



Curso Operador Linux

Módulo 4

ADMINISTRACION DE ARCHIVOS Y DIRECTORIOS



Presentación

En esta unidad se estudiara como realizar la administración de archivos y directorios



Objetivos

Los participantes al finalizar la Unidad:

- Comprender la estructura de directorios del sistema
- Montar dispositivos
- Utilizar los comandos básicos relacionados con archivos y directorios
- Visualizar y modificar contenidos de archivos y directorios



Temario

- 4.1 Sistemas de archivos
- 4.2 Montaje de dispositivos
- 4.3 Organización de archivos
- 4.4 Como copiar archivos
- 4.5 Como mover y renombrar archivos
- 4.6 Como eliminar archivos y directorios
- 4.7 Visualización de un archivo
- 4.8 Como buscar archivos



Actividad de aplicación del conocimiento

Los participantes encontraran la actividad de aplicación en un archivo por separado en caso de ser requerida.



Examen

Los participantes deberán rendir el examen online o presentar el material solicitado según corresponda.



4.1 Sistemas de archivos

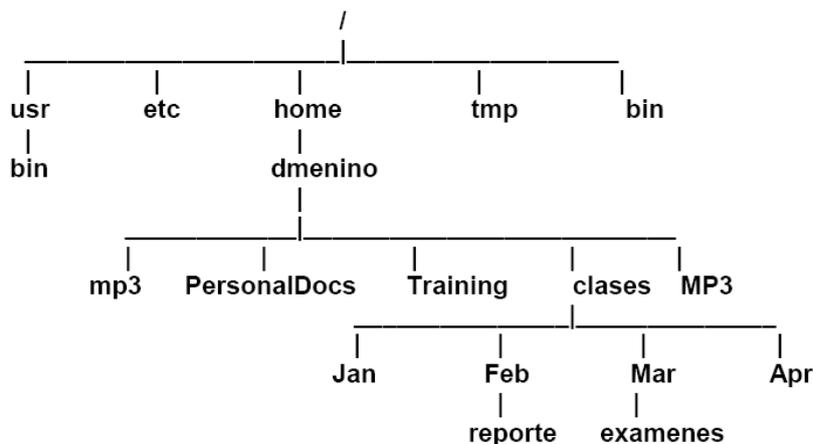
El sistema de archivos de Linux es similar al de UNIX y DOS

- Es un directorio en forma de árbol
- Utiliza paths para especificar directorios y archivos
- Los directorios y archivos se separan con la barra - slash (/)

A diferencia de DOS

- Todo está bajo una rama, comenzando por el directorio raíz o root (/)
- Los dispositivos no se identifican con letras sino que son parte del árbol

La organización de directorios puede ejemplificarse de la siguiente manera:



Linux distingue entre mayúsculas y minúsculas (es case-sensitive). En el ejemplo anterior el directorio mp3 no es el mismo que el nombrado MP3, son dos directorios distintos.



```
debian:/# cd /home/dmenino/  
debian:/home/dmenino# tree  
.  
|-- MP3  
|-- PersonalDocs  
|-- Training  
|-- clases  
|   |-- Apr  
|   |-- Feb  
|   |-- `-- reporte  
|   |-- Jan  
|   |-- Mar  
|   |-- `-- examenes  
|-- mp3  
  
9 directories, 2 files  
debian:/home/dmenino#
```

Hay dos maneras de especificar una ruta o path

Absoluta

Comienza con "/". Se indica desde el directorio raíz y se especifica en forma absoluta desde allí.

| | |
|----------------|--|
| / | Directorio raíz |
| /usr | Directorio usr del sistema |
| /bin | Directorio bin del sistema |
| /usr/bin | Directorio bin para las aplicaciones del usuario |
| /home/dmenino/ | Directorio del usuario dmenino |

Relativa

No comienza con "/". Se indica en forma relativa a partir del lugar donde estamos posicionados.

