AAS 2636
Trab
II 2011
L4

UNIVERSIDAD CATÓLICA ANDRÉS BELLO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

UCAB Universidad Católica
CENTRO CULTURAL PADRE CARLOS G. PLAZA
BIBLIOTECA CENTRAL - SALA - PLANTA SALA - REFERENCIA

TRABAJO DE ASCENSO:

GUÍA DE COMANDOS COMUNES Y BÁSICOS EN SISTEMAS OPERATIVOS DE CÓDIGO ABIERTO (LINUX/UNIX)

PRESENTADO POR:
ING. JESUS DARÍO LEÓN ZAMBRANO
C.I. 12.353.508
dario.leon@gmail.com

ANTE LA UNIVERSIDAD CATÓLICA ANDRÉS BELLO PARA OPTAR AL ESCALAFÓN DE ASISTENTE.

CARACAS, MAYO DE 2011

Guía de comandos comunes y básicos para sistemas operativos de código abierto (Linux/Unix)

Indice de contenido

Convenciones Tipográficas....4
Introducción....5
Importancia y uso de la línea de comandos en sistemas operativos Linux/Unix....6
Sintaxis de los comandos en código abierto....6
Ayuda sobre la línea de comandos....9
El comando man....9
Listando – visualizando contenido archivos y directorios....12
El comando ls....12
El comando more....14
El comando less....15
El comando cat....16
Comando "cat" sin opciones ni argumentos....16
Creación de archivos....17
Añadiendo líneas al final de un archivo....17
Concatenación....18
Verificando información al final de un archivo....18
Verificando información del comienzo de un archivo....19
Dónde estamos ubicados....20
Moviendose entre directorios....20
Rutas absolutas y rutas relativas....21
Manipulando archivos y directorios....22
Copiando archivos y directorios....22
Copiando archivos....22
Copiando un archivo a una localización distinta....22
Copiando directorios....23
Borrando archivos y directorios....23
Borrando archivos....23
Borrando directorios....24
Moviendo o renombrando archivos o directorios....25
Renombrando archivos....25
Moviendo archivos a un directorio distinto....25
Renombrando directorios....26
Moviendo directorios y su contenido a un directorio distinto....26
Creando directorios....27
Creando archivos....27
Creación de enlace simbólico....28
Manipulando el shell....29
Configurando una variable....29
Cuál es el shell que estamos usando....29

Desconfigurando una variable....30
Personalizando el uso de los shells....31
"Prompt"....31
"PATH"....32
history....32
Metodos de completación en bash....32
Comandos de redirección....33
Redirección (pipe)....34
Archivos de inicialización....35
Definición de alias....35
Cambiando permisos a archivos y directorios....35
Modo octal y su representación....36
Modo simbólico y su represtación....36
Modo octal....36
Modo simbólico....37
Modificando permiso por defecto sobre archivos y directorios....38
Buscando coincidencias de patrones en archivos y directorios....39
"grep"....39
"find"....41
Comandos para administrar procesos....44
"ps"....44
"pgrep"....44
"kill"....45
"pkill"....45
Editar texto....46
Editor de texto vi....46
Bibliográfia....46

Convenciones Tipográficas

Tipo de letra	Uso	Ejemplo
"***Times New Roman***"	Comandos y/o opciones durante el desarrollo del texto o ejercicios, se exceptúan en el contenido de los ejemplos.	El comando "***man***" La opcion "***-R***" del comando "***chmod***"
Times New Roman	Referencias a otras secciones en el manual.	Verificar el uso del comando "***su***" en la pagina 38
"Times New Roman"	Directorios, caracteres especiales, palabras en inglés usuarios o rutas que hagan referencia a una ubicación en el sistema operativo.	"/home/dario" el carácter especial ";" el "home directory"
<Times New Roman>	Teclas que presionar	Presionar la tecla <ctrl>
Times New Roman	Ejemplos	*dario@Leona:~$ ls -l /usr/local*

Introducción

El presente documento tiene como finalidad crear una guía con los principales comandos básicos que son usados para sistemas operativos de código abierto, en este documento se cuida que la información de los comandos básicos incluidos pueda ser usado en la mayoría de los GNU/Linux y en Unix sobre todo aquellos basados en "System V".

Mucho de los comandos incluidos acá, han sido probados por lo menos en los siguientes sistemas operativos: Ubuntu, RedHat, OpenSolaris, Solaris, Slackware, HP-UX.

Por tanto con esta guiá un usuario puede manejarse o usar la línea de comandos de una forma cómoda y adecuada.

Como la idea de esta guiá es ser "portable", es decir, que pueda ser usado sobre la mayoría de los GNU/Linux o Unix sin problemas, se decidió omitir el uso del ambiente gráfico (ni siquiera un recorrido general), de igual manera, se decidió no colocar variaciones de comandos u opciones que impliquen instalaciones de paquetes adicionales al sistema operativo, esto con el propósito de no ligarla a un sistema operativo especifico.

De igual manera este documento se ha querido incluir como un texto de ayuda en la materia "sistema de operacion I" en la escuela de informática de la Universidad Católica Andrés Bello, con el propósito de que a través de este los estudiantes reciban un primer acercamiento a los comandos usados en este tipo de sistemas.

Este documento cubre las respuestas a dudas e interrogantes de parte de los administradores de sistemas operativos, estudiantes, usuarios generales, ante la pregunta: ¿si me cambio entre sistemas operativos de código abierto o Unix, tengo que aprender los nuevos comandos básicos relacionados a este?, la respuesta es: No. Ejemplo, supongamos que manejo Solaris (Unix) y empiezo a trabajar con RedHat (Linux), ¿debo aprender los comandos básicos de RedHat?, la respuesta es: No.

Cualquier comentario estaría agradecido se escribiera a:

dario.leon@gmail.com o dario.leon@cacaoenbytes.com.

Importancia y uso de la línea de comandos en sistemas operativos Linux/Unix

Un sistema operativo en modo general es un conjunto de programas, que tratan de facilitar la comunicación entre un usuario y un dispositivo físico (hardware), entiéndase por ese dispositivo: desde una computadora de escritorio, hasta servidores, pasando por teléfonos celulares.

El uso de la línea de comandos en sistemas operativos de código abierto es muy frecuente, su ejecución se realiza desde un terminal. Un terminal actúa como un interprete de comandos, permite escribir un comando, un ejecutable y que este sea traducido a lenguaje máquina sin la necesidad de llevar al usuario a escribir largas líneas. Para la ejecución en un terminal existen según el sistema operativo distintas opciones. Algunas de ellas:

Barra de herramientas superior.
Aplicaciones en la barra de herramientas
Botón derecho sobre el escritorio.

Para los sistemas operativos GNU/Linux y Unix la estructura y la sintaxis de estos es muy similar, se decide incluir Unix: primero por que las grandes empresas emplean estos sistemas operativos en sus servidores y segundo para desmitificar un poco que el manejo de los comandos básicos en este tipo de sistemas operativos, son totalmente distintos.

Sintaxis de los comandos en código abierto

Así es la sintaxis de los comandos sobre los sistemas operativos de código abierto:

"Comando opción argumento"

Los comandos pueden ser usados solos o con opciones o argumentos o con opciones y argumentos:

Item	Descripción
Comando	Especifica que hará el sistema (ejecutable)
Opción	Especifica como se ejecuta el comando (modificador), se usa generalmente un guión (-)
Argumento	Especifica que es afectado (archivo, directorio, texto, etc)

Ejemplo:

```
dario@Leona:~$ ls
archivo  archivo2   directorio1_ren  directorio3  directorio5  dir_umask
archivo1 directorio1 directorio2     directorio4  directorio6  dir_umask1
```

Ejemplo:

dario@Leona:~$ ls -l
total 36
-rw------- 1 dario dario 0 2009-11-07 18:41 archivo
-rw------- 1 dario dario 0 2009-11-07 18:42 archivo1
-rw------- 1 dario dario 0 2009-11-07 18:42 archivo2
dr--r--r-- 2 dario dario 4096 2009-10-30 15:51 directorio1
drwxr-xr-x 4 dario dario 4096 2009-11-07 18:38 directorio1_ren

Ejemplo:
dario@Leona:~$ ls -l /usr/local
total 32
drwxr-xr-x 2 root root 4096 2009-04-20 09:29 bin
drwxr-xr-x 2 root root 4096 2009-04-20 09:29 etc
drwxr-xr-x 2 root root 4096 2009-04-20 09:29 games
drwxr-xr-x 2 root root 4096 2009-04-20 09:29 include
drwxr-xr-x 3 root root 4096 2009-04-20 09:29 lib
lrwxrwxrwx 1 root root 9 2010-07-15 19:49 man -> share/man
drwxr-xr-x 2 root root 4096 2009-04-20 09:29 sbin
drwxr-xr-x 11 root root 4096 2009-04-20 09:38 share
drwxr-xr-x 2 root root 4096 2009-04-20 09:29 src

De igual manera se pueden ejecutar múltiples comandos separados por el carácter especial ";"

Ejemplo:
dario@Leona:~$ cal;date
March 2010
Su Mo Tu We Th Fr Sa
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31

Thu Mar 11 20:00:32 VET 2010

La salida del conjunto de comandos anterior muestra el calendario del mes configurado en la máquina, además de la fecha que esté configurada en el sistema operativo (vale la pena destacar que en el orden en que se coloque así serán ejecutados).

Nota: En los sistemas operativos de código abierto, existen distintos tipos de usuarios, de ellos vale la pena destacar dos:

El superusuario conocido como "root", usuario con todos los privilegios dentro del sistema operativo, dicho usuario puede, desde listar el contenido de cualquier directorio o archivo hasta apagar la maquina. Por razones de seguridad se recomienda solo usarlo cuando se es un usuario del o los

sistemas operativos con experiencia o en casos de estricta necesidad, este usuario se representa mediante el prompt "#" (para mas detalles del prompt ver pagina 30, del presente documento).

El segundo tipo de usuario es o son los usuarios cotidianos del sistema operativo, una forma de crear estos usuarios es cuando se instalan los sistemas operativos *nix que cumplen el rol de ejecutar cualquier tarea cotidiana en el sistema, sea en un equipo personal, leer los correos, editar un documento o escuchar música o en un servidor, liberar espacio en un "sistema de archivos" o crear nuevos usuarios, monitorear la plataforma, etc. El nombre del usuario por lo general es el nombre de la persona dueña de ese perfil, ejemplo Dario Leon (la persona) es dueño del usuario en *nix "dario", el "prompt" para estos usuarios es generalmente el carácter especial "$" (esto ultimo depende del "Shell" que estemos usando) – ver pagina 28 del presente documento.

