos cargados en memoria.

rmmod: Descarga de memoria un módulo, pero sólo si no está siendo usado.

insmod <Nombre>: Carga en memoria un módulo.

gzip: Para instalar un archivo comprimido (Archivo.tar.gz, muy usado en internet) se usa:

gzip −cd Archivo.tar.gz | tar xvf − (Con guión al final)

mount /dev/<Dispositivo> <Directorio>: Monta una unidad de red en el sistema de archivos. Nótese que las unidades de disco, CD, etc. locales las trata como unidades de red, ubicadas en /dev/hd?. Su uso es igual a NET USE en Windows NT. Sólo root puede ejecutar este comando, a no ser que se especifique lo

contrario en el archivo /etc/fstab.

umount <Dispositivo>: Desmonta una unidad.

YaST: (¡Respetar mayúsculas/Minúsculas!) Es el programa de configuración de SuSE. Puede invocarse en cualquier momento desde el prompt, pero sólo como superusuario(root). Es algo así como el panel de control de W95. Permite instalar/Desinstalar archivos de la distribución, paquetes RPM, configurar

hardware, opciones de red, etc(Ver documento adjunto).

whereis <Archivo>: Devuelve la ubicación del archivo especificado, si existe.

find <Archivo>: Busca el archivo especificado. Necesita opciones alternativas, como el directorio donde

se debe buscar, etc Es complicado, pero muy potente.

grep <Cadena>: Busca la cadena de caracteres especificada en un texto. Cuando la encuantra imprime solamente las líneas en las que aparece la cadena. Por ejemplo: set | grep WINDOWMANAGER devolverá algo como:

WINDOWMANAGER=/opt/kde/bin/startkde

cat comandos Unix.doc | grep caca devolvería esta misma línea.

sort: Ordena un texto(lista de archivos, etc.)

more: Da control sobre el scroll de pantalla al intentar leer textos largos.

less: Es una función más completa que more.

make: Compila un módulo con sus librerías automáticamente. Requiere un archivo: Makefile.

pwd: Devuelve el directorio actual.

head: Imprime las primeras líneas de un archivo.

ln: Hace un link simbólico ó físico a un fichero. Ej.: ln −s /MisDocumentos/documento.txt texto

haría un enlace simbólico con nombre texto al archivo documento.txt.

lpr: Comando de impresión en linux.

wc: No es el retrete, no... Cuenta las palabras de un archivo.

who: Imprime una línea por cada usuario que se encuentre en el sistema.

mail: Para enviar/recibir correo a/de otros usuarios de la red, o dentro de nuestro ordenador.

Agrupamiento de mandatos:

Los shell de Unix, como el COMMAND.COM de MS−DOS permiten una serie de conexiones lógicas de unos mandatos con otros, de manera que se modifica la entrada/salida estándard de los comandos, para comunicarlos entre sí:

1.− Redirección de salida(>): Redirige la salida de un comando a un fichero.

Si hacemos: dir > fichero.txt

Se creará un archivo fichero.txt. Si miramos su contenido (cat fichero.txt) veremos que contiene exactamente lo que veríamos al ejecutar el comando dir.

2.− Redirección de entrada(<): Redirige la entrada estándard de un comando desde un fichero.

Si hacemos: grep /root < fichero.txt

De existir una línea que contenga esa cadena en el archivo antes creado, fichero.txt se imprimirá en pantalla la línea completa. Si no, no devolverá nada.

3.− Ejecución secuencial(;): Ejecuta secuencialmente una lista de comandos.

Ejemplo: mv archivo1 aux ; mv archivo2 archivo1; mv aux archivo1

4.− Redirección de salida estándard de error(>&).

Si utilizamos un comando y salen varias pantallas describiendo el error(Cosa poco usual), o si por ejemplo dejamos trabajando nuestro terminal ejecutando varias órdenes y queremos al volver mirar si todo ha sido correcto, hacemos:

comando **>&** fichero.txt

5.− Ejecución asíncrona(&): Ejecuta en paralelo el/los comandos introducidos.

Ejemplo: make programa_enorme.c & make programa_enorme2.c &

Compilará los dos programas a la vez, y además nos dejará el shell libre para poder seguir ejecutando mandatos.

6.− OR lógico(||): Se ejecutan de forma secuencial los mandatos introducidos hasta que uno de ellos devuelva un valor 0(verdadero).

Ejemplo: test −d archivo || lpr archivo

Sólo imprime el archivo si no es un directorio.

7.− AND lógico (&&): Se ejecutan de forma secuencial los mandatos introducidos hasta que uno de ellos devuelva un valor distinto de 0(falso).

Ejemplo: test −f archivo && lpr archivo

Imprime el fichero sólo si se trata de un fichero ordinario.

8.− Tuberías(pipes)(|): Se ejecutan los comandos de forma secuencial, pero redirigiéndose la salida de cada uno al siguiente comando.

Ejemplo1: dir | sort | less

Imprime la lista de los ficheros del directorio actual de trabajo ordenados alfabéticamente y línea a línea.

Ejemplo2: head −100 carta | grep Juan | sort | lpr

Imprime en orden alfabético las líneas que, estando entre las 100 primeras del archivo carta contengan la cadena de caracteres Juan.

Además el shell es una herramienta de programación muy versátil y completa. Proporciona una serie de mandatos compuestos que permiten escribir programas estructurados de forma condicional o incluso bucles. Entre estos mandatos están el if, case, until, while, for, etc. Además de la posibilidad de definir funciones de forma similar a la sintaxis de C, y variables. Esto permite hacer unos programas llamados Shell Scripts, que equivalen a los archivos .BAT de MS−DOS, pero con una sintaxis de programación mucho más completa.