Suponiendo que estamos posicionados en el directorio dmenino para indicar el directorio exámenes usaríamos clases/Mar/exámenes

Para referenciar el directorio mp3 usaríamos mp3

La versión absoluta de los anteriores serían /home/dmenino/clases/Mar/exámenes y /home/dmenino/mp



Permisos de archivos



Existen comandos en Linux que se utilizan para hacer cambios en los permisos de los archivos y directorios. Para realizar cualquier cambio sobre los permisos de los archivos, es completamente necesario tener permisos para realizar estas tareas.

Los permisos (así como los dueños y grupos) de los archivos y directorios se pueden ver con el comando `ls` utilizando las opciones apropiadas.

El comando básico para listar archivos es el `ls`. La forma en que `ls` despliegue los archivos depende de cómo se utilice el comando.

Comando: `ls`

Uso: `ls [opciones]`

Descripción: el comando `ls` se utiliza para listar archivos y directorios, nos permite ver además propiedades de los mismos.

Opciones:

-a: Se utiliza para ver los archivos ocultos, estos comienzan con punto «`.`»

-d: Si el argumento es un directorio, lista sólo el nombre (no el contenido); se utiliza por lo general con el flag `-l` para obtener el estado de un directorio.

-F: Marca los directorios con una diagonal «`/`», los ejecutables con un asterisco «`*`»; las vinculaciones con un arroba «`@`».

Las opciones empleadas con el comando `ls` pueden concatenarse o listarse por separado. Esto significa que los siguientes comandos son idénticos:

`ls -l -F`

`ls -lF`

```
debian:/home/dmenino# ls
clases mp3 MP3 PersonalDocs Training
debian:/home/dmenino#
debian:/home/dmenino# ls -l -F
total 20
drwxr-xr-x 6 root root 4096 2008-10-14 00:15 clases/
drwxr-xr-x 2 root root 4096 2008-10-14 00:13 mp3/
drwxr-xr-x 2 root root 4096 2008-10-14 00:13 MP3/
drwxr-xr-x 2 root root 4096 2008-10-14 00:13 PersonalDocs/
drwxr-xr-x 2 root root 4096 2008-10-14 00:14 Training/
debian:/home/dmenino#
debian:/home/dmenino# ls -lF
total 20
drwxr-xr-x 6 root root 4096 2008-10-14 00:15 clases/
drwxr-xr-x 2 root root 4096 2008-10-14 00:13 mp3/
drwxr-xr-x 2 root root 4096 2008-10-14 00:13 MP3/
drwxr-xr-x 2 root root 4096 2008-10-14 00:13 PersonalDocs/
drwxr-xr-x 2 root root 4096 2008-10-14 00:14 Training/
debian:/home/dmenino#
```



Comando: chmod

Uso: chmod [opciones] <modo> <archivo/s>

Descripción: El comando chmod se utiliza para modificar los permisos de los archivos o directorios pasados como argumento.

Opciones:

-R: Cambio recursivo (incluye subdirectorios)

Modo:

Existen dos formas de especificar permisos con este comando, por un lado esta la forma literal, que es más intuitiva y por el otro la octal, que es utilizando números octales para especificar el tipo de permisos que se le asignarán a los archivos y/o directorios.

Forma Literal:

Para especificar los permisos trabajando en forma literal se tiene que separar por el símbolo «+» (más) o «-» (menos), el segmento donde se van a aplicar los permisos del tipo de permisos que se aplicarán.

| | "r" (read) | "w" (write) | "x" (execute) |
|-------------|---------------|-----------------|-----------------|
| "u" (user) | dueño-lectura | dueño-escritura | dueño-ejecución |
| "g" (group) | grupo-lectura | grupo-escritura | grupo-ejecución |
| "o" (other) | otros-lectura | otros-escritura | otros-ejecución |

Todos estos permisos son combinables, siempre separados por un «+» (más) o «-» (menos). Se pueden colocar «u», «g» u «o», así como también «r», «w», «x» (del otro lado del separador).

Ejemplos:



Permisos de lectura, escritura y ejecución a todos los archivos.

chmod ugo+rwx *

Permisos de lectura y ejecución al grupo de todos los archivos y directorios.

chmod g+rx *

Quita los permisos de lectura, escritura y ejecución a todos los archivos

chmod o-rwx *



Formal octal:

Esta forma es menos intuitiva pero más veloz para realizar los cambios ya que el modo se especifica con un número octal de tres o más dígitos. Lo interesante que tiene esta forma de cambiar permisos es que con un solo modo se pueden cambiar todos los segmentos del permiso (dueño, grupo y ejecución) así como también el tipo de permiso asignado.