Para realizar tareas privilegiadas, a partir de un usuario cotidiano o de un usuario sin privilegios, dependiendo del sistema operativo se pueden usar comandos como "***sudo***", "***pfexec***" o roles. Estos comandos o usuarios especiales (caso de roles) permiten que el usuario ejecute ciertas tareas administrativas (privilegiadas) asignadas a el. Otro comando que permite que directamente nos cambiemos al usuario administrador "root" o al superusuario es el comando "***su***".

El comando ***"su"*** ("switch user" en inglés) permite a un usuario sin privilegios cambiarse (realizar un "switch") entre usuarios, el usuario puede usar el comando "su" seguido del carácter especial "-" (el carácter es necesario si queremos que al cambiarnos se usen todas las características de la sesión del nuevo usuario) seguido del "login" al que se quiere cambiar, este cambio ("switch" entre usuarios) permite a un usuario ejecutar tareas como otro, ejemplo cambiarse a "root" (debe conocer su password) y ejecutar tareas privilegiadas en el sistema evitando el proceso de salir de la sesión y entrar en la sesión del usuario con el que se quieren ejecutar las tareas.

Ejemplo:

```
dario@bt:~$ more /etc/sudoers
/etc/sudoers: Permission denied
dario@bt:~$ su - root
Password:
root@bt:~# more /etc/sudoers
# /etc/sudoers
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the man page for details on how to write a sudoers file.
#
Defaults env_reset
```

(SALIDA CORTADA POR CUESTIONES DE ESPACIO)

Vemos en el ejemplo anterior que el usuario no podía ver el contenido del archivo usando el

comando ***"more"*** para hacerlo se cambio a "root" y pudo verificar el contenido.

Ejercicios:

1.- Explique qué es un terminal

2.- ¿En un terminal puede ser ejecutado solo un comando?

3.- Dado los siguientes parámetros, indique como se deben ordenar para su correcta ejecución:
argumento: "root" "/etc/hosts" "/etc/passwd"
comando: "***grep***"
opción: "***-l***"

Ayuda sobre la línea de comandos

El comando man

Abreviatura de manual en inglés, dicho comando ofrece información detallada de comandos y como usarlos.

Una página de manual se ejecuta usando la palabra reservada "***man***" seguido del comando sobre el que se quiera obtener información. No es necesario conocer todos los comandos y sus opciones en un sistema operativo de código abierto, el "***man***" es una de las herramientas más potente con las que se puede contar a nivel de línea de comandos (CLI).

Ejemplo:

dario@Leona:~$ man ls
LS(1) User Commands LS(1)
NAME
ls - list directory contents
SYNOPSIS
ls [OPTION]... [FILE]...
DESCRIPTION
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort.
Mandatory arguments to long options are mandatory for short options too.
-a, --all
do not ignore entries starting with .
-A, --almost-all
do not list implied . and ..
--author
with -l, print the author of each file
(SALIDA CORTADA POR CUESTIONES DE ESPACIO)
dario@Leona:~$ man cal
CAL(1) BSD General Commands Manual CAL(1)
NAME
cal, ncal - displays a calendar and the date of easter
SYNOPSIS

cal [-3jmy] [[month] year]
ncal [-jJpwy] [-s country_code] [[month] year]
ncal [-Jeo] [year]
DESCRIPTION
The cal utility displays a simple calendar in traditional format and ncal offers an alternative layout, more options and the date of easter. The new format is a little cramped but it makes a year fit on a 25x80 termi□ nal. If arguments are not specified, the current month is displayed. The options are as follows:
-3 Print the previous month, the current month, and the next month all on one row. This flag will only work if you are not display□ ing Julian days (see -J below).
-J Display Julian Calendar, if combined with the -e option, display
(SALIDA CORTADA POR CUESTIONES DE ESPACIO)

Si necesitamos averiguar si existe una página del "***man***" a partir de una palabra, se debe ejecutar el comando "***man***" seguido de la opción "***-k***" seguido de la palabra que se quiere buscar (lo que sucede es que se buscan todas las ocurrencias de la palabra a buscar en las paginas del comando ***"man"***). De nuevo no es necesario saber todos los comandos, basta tener un poco de lógica y esto ayuda a conseguir el comando correcto.

Ejemplo:
dario@Leona:~$ man -k calculator
bc (1) - An arbitrary precision calculator language
dc (1) - an arbitrary precision calculator
gcalctool (1) - a desktop calculator
gnome-calculator (1) - a desktop calculator
xcalc (1) - scientific calculator for X

Todos los sistemas operativos Unix/Linux , permiten cambiar el idioma del "***man***" (cabe acotar que para un cambio efectivo en todas las paginas del "***man***", previo deben haber sido instaladas estas páginas en el idioma deseado sobre el sistema operativo). Para que el cambio de lenguaje sea efectivo hay que manipular dos variables de "shell" en el sistema operativo. La variable "LANG" y la variable "LC_ALL", es necesario colocar el formato del idioma que queremos usar, la definición del idioma es con la siguiente sintaxis "idioma_territorio.conjunto_de_caracteres,versión" (versión generalmente es optativo). La definición de las variables queda por ejemplo así (detalles de definición de una variable sobre un "shell" en la página 28 de la presente guía) :

"LANG=es_ES.utf8"
"LC_ALL=es_ES.utf8"

Ejemplo:
dario@Leona:~$ man ping
PING(8) BSD System Manager's Manual PING(8)

NAME

ping — envía paquetes ICMP ECHO_REQUEST a servidores de red

SINOPSIS

ping [-dfnqrvR] [-c count] [-i wait] [-l preload] [-p pattern] [-s packetsize]

DESCRIPCIÓN

Ping utiliza el datagrama obligatorio ICMP del protocolo ECHO_REQUEST para solicitar una ICMP ECHO_RESPONSE por parte de un servidor o una puerta de acceso. Los datagramas ECHO_REQUEST ("pings") tienen una IP y cabecera ICMP, seguidos por un "struct timeval" y un número arbitrario de bytes "pad" usados para completar el paquete. Las opciones son las siguientes:

-c count

Para después de enviar (y recibir) count paquetes ECHO_RESPONSE.

-d Establece la opción SO_DEBUG en el socket en uso.

(SALIDA CORTADA POR CUESTIONES DE ESPACIO)

Vemos como en la salida anterior, el comando "***ping***" es descrito en idioma español de España ("es_ES.utf8").

NOTA: Si nuestra intención es dejar fijo el cambio, podemos incluir dentro de los archivos del perfil (profile) del usuario, la variables antes mencionadas, para que al momento del usuario entrar en su sesión, esta información sea cargada satisfactoriamente (ver pagina 34 de la presente guía).

A continuación una tabla que muestra algunas opciones de como moverse dentro del "man":

Tecla de movimiento	**Acción**
Barra espaciadora	Despliega la próxima pantalla de una página del man
Enter	Despliega la próxima línea de una página del man
b	Mueve de regreso una pantalla completa
f	Mueve hacia adelante una página completa
/patrón	Busca hacia adelante por un *patrón*
n	Encuentra la próxima ocurrencia de un patrón después de haber usado */patrón*
N (mayuscula)	Encuentra la próxima ocurrencia hacia atras de un patrón después de haber usado /patrón
h	Provee una descripción de movimiento
q	Salida de la página del man

Ejercicios:

1.- ¿Cómo saber sobre un sistema operativo los distintos tipos de comandos relacionados al comando "***zip***" que existen, partiendo del hecho de saber que existe la palabra reservada "***zip***"?

2.- Ejecutando "***man find***" se quiere buscar dentro del manual la ocurrencia del patron "file",

¿cómo hacerlo?

3.- Ejecutando "***man find***" *se quiere* mover una ventana mas abajo, ¿cómo hacerlo?.

Listando – visualizando contenido archivos y directorios

El comando ls

El comando "***ls***" es una abreviatura de "list" en inglés (lista sería la traducción), dicho comando permite como su nombre lo indica listar o mostrar información detallada de archivos o directorios en el intérprete de comandos o terminal.

El comando "***ls***" puede ser ejecutado con o sin opciones, con o sin argumentos:

El comando "***ls***" sin opciones lista el contenido del directorio donde este posicionado el usuario, la salida será mostrada sin detalle acerca de los archivos o directorios.

El comando "***ls***" con la opción "***-l***" muestra en detalle el contenido del directorio.

Ejemplo:

```
dario@Leon:~/ejemplos$ ls -l
total 8 (fila 1)
-rw-r--r-- 1 dario dario  0 2009-10-23 21:06 archivo (fila 2)
drwxr-xr-x 2 dario dario 4096 2009-10-23 21:06 directorio1 (fila 3)
drwxr-xr-x 2 dario dario 4096 2009-10-23 21:07 directorio2
dario@Leon:~/ejemplos$
```

La salida anterior muestra:

Fila 1:

total 8 cantidad de archivos o directorios (resumen) que existen donde se ejecutó el comando medido en bloques (los bloques pueden variar de sistema operativo en sistema operativo).

Fila 2 y fila 3:

- El caracter "-" indica que es un archivo, el caracter "d" significa directorio.
- El conjunto de caracteres "rw- ", indican la permisología que ejerce el usuario sobre "archivo" o "directorio". Si es sobre un archivo la permisología es definida así: el caracter "r" significa que el bit de lectura esta activo, es decir, el usuario dueño del archivo dario puede leer "archivo", el caracter "w" significa que el bit de escritura esta activo y el usuario dueño (dario) puede escribir sobre "archivo", finalmente el caracter "-" o "x" nos especifica que el bit de ejecución esta apagado para el usuario dueño de "archivo" o aparecería con una x si estuviera encendido.

Si es sobre un directorio, los bits de ejecución son iguales, pero la interpretación es ligeramente distinta, veamos como cambia a continuación: si el bit r es el único encendido "r--", se podrá sobre el directorio solo listar su contenido, pero sin entrar en este, si esta la combinación de "r" con "x" se puede, listar el directorio desde fuera, se puede entrar en el directorio, se puede listar su contenido desde dentro, se pueden editar los archivos dentro del directorio, pero no se puede crear nuevos archivos. Si la combinación es "rw-", el dueño del directorio puede solo listar desde fuera del

directorio.

- El conjunto de caracteres "r-- " representan la permisología para el grupo, la distribución del conjunto de caracteres en el ejemplo resaltado nos enseña que solo esta permitida la lectura para el grupo dueño del archivo, los bits de escritura y ejecución están apagados.
- El último conjunto de caracteres "r--" manejan la permisología para el resto de usuarios (others), en este ejemplo lectura para cualquier otro usuario, pero escritura y ejecución apagados.
- El número "1" indica la cantidad de enlaces (archivos que comparten un mismo número de inodo).
- "dario" usuario dueño del archivo.
- "dario" grupo dueño del archivo.
- "0" representa el tamaño del archivo.
- "2009-10-23 21:06 "Fecha de creación del archivo y luego representara la fecha de modificacion del archivo.
- "archivo" es el nombre: puede ser: archivo, ejecutable, comando, directorio, etc.