Caracteres especiales:

La sintaxis de los shell de Linux permite definir los nombres de archivos con todo tipo de caracteres, pero hay una serie de caracteres especiales como el espacio, o símbolos como (; | > <), etc. que no se pueden especificar así como así en un nombre de archivo. Para proteger estos caracteres se usan los caracteres (\), () y (`).Un ejemplo:

Si queremos borrar un archivo que se llama *mio;tuyo nombre;>raro* debemos poner:

rm mio\;tuyo\ nombre\;\>raro

o bién (mas cómodo):

rm mio;tuyo nombre;>raro

El carácter tilde(~) se usa en el shell de Linux para especificar el directorio HOME de cada usuario en vez de, por ejemplo: /home/Informática/Explotación/Manuelva/

Además para cambiar a nuestro directorio HOME basta con ejecutar cd sin argumentos.

Ayuda:

La ayuda en Linux viene en forma de manuales de cada comando. Su uso es el siguiente:

man <Comando>

Son manuales algo antiguos, y tienen el inconveniente de estar en inglés, pero son muy útiles. Además prestan ayuda también a los múltiples archivos de configuración del sistema.

Variables de entorno:

En unix las variables de entorno son mas importantes que en MS−DOS. La principal razón es la variable PATH, que como todos sabemos contiene los directorios donde el shell debe buscar para encontrar cualquier comando xterno al shell. Esto es igual que en MS−DOS, pero en Linux toda la demás información que aparece con el comando **set** es usada continuamente por los diferentes programas.

Entre la información que aparece por allí, tenemos el shell(COMMAND.COM) que se usa por defecto, el nombre de usuario, el pid de nuestro propio shell, la configuración del servidor de ventanas, y un largo etcétera que será más o menos largo según los programas que usemos.

Para exportar nuevas variables primero se definen. Por ejemplo, si queremos que nuestro gestor de ventanas sea el KDE por defecto, simplemente haremos:

WINDOWMANAGER=/opt/kde/bin/startkde

Si ahora escribimos set en la línea de comandos, vemos que lo que hemos hecho no ha afectado a la variable de entorno WINDOWMANAGER. Para que el cambio tenga efecto, hay que exportarla al sistema; Esto se hace:

export WINDOWMANAGER

Si queremos que cualquiera de estas variables exista cada vez que entremos con nuestro usuario, habrá que añadirla al archivo .profile en nuestro directorio home.

Archivos importantes:

En Linux, como en MS−DOS o cualquier sistema operativo, existen una serie de archivos con información para mantener el sistema. Archivos como el AUTOEXEC.BAT o el CONFIG.SYS, el IO.SYS, etc. contienen la forma de configurar el arranque de nuestro ordenador, o hasta de algún programa, como el WIN.INI, etc...

A continuación se hace un breve resumen de estos archivos en Linux:

En cada directorio personal: **/home/Usuario/**

.**xinitrc:** Archivo de inicio del entorno de ventanas. Es propio de cada usuario.

.profile: Es el equivalente al AUTOEXEC.BAT de MS−DOS, pero con la particularidad de que en Linux este archivo es diferente para cada usuario. Se trata de un shell script, y hay que crearlo a mano, pues no se crea por defecto.

.bash_history: Es un archivo de seguridad en el que se graba cada comando que ejecuta un usuario durante varias sesiones seguidas.

En el directorio de configuración: **/etc/**

profile: Es el equivalente al AUTOEXEC.BAT; se ejecuta al arrancar.

fstab: Archivo que contiene el montaje de los dispositivos, los puntos de montaje, opciones de automontaje en el arranque, permisos, etc.

lilo.conf: Archivo de configuración del cargador del sistema operativo lilo(Linux loader), encargado del arranque de Linux y los demás s.s.o.o.

resolv.conf: Archivo de configuración de la resolución de nombres de red.

hosts: Archivo de configuración de redes(Ver documento adjunto).

En otros directorios:

/var/adm/backup: Cada vez que se hacen cambios en el sistema, se nos da la posibilidad de guardar una copia de seguridad de todos los archivos susceptibles de variar. Aquí se guardan dichos cambios en formato tar.gz

/opt/: Aquí se guardan los programas considerados opcionales, como son el navegador Netscape, o el entorno de ventanas KDE.

/vmlinuz: Es el núcleo del sistema operativo. Puede llamarse de otra forma.

/mnt/: Aquí se suelen montar las particiones o dispositivos que tenga el sistema.

/usr/doc/Howto: Múltiples ayudas catalogadas por temas.

/usr/doc/packages: Conjunto de README, y documentos varios para cada paquete instalado, en formato texto, postscript, html, etc.

En el directorio **/proc**:

Si miramos el contenido de cualquiera de estos archivos(**cat** <archivo>), obtendremos información útil sobre nuestro sistema:

devices: Información sobre los dispositivos que el sistema reconoce.

ioports: Info. sobre las direcciones de entrada/salida de los dispositivos.

interrupts: Info. sobre las interrupciones usadas por el sistema.

dma: Info. sobre los canales de acceso directo a memoria del sistema.

En el directorio de los dispositivos: (**/dev)**

hda: Disco primario maestro.

hdb: Disco primario esclavo.

hdc: Disco secundario maestro.

hdd: Disco secundario esclavo.

fd0: Disquetera.

mouse: Contiene un enlace al puerto del ratón(tty0, psaux, etc)

modem: Enlace al puerto del módem(tty0, tty1, etc)

lpd: Impresora.

fb0: Frame buffer device. Es un dispositivo que controla la tarjeta gráfica a bajo nivel, sin necesidad de entorno gráfico.

1

2