A continuación se describen los permisos más utilizados:

| Número octal | Forma literal | Permiso final |
|--------------|------------------|---------------|
| 777 | Ugo+rwx | rwXrwxrwx |
| 666 | ugo+rw,ugo-x | rw-rw-rw- |
| 755 | u+rwx,go+rx,go-w | rwXr-x-r-x |
| 444 | ugo+r,ugo-wx | r--r--r-- |
| 555 | ugo+rx,ugo-w | r-xr-xr-x |



La forma de obtener el valor numérico es la siguiente:

- se le asigna un valor a cada flag

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| r | w | x | r | w | x | r | w | x |
| 4 | 2 | 1 | 4 | 2 | 1 | 4 | 2 | 1 |

- y luego se suman para cada grupo de rwx los valores de los flags activos, por ejemplo:

rwXrwxrwx
7 7 7 = 777

rw-r--r--
6 4 4 = 644

rwXr-xr-x
7 5 5 = 755

r-----
4 0 0



Hay que tener cuidado con los permisos 666 (rw-rw-rw) o 777 (rwxrwxrwx) ya que estamos brindando acceso ilimitado a todos los usuarios del sistema.

Tabla de Permisos:

| Valor en octal | Permiso |
|----------------|---------|
| 7 | rwx |
| 6 | rw- |
| 5 | r-x |
| 4 | r-- |
| 3 | -wx |
| 2 | -w- |
| 1 | --x |

A continuación se ve un ejemplo en el cual listamos el directorio /home/dmenino/clases/Feb, vemos los permisos del archivo llamado "reporte".

Decidimos cambiar los permisos para darles escritura a todos los usuarios del sistema

```
debian:/home/dmenino/clases/Feb# ls -l
total 1072
-rw-r--r-- 1 root root 1093632 2008-10-14 00:16 reporte
debian:/home/dmenino/clases/Feb#
debian:/home/dmenino/clases/Feb# chmod ugo+rwx reporte
debian:/home/dmenino/clases/Feb#
debian:/home/dmenino/clases/Feb# ls -l
total 1072
-rwxrwxrwx 1 root root 1093632 2008-10-14 00:16 reporte
debian:/home/dmenino/clases/Feb#
```

Conociendo que es un archivo no ejecutable retiramos los permisos de ejecución sobre el archivo

```
debian:/home/dmenino/clases/Feb# chmod ugo-x reporte
debian:/home/dmenino/clases/Feb#
debian:/home/dmenino/clases/Feb# ls -l
total 1072
-rw-rw-rw- 1 root root 1093632 2008-10-14 00:16 reporte
debian:/home/dmenino/clases/Feb# _
```



Utilizando la forma octal

```
debian:/home/dmenino/clases/Feb# ls -l
total 1072
-rw-r--r-- 1 root root 1093632 2008-10-14 00:16 reporte
debian:/home/dmenino/clases/Feb#
debian:/home/dmenino/clases/Feb# chmod 666 reporte
debian:/home/dmenino/clases/Feb#
debian:/home/dmenino/clases/Feb# ls -l
total 1072
-rw-rw-rw- 1 root root 1093632 2008-10-14 00:16 reporte
debian:/home/dmenino/clases/Feb#
```



Como podrán intuir, existen diversas formas de escribir una sentencia para obtener los permisos deseados en un archivo o directorio

Los siguientes comandos se utilizan para cambiar el dueño y grupo de de los archivos o directorios:

Comando: chown

Uso: chown [opciones] <Nombre de usuario> <archivo/s>

Descripción: el comando chown se utiliza para modificar el dueño de los archivos o directorios pasados como argumento.

Opciones:

-R: Cambio recursivo (incluye subdirectorios)

Comando: chgrp

Uso: chgrp [opciones] <Nombre de grupo> <archivo/s>

Descripción: el comando chgrp se utiliza para modificar el grupo de los archivos o directorios pasados como argumento.