El comando "***ls***" con la opción "***-a***" permitirá mostrar archivos o directorios ocultos, en código abierto estos comienzan con el carácter punto ".".

Ejemplo:

```
dario@Leon:~/ejemplos$ ls -la
total 16
drwxr-xr-x  4 dario dario 4096 2009-10-25 21:58 .
drwxr-xr-x 36 dario dario 4096 2009-10-23 21:08 ..
-rw-r--r--  1 dario dario    0 2009-10-23 21:06 archivo
-rw-r--r--  1 dario dario    0 2009-10-25 21:58 .archivooculto
drwxr-xr-x  2 dario dario 4096 2009-10-23 21:06 directorio1
drwxr-xr-x  2 dario dario 4096 2009-10-23 21:07 directorio2
```

Donde "." representa al directorio actual, ".." representa al directorio superior, ".archivooculto" a un archivo oculto.

Para conocer información acerca de un directorio pero sin desplegar su contenido se usa el comando "***ls***" seguido de la opción "***-d***" y el nombre del directorio.

Ejemplo:

```
dario@Leon:~$ ls -ld ejemplos/
drwxr-xr-x 4 dario dario 4096 2009-10-25 21:58 ejemplos/
dario@Leon:~$
```

Ejercicios:

1.- ¿Como se lista el contenido de un directorio, incluyendo archivos ocultos por orden de fecha?
2.- ¿Como se representa un archivo oculto?
3.- ¿Que opción del comando ls puede ser usado para listar archivos y directorios de forma

recursiva?

4.- ¿Que hace la opción -t con el comando ls?.

El comando more

Como su nombre lo indica permite verificar mas ("cuando se podía programar sobre el UNIX se creo este comando para evitar el sonido incesante de las campanas sobre los PD-11 de Berkley"[1]) información dentro de un archivo (vale la pena acotar que actualmente se puede programar en Unix, la mención es cita textual de los creadores del comando).

Opciones que permiten navegar una vez desplegada la información dentro del archivo:

Tecla de movimiento	**Acción**
Barra espaciadora	Despliega la próxima pantalla de una página del archivo
Enter	Despliega la próxima línea dentro del archivo
b	Mueve de regreso una pantalla completa
f	Mueve hacia adelante una página completa
/*patrón*	Busca hacia adelante por un *patrón*
n	Encuentra la próxima ocurrencia de un patrón después de haber usado /patrón
N (mayúscula)	Encuentra la próxima ocurrencia hacia atrás de un patrón después de haber usado /patrón
h	Provee una descripción de movimiento
q	Salida del archivo

Ejercicios:

1.- Ejecute "more /var/log/messages" (sobre linux) o "more /var/adm/messages" (sobre Unix) e indique:

1.1.- ¿Cómo moverse páginas completas?.

1.2- ¿Cómo moverse línea a línea?.

1.3.- Indique que opción debo usar para que sobre el terminal que se este usando el comando "***more***" no se use la barra de desplazamiento ("scroll" en inglés), en vez de esta se limpie la pantalla cada nueva página.

2.-Indique como realizo un refrescamiento de pantalla, por ejemplo luego de ejecutar la tecla <h> en el uso del comando.

El comando less

Su nombre proviene del juego de palabras "más hacia atrás[2]" (en inglés "backward more").

Opciones de navegación una vez ejecutado el comando:

---

[1] http://www.halwitz.org/halbert/more.html

[2] http://www.greenwoodsoftware.com/less/faq.html#history

Tecla de movimiento	Acción
Barra espaciadora	Despliega la próxima pantalla de una página del archivo
Enter	Despliega la próxima línea dentro del archivo
b	Mueve de regreso una pantalla completa
f	Mueve hacia adelante una página completa
/patrón	Busca hacia adelante por un *patrón*
n	Encuentra la próxima ocurrencia de un patrón después de haber usado */patrón*
N (mayúscula)	Encuentra la próxima ocurrencia hacia atrás de un patrón después de haber usado /patrón
h	Provee una descripción más completa de la mostrada acá
q	Salida del archivo

NOTA: Cuando se realiza la búsqueda utilizando el comando anterior las coincidencias serán resaltadas. Las opciones son iguales que el comando more.

Ejercicios:

1.- Ejecute la siguientes instrucciones y comente cual es la principal diferencia:

"*man ls | more*" ejecute ahora barra espaciadora, luego enter, ahora ejecute "/dir"

"*man ls | less*" ejecute ahora barra espaciadora, luego enter, ahora ejecute "/dir"

Nota: Para la ejecución del ejercicio es necesario investigar para que se usa el "|", usando las paginas del man del sistema operativo o usando Internet o mas adelante buscando la referencia en esta guía.

El comando cat

El curioso nombre proviene de "CATenate" en inglés, ya que una de sus funciones es permitir la concatenación de archivos. Algunas de sus funciones son:

Permitir visualizar la información dentro de un archivo.

Ejemplo:

```
dario@Leona:/etc/default$ cat cron
# Cron configuration options
# Uncomment this option for LSB name support in /etc/cron.d/
#LSBNAMES='-l'
# Whether to read the system's default environment files (if present)
# This will make cron set a proper charset for the mails it sends
# Comment this or set it to something other than 'yes' to prevent
# cron from reading it.
READ_ENV="yes"
# Extra options for cron, see cron(8)
# For example, set a higher log level to audit cron's work
```

 # EXTRA_OPTS="-L 2"

Es importante destacar que si se usa para la lectura de archivos, estos deben ser de un tamaño reducido ya que no tiene las opciones de los comandos descritos anteriormente en comandos como "***more***", "***less***", etc. Tales como: avance de páginas, avance de líneas, etc.

NOTA: si la intención es obtener salidas como la de los comandos previos, es decir, que la salida sea mostrada pantalla por pantalla, hay que utilizar comandos de redirección, en este caso especifico la redirección refiere a la salida de un comando redirigrlo a la entrada de otro comando como lo hace el caracter de redirección "|" (para mas detalle del tema referirse a la página 33 del presente documento).

Otras funciones incluyen:

Comando "cat" sin opciones ni argumentos

Si se ejecuta sin nombre de archivo todo lo que se escriba al presionar la tecla <enter> será visualizado en la salida por defecto ("standard" output), es decir el monitor, ya que como no se redirecciona, entendiendo por redirección que no se enviá a un sitio distinto del monitor, ya que se puede enviar por el uso de caracteres especiales como ">" a un archivo por ejemplo (para conocer mas sobre el tema de redirección si es de interés referirse a la página 32 del presente documento).

Ejemplo:

dario@Leona:/etc/default$ cat
hola mundo
hola mundo

NOTA: Para terminar el uso del comando "***cat***" sin opciones presionamos la tecla <*ctrl*> en simultáneo con la tecla <*c*>

Creación de archivos

Este modo de creación de archivos es mucho más rápido que usar comandos como vi (para el tema de redirección referirse a la página 33 del presente documento) u otros.

La creación de archivos se realiza entonces usando el comando "***cat***" seguido del caracater especial ">" que indica redirección (redirigir refiere a enviar el resultado a una salida distinta del monitor (para mas detalle referirse a la página 32 del presente documento) y el nombre que tendrá al archivo.

La sintaxis a continuación:

"***cat > archivo***"

Ejemplo:

Presionando en simultáneo la tecla <ctrl> y la tecla <d> se guarda la información.

dario@Leona:~/ejemplos$ cat > archivocat
hola mundo
Adios
dario@Leona:~/ejemplos$

NOTA: Para finalizar la edición se ejecuta la tecla <ctrl> en simultáneo con la tecla <d> así como se menciona arriba, al ejecutar el comando "***cat > archivo***" se borrará la información ya almacenada.

Añadiendo líneas al final de un archivo

Así como se menciona en el título del apartado, para añadir líneas al final del archivo se usa el versátil comando "***cat***" seguido por los caracteres especiales ">>" y el nombre del archivo, estos caracteres en conjunto permiten una redirección distinta al monitor que es la redirección por defecto (para el tema de redirección referirse a la página 32 del presente documento).

Ejemplo:
dario@Leona:~/ejemplos$ cat >> archivocat
Esto va despues de adios
dario@Leona:~/ejemplos$ cat archivocat
Hola Mundo
Adios
Esto va despues de adios
dario@Leona:~/ejemplos$

Concatenación

En este caso el comando "***cat***" sirve para unir varios archivos y así crear uno solo.

Ejemplo:
dario@Leona:~/ejemplos$ more archivocat1 archivocat2
::::::::::::::
archivocat1
::::::::::::::
Hola
::::::::::::::
archivocat2
::::::::::::::
mundo
dario@Leona:~/ejemplos$ cat archivocat1 archivocat2 > archivocatfinal
dario@Leona:~/ejemplos$ more archivocatfinal
Hola

mundo
dario@Leona:~/ejemplos$

Ejercicios:

1.- Ejecute "cat /var/log/messages" (sobre linux) o "cat /var/adm/messages" (sobre unix), explique que diferencia encuentra con ejecutar "more /var/log/messages" (sobre linux) o "more /var/adm/messages" (sobre unix).

2.- Cree un archivo llamado "pruebacat" y escriba "hola" dentro de el, ahora añada al final del archivo "pruebacat": "final de archivo", muestre el contenido del archivo final. Escriba la secuencia de comandos y la salida final por pantalla.

3.- Una el archivo "/etc/passwd" con el archivo "/etc/hosts" y el archivo "/etc/nsswitch.conf" en un archivo llamado prueba. Muestre que comandos ejecutó y cuál es la salida por pantalla.

Verificando información al final de un archivo

El comando "***tail***" (algo así como cola o final en español) permite verificar ultimas 10 líneas sobre algún archivo.