Opciones:

-R: Cambio recursivo (incluye subdirectorios)



Metacaracteres (o comodines)

Cuando se trabaja en la línea de comando es muy necesario contar con una herramienta que nos permita integrar varios nombres de archivos, con el uso de uno o más caracteres especiales.

Esta técnica es muy común en todos, o la mayoría de los shell existentes, pero a pesar de que en Linux los caracteres que se utilizan son muy parecidos a los utilizados en otros sistemas operativos, no siempre tienen el mismo significado y/o resultado. En algunos casos el mal uso de estos metacaracteres puede traernos consecuencias no deseadas.

Se entiende por metacaracteres a los caracteres especiales utilizados para reemplazar uno o más caracteres comunes.

El asterisco «*», metacarácter universal

El asterisco * reemplaza uno o más caracteres alfanuméricos incluyendo caracteres especiales como «,», «.», «+», «-». etc. exceptuando el punto «.» al inicio de un nombre de archivo o directorio.

Ejemplo: En un directorio existen los archivos «.agenda», «a.g.e.n.d.a», «agenda» y «agenda.»

```
# ls *  
a.g.e.n.d.a agenda agenda.  
# ls .*  
.agenda  
  
..  
a.g.e.n.d.a agenda agenda.  
#
```

Como se puede observar, al ejecutar «ls *» solamente se pudo ver tres archivos: a.g.e.n.d.a, agenda y agenda.; al ejecutar «ls .*», se visualizó, primero, el archivo «.agenda», luego el contenido del directorio «.» y por último el contenido del directorio «..».

Como se habrá notado no es necesario utilizar «*.*» para reemplazar todos los archivos de un directorio. Es necesario destacar que en Linux no existe el concepto de extensión de archivo como en DOS o Windows. En Linux, como en todos los Unix, un archivo puede contener varios puntos, al igual que otros caracteres especiales como la coma «,», los dos puntos «:», etc.

Por lo tanto, si ejecutamos «ls *.*» solamente se verán los archivos que contienen un punto (después del primer carácter) dentro de su nombre.



El símbolo de pregunta «?», metacarácter de posición

El símbolo de pregunta «?» reemplaza un único carácter y en la posición donde se encuentra el símbolo. Este metacarácter reemplaza cualquier carácter, incluyendo el punto «.» a menos que este punto sea el primer carácter del nombre del archivo o directorio.

Ejemplo: En un directorio tengo los archivos «agen», «Agenda», «Lgenda», «.agenda»

```
# ls ?gen*  
agen Agenda Lgenda
```

```
# ls .?gen*  
.agenda
```

En este ejemplo notamos que el signo de pregunta «?» reemplazo la letra «A», la «a» y la «L», pero no el «.». Para poder visualizar el archivo «.agenda» hubo que poner el punto «.» expresamente.

Es importante destacar que a diferencia del asterisco «*», el signo de pregunta es un metacarácter de posición, eso quiere decir que solo reemplazará cualquier carácter que se encuentre en esa posición y únicamente en esa posición

4.2 Montaje de dispositivos

Dado que Linux es un sistema operativo orientado a filesystem, cualquier dispositivo que utilicemos deberá ser acoplado o montado al filesystem del sistema.

Una vez que el dispositivo este montado podremos acceder al contenido y movernos en el árbol de directorios como si fuera un directorio más.

La nomenclatura utilizada para distinguir los dispositivos se basa en el tipo de dispositivo y en la posición física en la cual se encuentra conectado el mismo

Al dispositivo conectado como master en el IDE primario se le llama hda
Al dispositivo conectado como esclavo en el IDE primario se le llama hdb
Al dispositivo conectado como master en el IDE secundario se le llama hdc
Al dispositivo conectado como esclavo en el IDE secundario se le llama hdd

Para los dispositivos scsi se utiliza sda, sdb, etc.

Además, si el dispositivo es un disco rígido, las particiones se numeran. Por ejemplo, la primera partición en el disco hda es hda1 la segunda hda2, etc.