Ejemplo:

dario@Leon:~$ tail /var/log/messages (ejecutado sobre linux)
Oct 27 18:43:43 Leon kernel: [317551.621932] scsi 11:0:0:1: CD-ROM Kingston DataTraveler U3 6.16 PQ: 0 ANSI: 0
Oct 27 18:43:43 Leon kernel: [317551.623290] sd 11:0:0:0: [sdb] Attached SCSI removable disk
Oct 27 18:43:43 Leon kernel: [317551.623374] sd 11:0:0:0: Attached scsi generic sg2 type 0
Oct 27 18:43:43 Leon kernel: [317551.629570] sr1: scsi3-mmc drive: 8x/40x writer xa/form2 cdda tray
Oct 27 18:43:43 Leon kernel: [317551.629814] sr 11:0:0:1: Attached scsi generic sg3 type 5
Oct 27 18:43:47 Leon kernel: [317554.979565] sd 11:0:0:0: [sdb] 4003711 512-byte hardware sectors: (2.04 GB/1.90 GiB)
Oct 27 18:43:47 Leon kernel: [317554.980209] sd 11:0:0:0: [sdb] Write Protect is off
Oct 27 18:43:47 Leon kernel: [317554.981849] sd 11:0:0:0: [sdb] 4003711 512-byte hardware sectors: (2.04 GB/1.90 GiB)
Oct 27 18:43:47 Leon kernel: [317554.982424] sd 11:0:0:0: [sdb] Write Protect is off
Oct 27 18:43:47 Leon kernel: [317554.982436] sdb: sdb1
dario@Leon:~$

Para verificar mas líneas de las por defecto usamos la opción "***-n***"

Ejemplo:

dario@Leon:~$ tail -5 /var/log/messages (ejecutado sobre linux)
Oct 27 18:43:47 Leon kernel: [317554.980209] sd 11:0:0:0: [sdb] Write Protect is off
Oct 27 18:43:47 Leon kernel: [317554.981849] sd 11:0:0:0: [sdb] 4003711 512-byte hardware sectors: (2.04 GB/1.90 GiB)
Oct 27 18:43:47 Leon kernel: [317554.982424] sd 11:0:0:0: [sdb] Write Protect is off
Oct 27 18:43:47 Leon kernel: [317554.982436] sdb: sdb1

Oct 27 18:56:05 Leon -- MARK --
dario@Leon:~$

Si la intención es monitorear en tiempo real algún archivo se puede usar la opción especial "***-f***" para el comando ***"tail"***, cada nueva línea aparecerá sobre dicho archivo, para detener el monitoreo del archivo se ejecuta la tecla <ctrl> seguido de la tecla <c>, el comando completo sería "***tail -f /var/log/messages***" (ejemplo sobre linux) o "***tail -f /var/adm/messages***" (ejemplo sobre unix) .

Ejercicios:

1.- Muestre las 7 últimas líneas del archivo "/var/log/messages".

2.- Resalte en el archivo "/var/log/messages" o /var/adm/messages" según sea su caso, las ocurrencias de la letra "a", hágalo en una sola línea de comando y muestre que uso para esto.

Verificando información del comienzo de un archivo

Para verificar la información correspondiente a la cabecera de un archivo usamos el comando "***head***" (cabeza o comienzo en español): Este comando enseña las primeras líneas de un archivo.

Ejemplo:

dario@Leon:/var/log/apt$ head -4 /var/log/apt/term.log
Log started: 2009-10-01 20:40:21
(Reading database ... 132248 files and directories currently installed.)
Removing gdm-guest-session ...
dario@Leon:/var/log/apt$

Ejercicios:

1.- Muestre las primeras 2 líneas del archivo "/etc/nsswitch.conf".

2.- Muestre las primeras 3 líneas y las 4 ultimas líneas del archivo "/etc/nsswitch.conf", escriba que secuancia de comandos uso para esto.

Dónde estamos ubicados

El comando "***pwd***" (print working directory) que significa imprimir el directorio de trabajo permite verificar el directorio donde se esta posicionado en un momento dado.

Ejemplo:

dario@Leon:~/ejemplos$ pwd
/home/dario/ejemplos

Ejercicios:

1.- Ejecute "cd /tmp" y ejecute el comando "***pwd***"

2.- Ejecute "cd /usr/bin" y ejecute el comando "***pwd***"

Moviendose entre directorios

Cuando hay que moverse de un directorio a otro se puede usar el comando "***cd***" ("change

directory" en inglés), obviamente permite moverse de un directorio a otro como su traducción lo indica. El comando "***cd***" sin argumento lleva al directorio "home" del usuario que está usando la sesión actual.

Ejemplo:

dario@Leon:~/ejemplos$ cd /usr/local/zabbix/
dario@Leon:/usr/local/zabbix$ cd
dario@Leon:~$ pwd
/home/dario
dario@Leon:~$

En el ejemplo anterior se ubica en un directorio distinto y ejecutando el comando "***cd***" nos lleva al directorio "home" del usuario "dario" quien es el usuario que inicio sesión (dependiendo del sistema operativo el "home directory" del usuario puede variar ej: "/export/home/usuario" en los tipo Unix o "/home/usuario" en los tipo Linux).

NOTA: El "home directory" de un usuario es la carpeta donde el usuario tiene permisos para almacenar información, música, documentos, crear sus propios archivos ejecutables, etc, sin verse en la necesidad de usar comandos especiales o que le sean otorgados permisos especiales.

Algunos atajos en el uso del comando "***cd***" son:
El comando "***cd***" seguido del carácter especial "-" nos lleva al directorio anterior.

Ejemplo:

dario@Leon:~$ pwd
/home/dario
dario@Leon:~$ cd -
/usr/local/zabbix
dario@Leon:/usr/local/zabbix$

El comando "***cd***" seguido del caracter especial "." colocado dos veces ".." posiciona al usuarios en el directorio inmediatamente superior (conocido también como directorio padre).

Ejemplo:

dario@Leon:/usr/local/zabbix$ cd ..
dario@Leon:/usr/local$ pwd
/usr/local
dario@Leon:/usr/local$

Para moverse a un directorio distinto y si se conoce donde está en el árbol se usa una ruta relativa.

Ejemplo:

dario@Leon:~/ejemplos/directorio1/subdirectorio1$ pwd
/home/dario/ejemplos/directorio1/subdirectorio1

dario@Leon:~/ejemplos/directorio1/subdirectorio1$ cd ../../directorio2/
dario@Leon:~/ejemplos/directorio2$ pwd
/home/dario/ejemplos/directorio2
dario@Leon:~/ejemplos/directorio2$

Rutas absolutas y rutas relativas

En sistemas operativos de código abierto sean GNU/Linux o Unix el directorio raíz o el directorio principal es representado por el caracter especial "/".

Cuando nos referimos a una ruta absoluta, esta comienza con el uso del caracter especial de directorio raíz definido en el párrafo anterior.

Ruta relativa no se coloca el caracter especial "/" para llegar a alguna ruta destino, las rutas relativas están por debajo de la ruta donde el usuario esta posicionado en un momento dado.

Ejemplo:

Ruta destino mediante el uso de una ruta absoluta:

dario@Leon:~$ cd /home/dario/ejemplos/directorio1/subdirectorio1/
dario@Leon:~/ejemplos/directorio1/subdirectorio1$ pwd
/home/dario/ejemplos/directorio1/subdirectorio1
dario@Leon:~/ejemplos/directorio1/subdirectorio1$

Desde una ruta se puede alcanzar una ruta destino sin el uso del caracter especial "/"

Ejemplo:

dario@Leona:~$ cd ejemplos/
dario@Leona:~/ejemplos$ cd directorio2/subdir1/suddir2/
dario@Leona:~/ejemplos/directorio2/subdir1/suddir2$

Ejercicios:

1.- Vaya al directorio "/tmp", ahora cámbiese al directorio "/usr/local", escriba los comandos que uso, muestre por lo menos dos formas de hacerlo.

2.- Regrese al directorio hogar ("home directory" en inglés) del usuario. Escriba el comando que usa para hacer esto.

3.- Regrese al directorio anterior. ¿Cuál comando usa para esto?

4.- Suponga que en el directorio hogar ("home directory" en inglés) del usuario "dario" ("/home/dario") existe la siguiente distribución de directorios:

"prueba/prueba1" "prueba/prueba1/prueba2 prueba3/prueba/prueba4"

Si estoy en el directorio "prueba2" como puedo ir al directorio "prueba3" , expliquelo usando rutas absolutas y rutas relativas.

5.- Asegúrese de estar usando el "shell" "bash" o el "shell" "ksh", ejecute "cd /tmp" ahora ejecute el comando "***cd***" seguido del caracter especial "~", ahora ejecute el comando "***pwd***" y explique lo que sucede.

Manipulando archivos y directorios

Copiando archivos y directorios

El comando "***cp***" ("copy" en inglés) permite copiar archivos de una carpeta a otra e incluso copiar carpetas enteras de una ubicación a otra, he aquí el formato del comando:

"cp -opción fuente destino"

Copiando archivos

Para realizar una copia de un archivo se usa el comando seguido del nombre original y este seguido del nombre de la copia.

Ejemplo:

```
dario@Leon:~/ejemplos/directorio1$ ls
original  subdirectorio1
dario@Leon:~/ejemplos/directorio1$ cp original copia
dario@Leon:~/ejemplos/directorio1$ ls
copia  original  subdirectorio1
dario@Leon:~/ejemplos/directorio1$
```

Copiando un archivo a una localización distinta

Para realizar esta tarea es suficiente colocar las rutas fuente y destino o si se está ubicado en el directorio fuente colocar la ruta destino.

Ejemplo:

```
dario@Leona:~/ejemplos$ cp archivo1 directorio2
dario@Leona:~/ejemplos$ ls directorio2/
archivo1  subdir1
dario@Leona:~/ejemplos$
```

Copiando directorios

La opción "***-r***" permite copiar el contenido de una ubicación a otra, trayendo incluso el contenido del directorio.

Ejemplo:

```
dario@Leon:~/ejemplos/directorio1$ cp -r diroriginal/ dircopia
dario@Leon:~/ejemplos/directorio1$ ls
copia  dircopia  diroriginal  original  subdirectorio1
dario@Leon:~/ejemplos/directorio1$
```

Si se quiere implementar un poco de restricción ejecutamos el comando "***cp***" con la opción "***-i***" que nos previene de sobrescribir un archivo ya existente, esta opción es manipulable con la opción "***-r***"

igualmente.

Ejercicios:

1.- Copie el archivo "/etc/nsswitch.conf" a "/tmp". ¿Qué comando ejecutar para eso?.

2.- Copie el archivo "nsswitch.conf" en "/tmp" en la misma ruta pero con un nuevo nombre. ¿Qué comando usa para esto?.

3.- ¿Qué opción se usa para copias interactivas?.

Borrando archivos y directorios

El comando "***rm***" ("remove" en inglés) se usa para borrar archivos, directorios o ambos, ES IMPORTANTE RECORDAR QUE EL ARCHIVO O DIRECTORIO QUE SE BORRE CON DICHO COMANDO NO PODRÁ SER RECUPERADO, ES BORRADO DEFINITIVAMENTE DEL SISTEMA OPERATIVO. El formato del comando:

*"**rm -opción archivo(directorio)**"*

Borrando archivos

Solamente con colocar el comando "***rm***" seguido o de la ruta absoluta o si se está en el directorio fuente el nombre del archivo, éste se borrará sin ningún tipo de pregunta adicional.

Ejemplo:

dario@Leon:~/ejemplos/directorio1$ ls
copia copia1 dircopia dircopia1 diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$ rm copia1
dario@Leon:~/ejemplos/directorio1$ ls
copia dircopia dircopia1 diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$

Borrando directorios

Para borrar un directorio, si el(los) directorios están vacíos se usa el comando "***rmdir***" ("remove directory" en inglés) seguido de la ruta absoluta o nombre del directorio.

Ejemplo:

dario@Leon:~/ejemplos/directorio1$ ls
copia dircopia dircopia1 diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$ rmdir dircopia1
dario@Leon:~/ejemplos/directorio1$ ls
copia dircopia diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$

Si el directorio que se desea borrar posee dentro información (archivos, subdirectorios, ejecutables, binarios, etc) se usa el comando "***rm -r***".

Ejemplo:

dario@Leon:~/ejemplos/directorio1$ ls
copia dircopia diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$ rm -r dircopia
dario@Leon:~/ejemplos/directorio1$ ls
copia diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$

Algunos administradores deciden implementar seguridad para el borrado de archivos y directorios mediante una pregunta de validación, para eso se incluye la opción "***-i***" en la ejecución del comando.

Ejemplo:

dario@Leon:~/ejemplos/directorio1$ ls
copia diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$ rm -i copia
rm: remove regular empty file `copia'? y
dario@Leon:~/ejemplos/directorio1$ ls
diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$

NOTA: Funciona de forma similar con directorios pero agregándole la opción "-r" además de la opción "***-i***".

Ejercicios:

1.- Borre el archivo copiado en "/tmp" del ejercicio anterior, ¿escriba el comando qué usa?

2.- Ejecute "cd /tmp" y ahora ejecute "mkdir -p uno/dos/tres" ahora borre el directorio tres ¿Qué comando usa?, ¿cuál es el resultado?

3.- Borre el directorio "uno" ¿qué comando usa?, ¿cuál es el resultado?.

Moviendo o renombrando archivos o directorios

El comando "***mv***" ("move" en inglés) permite mover o renombrar archivos o directorios según sea la necesidad. La sintaxis del comando:

*"**mv fuente destino**"*

Hay una ligera diferencia entre el uso del comando cuando funciona para renombrar o mover archivos.

Renombrando archivos

Basta con colocar el nombre del archivo original y el nuevo nombre si lo que buscamos es renombrar.

Ejemplo:

dario@Leon:~/ejemplos/directorio1$ ls

diroriginal original subdirectorio1
dario@Leon:~/ejemplos/directorio1$ mv original copia
dario@Leon:~/ejemplos/directorio1$ ls
copia diroriginal subdirectorio1
dario@Leon:~/ejemplos/directorio1$

Moviendo archivos a un directorio distinto

En este caso la fuente será el archivo a mover y el destino será el directorio al cual vamos a mover el archivo.

Ejemplo:
dario@Leona:~/ejemplos$ ls
archivo archivocat archivocatfinal directorio2 directorio5 dir_umask1
archivo1 archivocat1 directorio1 directorio3 directorio6
archivo2 archivocat2 directorio1_ren directorio4 dir_umask
dario@Leona:~/ejemplos$ mv archivo directorio2/
dario@Leona:~/ejemplos$ ls
archivo1 archivocat1 directorio1 directorio3 directorio6
archivo2 archivocat2 directorio1_ren directorio4 dir_umask
archivocat archivocatfinal directorio2 directorio5 dir_umask1
dario@Leona:~/ejemplos$ ls directorio2/
archivo archivo1 subdir1
dario@Leona:~/ejemplos$

La opción "***-i***" alerta de sobrescribir mientras se mueve algún archivo.

Ejemplo:
dario@Leon:~/ejemplos$ ls
archivo directorio1 directorio2
dario@Leon:~/ejemplos$ mv -i archivo directorio1/
mv: overwrite `directorio1/archivo'? no
dario@Leon:~/ejemplos$

Renombrando directorios

Lo imprescindible es definir directorio fuente y directorio destino.

Ejemplo:
dario@Leon:~/ejemplos$ ls
archivo directorio1 directorio2
dario@Leon:~/ejemplos$ mv directorio1 directorio1_ren
dario@Leon:~/ejemplos$ ls

archivo directorio1_ren directorio2
dario@Leon:~/ejemplos$

Moviendo directorios y su contenido a un directorio distinto

Si hay necesidad de mover directorios completos, el comando "***mv***" permite realizar esta tarea, de nuevo es imprescindible definir la fuente y el destino.

Ejemplo:
dario@Leon:~/ejemplos$ ls
archivo directorio1_ren directorio2 directorio3
dario@Leon:~/ejemplos$ ls directorio2/
archivo3 archivo4
dario@Leon:~/ejemplos$ ls directorio3/
archivo archivo1 subdir1 subdir2
dario@Leon:~/ejemplos$ mv directorio2/ directorio3/
dario@Leon:~/ejemplos$ ls directorio3/
archivo archivo1 directorio2 subdir1 subdir2
dario@Leon:~/ejemplos$ ls
archivo directorio1_ren directorio3
dario@Leon:~/ejemplos$ mv directorio3/directorio2/ .
dario@Leon:~/ejemplos$ ls
archivo directorio1_ren directorio2 directorio3
dario@Leon:~/ejemplos$

Ejercicios:
Ejecute "cd /tmp" y ahora ejecute "mkdir -p cuatro/cinco/seis" y ahora "touch archivo1 archivo2"
1.- Renombre el archivo "archivo1" a "archivo1_t" ¿escriba el comando que usa?
2.- Renombre el directorio "cuatro" a "cuatro_t" ¿escriba el comando que usa?
3.- Mueva el archivo que creo en el ejercicio 1 al directorio cinco
4.- Renombre el "archivo2" a "archivo_new".

Creando directorios

Para esta tarea el comando "***mkdir***" ("make directory" en inglés) es el apropiado, basta con definir el nombre del directorio.

Con la opción "***-p***" permite crear subdirectorios.

Ejemplos:
dario@Leon:~/ejemplos$ mkdir directorio1 directorio4 directorio5
dario@Leon:~/ejemplos$ ls
archivo directorio1 directorio1_ren directorio2 directorio3 directorio4 directorio5
dario@Leon:~/ejemplos$ mkdir -p directorio6/subdirectorio1/subdirectorio2
dario@Leon:~/ejemplos$ ls -F directorio6/

subdirectorio1/
dario@Leon:~/ejemplos$ ls -lrt directorio6/
total 4
drwxr-xr-x 3 dario dario 4096 2009-10-30 15:52 subdirectorio1
dario@Leon:~/ejemplos$

Ejercicios:

1.- Vaya al directorio hogar de su usuario ("home directory" en inglés), y cree un directorio llamado "uno" y dentro de este un directorio llamado "dos". ¿qué comando uso para esto?.

2.- En el ejercicio anterior se creo el directorio "uno", si se pide crear el directorio "tres" dentro del directorio "dos" y dentro del directorio "tres" el directorio "cuatro". ¿Que opción se debería usar?.

Creando archivos

El comando "***touch***" (algo así como toque o nexo en inglés) permite crear archivos vacíos o actualizar el tiempo de modificación.

Ejemplo:

dario@Leon:~/ejemplos$ ls -l archivo
-rw-r--r-- 1 dario dario 0 2009-10-23 21:06 archivo
dario@Leon:~/ejemplos$ touch archivo
dario@Leon:~/ejemplos$ ls -l archivo
-rw-r--r-- 1 dario dario 0 2009-10-30 15:55 archivo
dario@Leon:~/ejemplos$ touch archivo_vacio1 archivo_vacio2 archivo_vacio3
*dario@Leon:~/ejemplos$ ls -l archivo**
-rw-r--r-- 1 dario dario 0 2009-10-30 15:55 archivo
-rw-r--r-- 1 dario dario 0 2009-10-30 15:56 archivo_vacio1
-rw-r--r-- 1 dario dario 0 2009-10-30 15:56 archivo_vacio2
-rw-r--r-- 1 dario dario 0 2009-10-30 15:56 archivo_vacio3
dario@Leon:~/ejemplos$

Ejercicios:

1.- Cree en el directorio hogar ("home directory" en inglés) tres archivos de nombre "arch1" "arch2" "arch3". ¿Escriba el comando que usa para esto?.

2.- Si ejecuta el comando "***touch***" sobre un archivo ya creado, ¿que sucede?.

3.- Si sobre un archivo que contiene data se usa el comando "***touch***", ¿se borrara su contenido?.

Creación de enlace simbólico

Los enlaces simbólicos sirven como su nombre lo indica para crear un enlace de un archivo o un directorio a otro, el enlace simbólico permite entre otras cosas proteger el archivo original ya que al momento de borrar el enlace solo se borrará éste y no el archivo o directorio original, los cuales permanecerán intactos, cualquier cosa que se realice sobre el archivo o directorio simbólico se verá reflejado en la ruta original. El comando para crear enlaces simbólicos es el comando "***ln***" (enlace en inglés).

La estructura del comando se muestra a continuación:

"ln -s fuente destino"

Ejemplo:

dario@Leon:~/ejemplos$ ln -s directorio3/subdir1/sub-subdir/archivo_fuente enlace_simbolico
dario@Leon:~/ejemplos$ ls -l enlace_simbolico
lrwxrwxrwx 1 dario dario 45 2009-10-30 16:13 enlace_simbolico -> directorio3/subdir1/sub-subdir/archivo_fuente
dario@Leon:~/ejemplos$ ls -l directorio3/subdir1/sub-subdir/archivo_fuente
-rw-r--r-- 1 dario dario 93 2009-10-30 16:12 directorio3/subdir1/sub-subdir/archivo_fuente
dario@Leon:~/ejemplos$

NOTA: Como se puede verificar el enlace simbólico comienza con la letra "l", el tamaño del enlace simbólico es fijo y difiere del tamaño original del archivo o directorio fuente, el tamaño real es el número de caracteres que forman el nombre del archivo o directorio fuente. El tamaño del enlace simbólico no crecerá, por eso con frecuencia estos se colocan en sistemas de archivos ("filesystems" en inglés) que no tienen mucho espacio disponible.

Ejercicios:

1.- Cree un archivo vacío y ahora cree el respectivo enlace simbólico. ¿Qué comandos se usa para esto?.

2.- Añada la palabra "hola" al enlace simbólico. Verifique el archivo original, ¿qué sucedió?.

3.- ¿Si sobre un directorio creo varios subdirectorios, estos se conservan de forma igual sobre el original y sobre el enlace simbólico?.

4.- ¿Que sucede si borro el archivo original y no el enlace simbólico?, muestre los resultados de sus pruebas.

Manipulando el shell

¿Qué es un shell? Un shell es un interpretador de comandos o un macro procesador lo que permite expandir textos y símbolos para crear expresiones más largas.

Un shell permite ser ejecutado de forma interactiva cuando acepta comandos directamente desde el teclado, no interactivos cuando aceptas comandos que provienen de un archivo.