Los dispositivos están en el directorio /dev, así que para referiros a alguno de ellos utilizaremos por ejemplo /dev/hda1.

Comando: mount

Uso: mount [opciones] <dispositivo> <punto de montaje>

Descripción: el comando mount se utiliza para poder acceder al sistema de archivos y directorios del dispositivo deseado.

Opciones:

-t: Sistema de archivos (incluye subdirectorios)

-o: Opciones

Sistema de archivos

Puede ser uno de los siguientes:

- **ext2** o **ext3** Para particiones Linux que usen ese sistema de ficheros.
- **vfat** Para particiones Windows que usen fat32.
- **ntfs** Para particiones Windows que usen NTFS.
- **iso9660** para CD-roms.

Para los sistemas de archivos **ntfs** es necesario utilizar un kernel de la rama 2.6 para tener soporte de escritura. Se recomienda utilizar la opción de solo lectura **ro**.

Opciones de montaje

Las opciones más utilizadas son:

ro El dispositivo se monta con permisos de solo lectura

rw El dispositivo se monta con permisos de lectura/escritura

user Cualquier usuario puede montar y desmontar el dispositivo

auto El dispositivo se monta automáticamente al iniciar

defaults Para emplear las opciones predeterminadas: rw, suid, dev, exec, auto, nouser, async.

Cuando introducimos las opciones, hemos de ponerlas separadas por comas, por ejemplo: -o ro,auto,user



Dispositivos

Debemos seleccionar el dispositivo a montar, para ver el nombre del dispositivo existen una serie de comandos

```
# dmesg | grep hd
# dmesg | grep sd
# df -k
```

Como resultado tendremos una lista de dispositivos, identificamos el deseado y procedemos a montarlo. Por ejemplo veremos **/dev/sda1** si es una partición o **/dev/hdc** para la lectora de CD.

```
debian:/# dmesg | grep sd
Kernel command line: root=/dev/sda1 ro
SCSI device sda: 16777216 512-byte hdwr sectors (8590 MB)
sda: Write Protect is off
sda: Mode Sense: 5d 00 00 00
sda: cache data unavailable
sda: assuming drive cache: write through
SCSI device sda: 16777216 512-byte hdwr sectors (8590 MB)
sda: Write Protect is off
sda: Mode Sense: 5d 00 00 00
sda: cache data unavailable
sda: assuming drive cache: write through
sda: sda1 sda2 < sda5 >
sd 0:0:0:0: Attached scsi disk sda
Adding 273064k swap on /dev/sda5. Priority:-1 extents:1 across:273064k
EXT3 FS on sda1, internal journal
debian:/#
debian:/# df -k
Filesystem          1K-blocks      Used Available Use% Mounted on
/dev/sda1            7985248    1424364   6155248  19% /
tmpfs                 47368         0     47368   0% /dev/shm
tmpfs                 10240         84     10156   1% /dev
debian:/#
```

```
debian:/# dmesg | grep hd
ide1: BM-DMA at 0x1058-0x105f, BIOS settings: hdc:DMA, hdd:pio
SCSI device sda: 16777216 512-byte hdwr sectors (8590 MB)
SCSI device sda: 16777216 512-byte hdwr sectors (8590 MB)
hdc: VMware Virtual IDE CDROM Drive, ATAPI CD/DVD-ROM drive
hdc: ATAPI 1X CD-ROM drive, 32kB Cache, UDMA(33)
debian:/#
```



Punto de montaje

Nos referimos con esto al directorio donde queremos montar (o acoplar) el filesystem del dispositivo. Usualmente es un subdirectorio dentro de /mnt o /media, este subdirectorio debe existir antes de montar un dispositivo.

Ejemplos



```
# mount -t vfat -o defaults /dev/hda1  
/media/Windows  
  
# mount -t vfat -o rw /dev/floppy /media/diskette  
  
# mount -t iso9660 -o ro,exec /dev/hdc  
/media/cdrom  
  
# mount -t vfat -o rw /dev/sda1 /media/pendrive  
  
# mount -t ext3 -o rw /dev/hdb2 /media/linux
```

Desmontando los dispositivos

Es muy importante desmontar cualquier dispositivo previo a desconectarlo. Para ello tenemos el comando umount

Comando: umount

Uso: umount <dispositivo>

Descripción: el comando umount se utiliza para desmontar el filesystem de un dispositivo.