"Existen varios tipos de shells en sistemas de código abierto, entre ellos: "Bash" ("Bourne-again shell"), "Ksh" ("Korn-Shell"), "csh" ("C shell"), "tsh" ("TC shell"), "sh" ("Bourne Shell"). Los que esten por defecto o no depende de la distribución. Acá nos enfocamos en "Bash" que es el más popular"[3].

Configurando una variable

"Var=valor, export Var"

---

[3] http://es.kioskea.net/contents/unix/unix-shell.php3

Ejemplo:

dario@Leona:~/ejemplos$ HOLA="HOLA MUNDO"
dario@Leona:~/ejemplos$ export HOLA
dario@Leona:~/ejemplos$ echo $HOLA
HOLA MUNDO
dario@Leona:~/ejemplos$

Ejercicios:

1.- Cree una variable de nombre "Prueba" que tenga la palabra "prueba" dentro de este. ¿Qué comando uso?.

2.- Averigue cual es el valor de la variable HOME, liste el comando y como interroga a la variable.

3.- ¿El nombre de una variable puede estar separada por espacios?.

Cuál es el shell que estamos usando

Como se explico con anterioridad, para mostrar el valor de una variable se usa el comando "***echo***", vale la pena destacar que para crear una variable usamos "var=valor", para ver dicha variable se debe usar el signo de dolar "$" seguido del nombre que se le de a la variable.

Ejemplo:
dario@Leon:~/ejemplos$ echo $SHELL
/bin/bash
dario@Leon:~/ejemplos$

dario@Leon:~/ejemplos$ ps
PID TTY TIME CMD
10033 pts/6 00:00:00 ps
30254 pts/6 00:00:01 bash

Un camino alterno para determinar el "shell" que se esta usando sobre el terminal, es el comando "***ps***" ("process status" en inglés), tal como se muestra en el ejemplo anterior.

Ejercicios:

1.- Coloque el comando que debe usar para identificar el "shell" que esta usando.

2.- Cambie al "shell Bourne Shell". Escriba el comando que usa para eso.

Desconfigurando una variable

Para hacer que la variable deje de tener el valor que hemos asignado se usa el comando "***unset***" (desconfigurar), el formato del comando:

"***unset Var***"

Ejemplo:

dario@Leon:~/ejemplos$ echo $SHELL
/bin/bash
dario@Leon:~/ejemplos$ unset SHELL
dario@Leon:~/ejemplos$ echo $SHELL
dario@Leon:~/ejemplos$

Para desplegar el nombre de las variables y sus valores, así como las funciones uno de los comandos que se usa el comando "***set***".

Algunas variables que están predefinidas en los "shell":

Variable	Valor
HOME	Coloca el valor producto del uso del comando cd siempre Que sea sin argumentos
LOGNAME	Usuario con el que se esta trabajando
SHELL	Indica el shell que se esta usando
PATH	Ejecutados por necesidad del shell
PS1	Especifica el prompt principal
PS2	Prompt secundario

Ejercicios:

1.- Muestre el valor de las variables "HOME", "LOGNAME", "PS1" y "PS2" de la máquina en la que esta trabajando, escriba los resultados y el comando que uso.

2.- Desconfigure la variable "PS2", escriba el comando que uso, cuales son las consecuencias de desconfigurar dicha variable sobre el sistema.

Personalizando el uso de los shells

"Prompt"

El "PS1" ("prompt string 1" en íngles) algo así como cadena de caracteres en español.

Se conoce un "prompt" como uno o varios caracteres que aparecen en una computadora para indicar que el sistema operativo esta disponible para empezar a recibir comandos, todos los sistemas operativos cuentan con uno o varios "prompt", algunos de error, otros de información, etc.

Para definir el "PS1 prompt" usamos la definición de una variable como se definió en la sección anterior.

Ejemplo:
dario@Leona:~$
dario@Leona:~$PS1="Este es el prompt$ "
Este es el prompt$

El manejo del prompt "PS1" permite añadir caracteres de escape especial, algunos son:

\d la fecha en formato día mes año (ej., "mié jul 02")
\h el nombre del host hasta el primer "."
\H el nombre del la máquina completo (FQDN)
\j el número de trabajos actualmente gestionados por el interprete
\t la hora actual en formato 24-horas HH:MM:SS
\u el nombre del usuario actual
\w el directorio actual de trabajo
\! el número del comando actual en el histórico
\$ si el UID efectivo es "0", un "#"; en otro caso, "$"

Ejemplo:

dario@Leona:~$ echo $PS1
"\[\e]0;\u@\h: \w\a\]${debian_chroot:+($debian_chroot)}\u@\h:\w\$"
dario@Leona:~$

NOTA: Recuerda usar siempre las dobles comillas para definir el "PS1".[4]

Ejercicios:

1.- Cambie el prompt "PS1" de su terminal y agregue: la hora actual, escriba el comando que uso.
2.- Cambie el prompt "PS1" de su terminal y agregue: el nombre del usuario actual, escriba el comando que uso.

"PATH"

Define las rutas donde se encuentran los ejecutables y comandos definidos por el usuario, esto hace más fácil la vida al usuario ya que ahorra escribir la ruta completa para ejecutar un comando.

Ejemplo:

dario@Leon:/$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
dario@Leon:/$ /bin/ls
bin cdrom etc .....
dario@Leon:/$

Como se aprecia en el ejemplo anterior si no se tiene definido la ruta "/bin", se debe ejecutar el comando "***ls***" con su ruta completa, para incluir nuevas rutas se hace como sigue a continuación:

dario@Leon:/$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
dario@Leon:/$ PATH=$PATH:/etc
dario@Leon:/$ echo $PATH

---

[4] http://wiki.archlinux.org/index.php/Color_Bash_Prompt_(Espa%C3%B1ol)

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/etc
dario@Leon:/$

NOTA: En caso de desconocer la ruta para ejecución de algún comando se ejecuta el comando "***which***".

Ejemplo:
dario@Leon:/$ which find
/usr/bin/find
dario@Leon:/$

Ejercicios:
1- Añada el directorio hogar ("home directory" en inglés) como el "path" de su maquina. ¿Qué comando usa?.
2.- ¿Que traduce el "." al ser aplicado sobre el "path".

history

Muestra como su nombre lo indica, un histórico de los comandos ejecutados sobre el shell.

Metodos de completación en bash

Para completar una ruta se usa la tecla <tab> (tabulador), para buscar dentro del historial de comandos se usan la teclas de cursor.

Comandos de redirección

En los sistemas operativos se puede redireccionar la entrada o la salida de comandos usando los metacaracteres "<" o ">" o el metacaracter "|". Sobre los sistemas operativos de código abierto la entrada por defecto es el teclado, la salida por defecto y los errores son la pantalla o monitor.

Cada archivo de redirección ("filedescriptor" en inglés) debe tener un número asociado en el sistema operativo, algunos archivos de redirección:

Abreviación del file descriptor	Valor	Metacaracter asociado
stdin (Entrada)	0	0<
stdout (Salida)	1	1>
stderror (Error)	2	2>

Redireccionando la salida a un archivo
Ejemplo:
dario@Leona:~$ ls > /tmp/prueba

dario@Leona:~$ more /tmp/prueba
Desktop
dewd
Documents
downloads
Music

Redireccionando los errores por defecto a un archivo.

Ejemplo:
dario@Leona:~/ejemplos/directorio2$ ls; ls /temp 2>salida
archivo archivo1 salida subdir1
dario@Leona:~/ejemplos/directorio2$ more salida
ls: cannot access /temp: No such file or directory
dario@Leona:~/ejemplos/directorio2$

NOTA: En el ejemplo se puede observar que la salida del error es redireccionada por defecto al archivo "salida" por el uso del caracter de redirección.

Se puede usar el archivo de redirección de errores para mostrar la salida de errores y salida estándar juntas.

Ejemplo:
dario@bt:~/dario$ ls . opto > /tmp/redireccion 2>&1
dario@bt:~/dario$ more /tmp/redireccion
ls: cannot access opto: No such file or directory
.:
hw_breakpoint
Kconfig
kobject
kprobes
Makefile
trace_events
tracepoints

Vemos en el ejemplo como la salida y los errores son enviados a un archivo, para lograr esta redirección usamos los archivos de redirección representados por los números "2" y "1" ademas el caracter especial "&".

Ejercicios:
1.- Del ejemplo de anterior, ¿como puedo redireccionar solo los errores?.
2.- ¿Como se puede redireccionar un dispositivo de entrada?.

Redirección (pipe)

El caracter especial "pipe |" (tubería en español) se usa para redirigir la salida de un comando a la entrada de otro comando. La analogía en este caso es a las tuberías de casas o edificios, funciona de la misma manera en la que se trabaja para encausar el agua.

Ejemplo:
dario@Leon:/$ ls -l | more
acpi
adduser.conf
adjtime
aliases
aliases.db
alsa
alternatives
--More--

En el ejemplo, se observa, la salida del comando "***ls***" es muy extensa por tal motivo es necesario mostrarlo en pasos de pantalla sucesivas, para conseguir esto, se redirecciona su salida al comando "***more***" que permitirá que la salida sea manejada pantalla por pantalla.

Ejercicios:
1.- Escriba a continuación dos ejemplos de redirecciones usando "pipe".
2.- Muestre como usar el caracter "|" para contar el número de líneas producto de usar el comando "***ls***".

Archivos de inicialización

Los archivos de inicialización en los sistemas operativos de código abierto permiten cargar información para el usuario tal como atajos de comandos (alias), funciones, información como las variables "PATH", "EDITOR", etc. Esto ocurre al momento de entrar en la sesión del usuario o al usuario ejecutarlo directamente en el terminal. A continuación los principales para el "shell bash".

Shell	Archivo de inicialización para todos los usuarios	Archivo de inicialización primario para el usuario al iniciar sesión	Archivo de inicialización cuando un shell inicia
Bash	/etc/profile	$HOME/.profile	$HOME/.bashrc

Sobre los archivos de inicialización primaria se acostumbra a colocar información tal como las variables "PATH", "LOGNAME", etc.

Sobre los archivos de inicialización, cuando el "shell" ya ha comenzado, se define información como: "funciones", "alias", "prompt" por defecto o "PS1", etc.

Definición de alias

Un "alias" sobre el sistema operativo ayuda a definir atajos para la definición de comandos, haciendo más fácil para el usuario final recordar el "alias" al que esta acostumbrado, quizás en otro sistema operativo o que le sea mas fácil recordarlo en su vida cotidiana.

Ejemplo:

```
dario@Leona:~/ejemplos/directorio2$ alias dir='ls -l'
dario@Leona:~/ejemplos/directorio2$ dir
total 8
-rw------- 1 dario dario    0 2010-02-11 17:45 archivo
-rw------- 1 dario dario    0 2010-03-27 16:04 archivo1
-rw-r--r-- 1 dario dario   51 2010-03-30 19:45 salida
drwx------ 3 dario dario 4096 2010-02-11 17:45 subdir1
dario@Leon:/$ alias cls=clear
```

Ejercicios:

1.- Escriba un "alias" para que al momento de borrar un archivo se pregunte si se desea borrar. Muestre a continuación la secuencia de pasos que uso.

2.- Escriba un "alias" que muestre la hora actual y muestre la secuencia de pasos que uso.

Cambiando permisos a archivos y directorios

Para hacer el cambio a permisos y directorios sobre los sistemas operativos de código abierto se usa el comando "***chmod***" ("change mode" en inglés), este puede ser usado en modo octal o en modo simbólico:

Modo octal y su representación

Representado por números, los números van con valores binarios de derecha a izquierda y el más alto representa la lectura hasta el más bajo que representa la ejecución, a continuación una tabla resumen del modo octal y el permiso que otorga:

Modo octal	Permiso
4	Lectura
2	Escritura
1	Ejecución

Modo simbólico y su represtación

El modo simbólico permite mediante el uso de signos algebraicos y de letras cambiar la permisología de un archivo o directorio. A continuación una tabla resumen con información de a quién se le cambia el permiso, de que forma se le cambia (opción) y que permiso se cambia:

Quien	Opción	Permisos
Usuario=u	+ ó -ó =	Lectura=r, Escritura=w, Ejecución=x
Grupo=g	+ ó -ó =	Lectura=r, Escritura=w, Ejecución=x
Otros usuarios=o	+ ó -ó =	Lectura=r, Escritura=w, Ejecución=x
Todos=a	+ ó -ó =	Lectura=r, Escritura=w, Ejecución=x

Modo octal

Para su ejecución se usan números que sumen el tamaño de los distintos caracteres para permisología sobre un archivo o directorio, esos caracteres son "rwx", los valores son listados en la sección anterior.

Ejemplos:

Sobre archivos:

dario@Leon:~/ejemplos$ ls -l archivo
-rw-r--r-- 1 dario dario 11 2009-11-03 20:34 archivo
dario@Leon:~/ejemplos$ chmod 755 archivo
dario@Leon:~/ejemplos$ ls -l archivo
-rwxr-xr-x 1 dario dario 11 2009-11-03 20:34 archivo
dario@Leon:~/ejemplos$

En el ejemplo la permisologia usada sobre el archivo "archivo" es para el usuario (propietario) lectura, escritura y ejecución, para el grupo propietario es lectura y ejecución y de igual manera para el resto de usuarios (other).

Sobre directorios:
dario@Leon:~/ejemplos$ ls -ld directorio1
drwxr-xr-x 2 dario dario 4096 2009-10-30 15:51 directorio1
dario@Leon:~/ejemplos$ chmod -R 444 directorio1
dario@Leon:~/ejemplos$ ls -ld directorio1
dr--r--r-- 2 dario dario 4096 2009-10-30 15:51 directorio1
dario@Leon:~/ejemplos$

La opción "***-R***" permite que sea aplicado de forma recursiva es decir del directorio donde se está hacia abajo. El ejemplo muestra que la permisología aplicada al directorio "directorio1" es para el usuario propietario, para el grupo propietario y para el resto de usuarios es de solo lectura.

Ejercicios:

1.- Cree un archivo y un directorio. Escriba los comandos que usaría. Ahora cambie la permisología y escriba a continuación los comandos que usó para cada caso:

1.1.- Solo ejecución en el archivo:

1.2.- Solo listar el contenido del directorio:

1.3.- Poder editar el contenido del directorio:

1.4.- Todos los permisos para el archivo:

Modo simbólico

Para la ejecución de este modo es necesario tener claro a que corresponde cada caracter especial.

Ejemplo:

Sobre archivos:

dario@Leon:~/ejemplos$ ls -l archivo2
-rw-r--r-- 1 dario dario 0 2009-11-03 21:40 archivo2
dario@Leon:~/ejemplos$ chmod u+rw,g+r,o+rw archivo2
dario@Leon:~/ejemplos$ ls -l archivo2
-rw-r--rw- 1 dario dario 0 2009-11-03 21:40 archivo2
dario@Leon:~/ejemplos$

En el ejemplo anterior podemos ver que se le añadió a la permisología del archivo "archivo2" a "user" (propietario) lectura y escritura a "group" (grupo propietario) solo lectura y "other" (resto de usuarios) lectura y escritura.

Sobre directorios:

dario@Leon:~/ejemplos$ ls -ld directorio2/
drwxr-xr-x 2 dario dario 4096 2009-10-30 15:46 directorio2/
dario@Leon:~/ejemplos$ chmod -R u+rw,g+r,o+rw directorio2
dario@Leon:~/ejemplos$ ls -ld directorio2/
drwxr-xrwx 2 dario dario 4096 2009-10-30 15:46 directorio2/
dario@Leon:~/ejemplos$

En el ejemplo anterior se añadió a la permisología del directorio "directorio2" a "user" (propietario) lectura y escritura al grupo (grupo propietario) sólo lectura y al resto del mundo (resto de usuarios) lectura y escritura.

Ejercicios:

1.- Cree un archivo y un directorio, Escriba los comandos que usaría. Ahora cambie la permisología y escriba como cambiarla, para los siguientes casos:

1.1.- Sólo ejecución en el archivo:

1.2.- Sólo listar el contenido del directorio:

1.3.- Poder editar el contenido del directorio:

1.4.- Todos los permisos para el archivo:

Modificando permiso por defecto sobre archivos y directorios

Cada archivo o directorio se crea con una permisología básica sobre los sistemas operativos tipo *nix, para todos los archivos creados en GNU/Linux y Unix el valor por defecto en octal es 666 y para todos los directorios es 777, el sistema operativo debe decidir con que permisología por defecto serán creados, es trabajoso, poco organizado para uno o varios usuarios aplicar una permisología distinta a cada archivo o directorio que cree nuevo e informar de esta al resto de usuarios, quien decide esta permisología sobre todo el ancho del sistema operativo es el comando de nombre "***umask***".

Ejemplo:

```
dario@Leon:~/ejemplos$ umask
0022
dario@Leon:~/ejemplos$ umask 077
dario@Leon:~/ejemplos$ umask
0077
dario@Leon:~/ejemplos$ touch archivo_umask1
dario@Leon:~/ejemplos$ ls -l archivo_umask1
-rw------- 1 dario dario 0 2009-11-03 21:47 archivo_umask1
dario@Leon:~/ejemplos$ mkdir dir_umask1
dario@Leon:~/ejemplos$ ls -ld dir_umask1
drwx------ 2 dario dario 4096 2009-11-03 21:47 dir_umask1
dario@Leon:~/ejemplos$
```

En el ejemplo anterior podemos determinar que si ejecutamos el comando sin opciones ni argumentos, el valor que se despliega es 0022 el primer 0 antes es generalmente desplegado en GNU/Linux y refiere que se esta haciendo un despliegue de la mascara ("umask" en inglés) en modo octal. El valor de la aplicación del comando "***umask***" son los mismos valores de modo octal para cambio de permisos de archivos y directorios en la pagina 36 del presente documento. Para ser aplicado como indica el ejemplo es "***umask***" seguido por el valor que quiera ser aplicado, el valor a ser aplicado resulta de la aplicación de la negación de la mascara y la aplicación de un "and" lógico entre la negación y el valor por defecto. Como se muestra a continuación:

En el ejemplo el valor que se aplica sobre archivo es 077 y el valor por defecto del archivo es 666. transformando a binario queda:

"umask"	000 111 111
"not umask"	111 000 000
Archivo (por defecto)	110 110 110
Valor final	110 000 000 esto traduce rw-------

Ejercicios:

1- Manipule el "***umask***" para poder crear archivos que solo tengan permiso de lectura. Escriba el comando y como se verán afectados los nuevos directorios que se creen una vez realizado el cambio.

2.- ¿Si quiero aplicar una mascara que permita máxima seguridad a archivos y directorios, cual seria?.

Buscando coincidencias de patrones en archivos y directorios

Para buscar información sobre el contenido de archivos que coincidan con un patrón se usa el comando "***grep***".

Para buscar un archivo o directorio sobre el sistema operativo se usa el comando "***find***".

"grep"

"grep" busca en la entrada estandar o en la que especifica con nombres de ficheros líneas que concuerden o coincidan con un patrón dado. Si no se dice otra cosa "***grep***" muestra las líneas que concuerden.

Usa un algoritmo compacto no deterministico.".[5]

La sintaxis del comando es:

"grep opción patrón nombre_del_archivo"

Algunas de las opciones disponibles para el comando "***grep***"

Opción	Descripción
-l	Lista el nombre de archivo del patrón de ocurrencia
-i	Realiza la búsqueda por minúsculas o mayúsculas
-n	Coloca el número de línea de cada ocurrencia
-v	Imprime las líneas que no corresponden a la búsqueda del patrón
-c	Contador de líneas que contienen el patrón

Ejemplos:

dario@Leon:/etc$ grep -n dario group
5:adm:x:4:dario
15:dialout:x:20:dario
18:cdrom:x:24:dario
35:plugdev:x:46:dario
45:lpadmin:x:106:dario
60:admin:x:121:dario
61:dario:x:1000:
62:sambashare:x:122:dario
dario@Leon:/etc$

La opción "***-n***" coloca la línea en la que aparece el patrón.

---

[5] http://manpages.unixforum.co.uk/man-pages/unix/solaris-10-11_06/1/grep-man-page.html

dario@Leon:/etc$ grep -v localhost hosts

 # The following lines are desirable for IPv6 capable hosts
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
dario@Leon:/etc$

En el ejemplo anterior se imprimen las líneas que no contienen el patrón "localhost" en el archivo "hosts".

El comando grep soporta metacaracteres (para construir expresiones regulares). A continuación algunos de esos metacarecteres que pueden ser usados:

Patrón	Estructura	Uso	Resultado
^	'^item'	Comienzo de línea	Encuentra todas las líneas que comiencen con ítem.
$	'item$'	Final de línea	Encuentra todas las líneas que finalicen con ítem.
.	'i..m'	Caracter cualquiera	Encuentra las que líneas que contengan el ítem que comience con i seguido de 2 caracteres cualesquiera y finaliza con m.
[]	'[0123]'	Estructura de soporte	Coincidencia de un solo caracter por lo menos.
*	'*item'	Coincidencia una o mas veces	El ítem que precede coincida una o mas veces.

Ejemplos:
dario@Leon:~/ejemplos$ grep '^dario' archivogrep
dario leon
dario@Leon:~/ejemplos$
dario@Leon:~/ejemplos$ grep 'persona$' archivogrep
Otra persona
dario@Leon:~/ejemplos$
dario@Leon:~/ejemplos$ grep '[1]' archivogrep
coincidencia de 1 o mas veces
dario@Leon:~/ejemplos$

En los ejemplos anteriores vemos que:

Se imprime la línea donde aparece "dario" al principio de la línea en el archivo "archivogrep".