Opciones:

- a: Desmontar todos los dispositivos cargados en /etc/mstab
- t: Desmontar los dispositivos con el formato indicado
- f: Forzar la operación



Automatizando el montaje de dispositivos

En GNU/Linux podemos automatizar la tarea de montaje de dispositivos mediante el agregado de los mismos al archivo **/etc/fstab**

Al iniciar el sistema se montaran los dispositivos enumerados en el mencionado archivo si tienen la opción auto. Como se ve a continuación el archivo tiene una serie de columnas y opciones siendo la sintaxis la que sigue:

<dispositivo> <punto montaje> <formato> <opciones> <dump> <pass>

dump y **pass** suelen dejarse a 0, excepto para las entradas del sistema operativo para el directorio raíz / o /home, por ejemplo

```
debian:/# cat /etc/fstab
# /etc/fstab: static file system information.
#
# <file system> <mount point> <type> <options> <dump> <pass>
proc /proc proc defaults 0 0
/dev/sda1 / ext3 defaults,errors=remount-ro 0 1
/dev/sda5 none swap sw 0 0
/dev/hdc /media/cdrom0 udf,iso9660 user,noauto 0 0
/dev/fd0 /media/floppy0 auto rw,user,noauto 0 0
debian:/#
```

La ventaja de tener configurado el archivo fstab es que la orden de montar un dispositivo se abrevia. No necesitamos especificar opciones dado que estas están contenidas en el archivo. Podemos de este modo montar como se muestra a continuación:

mount <dispositivo>
mount <punto montaje>

Además si en el archivo está especificada la opción “user” les estamos dando permisos a los usuarios para poder montar el dispositivo.

Ejemplos de configuración de fstab para montar NTFS

Suponiendo queremos montar una partición tipo NTFS localizada en /dev/hdb1 en /media/windows dando permiso a los usuarios del grupo users (gid=100)

Creamos el directorio

```
mkdir /media/Windows
```



Editamos el archivo fstab

```
vi /etc/fstab
```

Agregamos la siguiente línea al final

```
/dev/hdb1 /media/windows ntfs nls=utf8,auto,ro,exec,users,umask=0222,dmask=0222  
0 0
```

Montamos el dispositivo

```
mount -a
```

4.3 Organización de archivos

Los directorios en Linux se pueden clasificar en estaticos/dinamicos y compartibles/no compartibles.

Estáticos:

Son aquellos que contienen binarios, bibliotecas, documentación y otros archivos que no cambian sin intervención del administrador. Pueden estar en dispositivos de solo lectura (read-only) y no necesitan que se hagan copias de seguridad tan a menudo como con los dinámicos.

Por ejemplo se pueden mencionar

```
/bin, /sbin, /opt, /boot, /usr/bin
```

Dinámicos:

Contienen archivos que no estáticos. Deben de encontrarse en dispositivos de lectura- escritura (read-write). Necesitan que se hagan copias de seguridad a menudo.

Pueden mencionarse

```
/var/mail, /var/spool, /var/run, /var/lock, /home
```

Compartibles:

Contienen archivos que se pueden encontrar en un ordenador y utilizarse en otro.

Por ejemplo

```
/usr/bin, /opt
```

No compartibles:

Son los que contienen archivos que no son compartibles

Como ejemplo

```
/etc, /boot, /var/run, /var/lock
```



Todos los archivos y directorios aparecen debajo del directorio raíz «/» aunque se encuentren en discos/dispositivos distintos

4.4 Cómo copiar archivos

El comando para copiar archivos es **cp**. Se necesita permiso de lectura para copiar el archivo y permiso de escritura para el directorio (o para el archivo si piensa sobrescribir en uno existente) en donde se hará la copia.

Consideraciones a tener en cuenta al copiar archivos:

Si copia un archivo y le da el nombre de un archivo que ya existe, para el cual tiene permiso de escritura, entonces se sobrescribirá en el archivo original.