Se imprime la línea donde aparece "persona" al final de la línea en el archivo "archivogrep".

Se imprime la línea donde aparece el número "1" en cualquier parte de la línea en el archivo "archivogrep".

Ejercicios:

1.- Muestre a continuación como debería ser realzada la busqueda por la ocurrencia de la palabra "hosts" sobre el archivo "/etc/nsswitch.conf".

2.- Muestre a continuación como debería buscar por la ocurrencia de la palabra "hosts" a principio de la línea sobre el archivo "/etc/nsswitch.conf".

"find"

Dicho comando (buscar, sería una traducción del comando) permite buscar por la coincidencia de un archivo sobre un directorio, grupos de directorios o todos los sistemas de archivos.

La sintaxis del comando es como sigue:

"find ruta expresión acción"

Ruta: refiere a donde comenzaremos la búsqueda

Expresión: refiere al criterio de búsqueda que estaremos aplicando, usando múltiples expresiones utiliza la expresión boleana y ("and" en inglés).

A continuación algunas expresiones posibles:

Expresión	Uso
(+ ó -)n	+ mayor que, - menor que, n igual a.
-name nombre_de_archivo	-name es el más usado y refiere a la búsqueda por nombre de archivo.
-atime (+ ó -)n	Búsqueda por el tiempo de acceso a un archivo +n mayor al número de días, -n menor al número de días, n igual al número de días.
-mtime (+ ó -)n	Búsqueda por el tiempo de modificación de un archivo +n mayor al número de días, -n menor al número de días, n igual al número de días.
-user name	Propietario del archivo.

Acción: refiere a que se ejecutará sobre un archivo, a continuación una tabla con algunas posibles ejecuciones:

Acción	Uso
-exec command {} \;	Ejecutar un comando sobre cada coincidencia del archivo lo delimita el uso de las llaves {}, delimita el comando el uso del signo \; (un espacio entre llaves y backslash es necesario)
-ok command {} \;	Sobre las coincidencias del archivo serán ejecutados el comando pero de forma interactivo, el resto de la construcción del comando será como el apartado anterior.
-ls	Lista la ruta de la coincidencia.

Algunos ejemplos:

dario@Leon:~$ find . -name archivofind
./ejemplos/directorio2/subdir1/suddir2/subdir3/archivofind
./ejemplos/directorio6/subdirectorio1/subdirectorio2/archivofind
find: `./.config/menus': Permission denied
dario@Leon:~$

En la salida anterior vemos que se realiza la búsqueda del archivo de nombre "archivofind" sobre el directorio donde estoy posicionado.

dario@Leon:~$ find . -atime +100 -ls
find: `./.config/menus': Permission denied
276965 16 -rw-r--r-- 1 dario dario 14420 Apr 14 2009 ./.openoffice.org/3/user/config/web.soc
276969 16 -rw-r--r-- 1 dario dario 13132 Apr 14 2009 ./.openoffice.org/3/user/config/cmyk.soc
276959 4 -rw-r--r-- 1 dario dario 3919 Apr 14 2009 ./.openoffice.org/3/user/config/sun-color.soc
276972 8 -rw-r--r-- 1 dario dario 4408 Apr 14 2009 ./.openoffice.org/3/user/config/gallery.soc
276971 12 -rw-r--r-- 1 dario dario 10766 Apr 14 2009 ./.openoffice.org/3/user/config/html.soc
276960 48 -rw-r--r-- 1 dario dario 48408 Apr 14 2009 ./.openoffice.org/3/user/config/autotbl.fmt
276946 4 -rw-r--r-- 1 dario dario 567 Apr 14
dario@Leon:~$

La búsqueda del ejemplo es por todos los archivos o directorios que tengan más de 100 días de accedidos en la carpeta en donde nos encontramos.

```
dario@Leon:~/ejemplos$ find . -perm u=rw- -ls
383154   0 -rw-------   1 dario   dario        0 Nov  7 18:27
./directorio2/subdir1/suddir2/subdir3/archivofind
383143   4 -rw-------   1 dario   dario       77 Nov  7 17:12 ./archivogrep
383433   0 -rw-------   1 dario   dario        0 Nov  3 21:47 ./archivo_umask1
383155   0 -rw-------   1 dario   dario        0 Nov  7 18:27
./directorio6/subdirectorio1/subdirectorio2/archivofind
dario@Leon:~/ejemplos$
```

En el ejemplo anterior se buscan todos los archivos o directorios que tengan permisos de lectura y escritura sobre la carpeta donde estamos posicionados.

```
dario@Leon:~/ejemplos$ find . -exec grep -l dario {} \;
./archivogrep
./archivo
dario@Leon:~/ejemplos$
```

La construcción de la estructura anterior permite realizar la localización de todos los archivos que contengan el patrón "dario" en su contenido.

```
dario@Leon:~/ejemplos$ find . -exec rm archivo {} \;
```

Remueve del sistema operativo todos los archivos de nombre "archivo" desde el directorio donde esta el usuario en adelante sin ser interactivos.

Ejercicios:

1.- Realice una búsqueda desde la raíz para localizar todos los archivos que tengan un tamaño mayor a 2192 kbytes.

2.- Realice una búsqueda de todos los archivos que en el sistema operativo tengan en su contenido la palabra "hosts". ¿Como sería la estructura del comando?.

3.- Realice una búsqueda de todos los archivos que tengan por nombre "hosts" e imprimalos. Suprima la salida de los errores por defecto. ¿Cómo sería la estructura del comando?.

Comandos para administrar procesos

"ps"

El comando "***ps***" ("process status" en inglés) muestra información acerca de los procesos que estén siendo usados sobre el sistema operativo en espacio kernel y espacio usuario, si el comando se ejecuta sin opciones mostrara información de los procesos que están corriendo sobre el terminal donde se ejecutó, la información que se despliega si se ejecuta el comando sin opciones, es como sigue:

"PID" ("process identifier" en inglés) de los procesos, el número de "pseudo" terminal, el

tiempo acumulativo (valor despreciable ya que no cambia en el tiempo en la mayoría de los sistemas operativos de código abierto) y el comando.

Ejemplo:
dario@Leon:~$ ps
PID TTY TIME CMD
18562 pts/6 00:00:00 ps
20037 pts/6 00:00:00 bash
dario@Leon:~$

Si se usa con la opción "***-e***" nos despliega una salida igual a la anterior pero de todos los comandos que están siendo ejecutados sobre el sistema operativo.

Si se usa con la opción "***-f***" nos mostrara el detalle completo, incluyendo opciones como el "PPID" y el dueño del proceso.

Ejercicios:
1.- Ejecute el comando "ps -ef" y explique que muestra cada columna.

"pgrep"

El comando "***pgrep***" (por sus siglas en inglés "**p**rocess **g**lobal **r**egular **e**xpression **p**rint"), busca por la ocurrencia de un proceso.

Generalmente se puede usar con las opciones "***-lf***" para ver la ruta completa del comando y su "PID" ("process identifier" en inglés).

Ejemplo:
dario@Leona:~$ pgrep -lf firefox
3905 /usr/lib/firefox-3.0.8/firefox
dario@Leona:~$

Ejercicios:
1.- Busque usando el comando "***pgrep***" por las opciones de "mail" que estan siendo usadas en su maquina. ¿Escriba cómo sería el comando?.

"kill"

"Kill" (matar es la traducción apropiada) permite enviar una señal a un comando, se usa el comando "***kill***" y la opción de ser necesario, seguido del PID ("process identifier" en inglés), el detalle acerca de las señales posibles lo encontramos ejecutando "***kill -l***".

Ejemplo:
dario@Leona:~$ pgrep -lf firefox
3905 /usr/lib/firefox-3.0.8/firefox
dario@Leona:~$

dario@Leona:~$ kill 3905
dario@Leona:~$ pgrep -lf firefox
dario@Leona:~$

De la salida del comando "***pgrep***" su "PID" (3905) y al aplicar el comando "***kill***" sobre este "PID" ("process identifier" en inglés). El comando "***kill***" matará (eliminará) el proceso, sin opciones la señal por defecto será la señal 15, si deseamos que el proceso se detenga de inmediato usamos la señal 9.

Ejercicios:

1.- Abra un nuevo terminal y busque por línea de comandos el ID del terminal que se abrió antes, a continuación elimine este. Escriba los pasos que uso y que resultado obtuvo, coloque además que señal debe usar para cerrar el terminal si en la primera oportunidad no se cerro.

"pkill"

Comando "***pkill***" (matar o detener un proceso) ejecuta la misma tarea del comando "***kill***" pero en vez del "PID" ("process identifier" en inglés) del proceso se usa el nombre del proceso.

Ejemplo:

```
dario@Leona:~$ pkill firefox
dario@Leona:~$ pgrep -lf firefox
dario@Leona:~$
```

Ejercicios:

1.- Ejecute el comando "***mail***", a continuación envie la señal 15 a la aplicación, usando el comando "***pkill***". Escriba los pasos que usó y el resultado.

Editar texto

Editor de texto vi

El editor de texto "***vi***" (view) ver en inglés es el editor por excelencia de los administradores de sistemas operativos *nix a nivel mundial. Este editor es usado sin consumir recursos adicionales sobre el sistema operativo, si se arranca el sistema en modo "single user" y necesitamos realizar una edición de algún archivo, el editor vi es perfecto para esto. Una recopilación puede ser encontrado en mi blog personal:

http://www.cacaoadmin.com/2010/04/el-editor-de-texto-vi.html

Bibliográfia

- Hanbelrt Dan, The early History of the more command [En línea] Disponible en: http://www.halwitz.org/halbert/more.html [Consultado el 20 de diciembre de 2009].
- Less Home page [En línea] Disponible en: http://www.greenwoodsoftware.com/less/index.html [Consultado el 3 de enero de 2010].
- Sistema Unix: El shell, Introducción al shell [En línea] Disponible en: http://es.kioskea.net/contents/unix/unix-shell.php3 [Consultado el 4 de marzo de 2010].
- Color Bash Prompt (En español) [En línea] Disponible en: http://wiki.archlinux.org/index.php/Color_Bash_Prompt_(Espa%C3%B1ol) [Consultado el 10 de mayo de 2010].
- Grep [En línea] Disponible en: http://manpages.unixforum.co.uk/man-pages/unix/solaris-10-11_06/1/grep-man-page.html [Consultado el 23 de enero de 2012].
- Grep [En línea] Disponible en: http://unixhelp.ed.ac.uk/CGI/man-cgi?grep [Consultado el 23 de enero de 2012].
- Solter, Nicholas A, Gerald Jelinek, David Miner. Opensolaris Bible, Indianapolis, Indiana: Wikey Publishing, Inc. 2009.