Si da el nombre de un directorio como el destino del comando cp, el archivo se copia en ese directorio con su nombre original. Por ejemplo, si se ejecuta el comando cp <archivo> <directorio> el archivo se copia en el <directorio> como <directorio>/<archivo>.

Puede copiar una lista de archivos en un directorio con el comando:

cp <archivo1, archivo2, ... archivoN> <Directorio>. Si el último elemento de la lista no es un directorio, aparecerá un mensaje de error.

Se debe tener cuidado al usar metacaracteres con el comando cp, pues se podría copiar más de lo que se pretende.

Ejemplos:



Copia el archivo agenda al directorio /home/mfernandez

cp agenda /home/mfernandez

Copia el archivo agenda al directorio /home/mfernandez pidiendo confirmación

cp -i agenda /home/mfernandez

Comando: cp

Uso: cp [opciones] <arch1 arch2 ... archN> <destino>

Descripción: el comando cp se utiliza para copiar archivos en el sistema Opciones:

- r Copia de forma recursiva.
- p Mantiene los permisos originales.
- i Interroga antes de copiar.



- f Anula la opción -i y fuerza la copia de los archivos.
- v Muestra lo que esta copiando.

4.5 Como mover y renombrar archivos

En Linux, mover y renombrar archivos se lleva a cabo con el mismo comando: mv. La sintaxis y las reglas del comando mv son iguales a la del comando cp

Algo que se puede hacer con mv, pero no con cp, es mover o renombrar directorios. Al mover o renombrar un archivo, lo único que hace es modificar la entrada en el archivo de directorio (a menos que la nueva ubicación esté en otro disco físico o partición, en cuyo caso el archivo y el contenido se desplazan de sitio físico)

Ejemplo:



Renombra el archivo agenda por agenda2

```
# mv agenda agenda2
```

Mueve el archivo agenda al directorio /usr/agenda

```
# mv agenda /usr/agenda
```

Comando: mv

Uso: mv [opciones] <archivo> <destino o nuevo nombre>

Descripción: el comando mv se utiliza para mover y renombrar archivos Opciones:

- i Interroga antes de copiar.
- f Anula la opción -i.
- v Muestra lo que esta realizando el comando



4.6 Como eliminar archivos y directorios

El comando para eliminar un archivo es `rm`. Para borrar un archivo del cual no es propietario, necesita permiso tanto de lectura como de escritura. Si el archivo le pertenece, se le permite borrarlo, siempre que no haya cerrado su propio permiso hacia ese archivo. Por ejemplo, si cierra el permiso de escritura para un archivo al ejecutar `chmod 000 <archivo>`, debe abrirlo de nuevo con el comando `chmod` (para lo cual es preciso escribir `chmod 644 <archivo>`, por ejemplo) para poder borrarlo

Si por accidente escribe `rm *`, borrará del directorio actual los archivos para los cuales tiene permiso de borrar; no eliminará los subdirectorios. Para borrar subdirectorios se debe utilizar la opción `-r`

Comando: `rm`

Uso: `rm [opciones] <archivos o directorios>`

Descripción: el comando `rm` se utiliza para borrar archivos o directorios. Opciones:

- r Borra el contenido del directorio actual y los subdirectorios.
- i Interroga antes de borrar.
- f Anula la opción `-i`.
- v Muestra lo que esta realizando el comando

4.7 Visualización de un archivo

Hay muchos comandos para visualizar archivos, a continuación describiremos los mas útiles

Comando: `cat`

Uso: `cat <archivo1, archivo2, ... archivoN>`

Descripción: el comando `cat` se utiliza para mostrar por pantalla uno o varios archivos. Al mostrar varios archivos, estos son concatenados, es decir mostrado uno atrás de otro, como si fuese un solo archivo. El comando `cat` no realiza pausas entre hojas.

```
debian:/# cat texto.txt
Linea1
Linea2
Linea3
Linea4
Linea5
Linea6
Linea7
Linea8
Linea9
Linea10
Linea11
Linea12
Linea13
Linea14
Linea15
debian:/#
```



Comando: more

Uso: more <archivo1, archivo2, ... archivoN>

Descripción: este comando se utiliza para mostrar por pantalla uno o varios archivos. A diferencia de cat este comando permite ir recorriendo los archivos pantalla por pantalla presionando la tecla <Enter>. La tecla «Q» se utiliza para salir del comando more.

```
debian:/# more texto.txt
Linea1
Linea2
Linea3
Linea4
Linea5
Linea6
Linea7
Linea8
Linea9
Linea10
Linea11
Linea12
Linea13
Linea14
Linea15
debian:/#
```

Comando: less

Uso: less <archivo1, archivo2, ... archivoN>

Descripción: El comando less es un programa similar al more, pero permite mas libertad de movimiento.

```
Linea1
Linea2
Linea3
Linea4
Linea5
Linea6
Linea7
Linea8
Linea9
Linea10
Linea11
Linea12
Linea13
Linea14
Linea15
texto.txt (END)
```



Comando: zcat

Uso: zcat <archivo1, archivo2, ... archivoN>

Descripción: El comando zcat es igual al comando cat, su diferencia radica en que nos permite visualizar archivos comprimidos

```
debian:/# zcat texto.txt.gz
Linea1
Linea2
Linea3
Linea4
Linea5
Linea6
Linea7
Linea8
Linea9
Linea10
Linea11
Linea12
Linea13
Linea14
Linea15
debian:/#
```

Comando: tail

Uso: tail [opciones] <archivo>

Descripción: muestra las últimas líneas de un archivo. En el caso de no tener opciones, muestra las últimas 10 líneas del archivo.

Opciones:

- f Muestra las últimas líneas constantemente, es decir que si se agrega una línea al archivo se actualiza el contenido en pantalla.
- n n Muestra n números de líneas

```
debian:/# tail texto.txt
Linea6
Linea7
Linea8
Linea9
Linea10
Linea11
Linea12
Linea13
Linea14
Linea15
debian:/#
debian:/# tail -n 3 texto.txt
Linea13
Linea14
Linea15
debian:/# █
```



Comando: head

Uso: head [opciones] <archivo>

Descripción: muestra las primeras líneas de un archivo. En el caso de no tener opciones, muestra las primeras 10 líneas del archivo.

Opciones:

-n n Muestra n números de líneas

```
debian:/# head texto.txt
Linea1
Linea2
Linea3
Linea4
Linea5
Linea6
Linea7
Linea8
Linea9
Linea10
debian:/# head -n 3 texto.txt
Linea1
Linea2
Linea3
debian:/#
debian:/#
```

4.8 Como buscar archivos

Comando: locate

Uso: locate [archivo]

Descripción: El comando locate busca en una pequeña base de datos de indexación de archivos.

Comando: updatedb

Uso: updatedb

Descripción: El comando updatedb actualiza la base de datos donde buscara el comando locate.

```
debian:/# locate texto.txt
locate: /var/cache/locate/locatedb: No such file or directory
debian:/#
debian:/# updatedb
debian:/#
debian:/# locate texto.txt
/texto.txt
debian:/#
```



Comando: find

Uso: find [path] [opciones]

Descripción: El comando find se utiliza para buscar archivos dentro del sistema.

Opciones:

| | |
|------------------|---|
| -name [archivo] | busca [archivo] en el path indicado. |
| -iname [archivo] | Idem al anterior, pero sin hacer diferencias entre mayúsculas y minúsculas. |
| -amin n | Busca todos los archivos accedidos en los n minutos atrás. |
| -cmin n | Busca todos los archivos creados en los n minutos atrás. |
| -mmin n | Busca todos los archivos modificados en los n minutos atrás. |
| -atime | Busca todos los archivos accedidos en el momento n*24 hs. |
| -ctime n | Busca todos los archivos creados en el momento n*24hs. |
| -mtime n | Busca todos los archivos modificados en el momento n*24 hs. |
| -perm mode | Busca todos los archivos que tengan los permisos indicados en mode. |

```
debian:/# find / -name texto.txt  
/texto.txt  
debian:/#
```