ADMINISTRACIÓN EN LINUX, ASIGNACIÓN Y MODIFICACIÓN DE USUARIOS, GRUPOS Y PERMISOS.

1 OBJETIVO:

El objetivo de esta clase es profundizar en los premisos establecidos en Linux vistos la clase pasada a través del listado largo de un directorio. Hoy cambiaremos esos permisos y el usuario propietario del archivo como así también el grupo propietario del mismo.

Así también veremos como cambiar la forma en que los usuarios crean sus archivos y la generación de links. Todas estas tareas que realizaremos tienen como propósito la administración de grupos, usuarios y recursos en Linux.

2 INTRODUCCIÓN

En la clase anterior hemos aprendido a leer los permisos en Unix, estudiamos como se interpretan en general, como se diferencia entre los aplicados a un archivo y a un directorio, hemos profundizado en la comprensión de la información que nos brinda el listado completo de un directorio, y también cual es la asignación de permisos que se establece de forma predeterminada al crear un archivo o un directorio.

En la clase de hoy profundizaremos en la temática estudiando los procedimientos para cambiar los permisos ya establecidos, el usuario o el grupo propietario de un archivo o directorio, y por último como modificar la asignación de permisos predeterminada del sistema.

Cuando trabajamos con los permisos en Unix debemos tener siempre muy presente la forma en que operan los mismos, y es muy importante planificar como realizaremos su asignación. Los cambios que hagamos en un directorio pueden influir en el resto de elementos que están contenidos el, por ejemplo si quitamos el permiso de ejecución sobre un directorio a un usuario, este no podrá acceder a su contenido, por ende, tampoco podrá ejecutar ni modificar los archivos allí contenidos, por más que tenga permisos que así se lo permitan sobre los mismos.

Para realizar todas estas tareas administrativas que hemos mencionado (cambiar usuarios, grupos, permisos, etc) existen una serie de herramientas que estudiaremos a continuación:

3 HERRAMIENTAS ADMINISTRATIVAS

3.1 *CHOWN: CAMBIAR PROPIETARIO* (CHANGE OWNER)

Este comando permite cambiar al usuario propietario (*owner)* o grupo que está asociado con algún archivo o directorio. El *owner* o propietario de un archivo solo lo puede cambiar el usuario *root* mientras que el grupo además de *root*, lo puede cambiar el propio dueño siempre que pertenezca al nuevo grupo.

3.1.1 Sintaxis:

chown -[OPCIONES] USUARIO:GRUPO NOMBRE_ARCHIVO

3.1.2 Opciones:

-R opera sobre directorios recursivamente, cambiando también los archivos contenidos en el directorio.

-v (modo verbose): Muestra un mensaje por cada archivo procesado.

-c Muestra un mensaje solo por cada archivo modificado efectivamente.

3.1.3 Ejemplos:

- *chown ita p8013.jpg*

  Cambia el *owner* del archivo p8013.jpg de *root* a *ita* y mantiene al grupo propietario actual.

- *chown ita:users p8013.jpg*

  Cambia el *owner* del archivo p8013.jpg de *root* a *ita* y al grupo propietario actual a *users*.

3.2 CHGRP : CAMBIAR GRUPO (CHANGE GROUP)

Este comando realiza una tarea similar a *chown*, con la diferencia que solo cambia el grupo asociado a un archivo.

3.2.1 Sintaxis:

chgrp –[OPCIONES] NOMBRE_ARCHIVO

3.2.2 Opciones:

-R opera sobre directorios recursivamente, cambiando también los archivos contenidos en el directorio.

-v (modo verbose): Muestra un mensaje por cada archivo procesado.

-c Muestra un mensaje solo por cada archivo modificado efectivamente.

3.2.3 Ejemplo:

chgrp –vR admin flash

Cambia el grupo asociado al directorio *flash* al grupo *admin*, mostrando mensajes y cambiando también a los directorios y archivos contenidos en el mismo.

3.3 CHMOD: CAMBIAR MODO (CHANGE MODE)

El comando *chmod* se utiliza para modificar los permisos ligados con algún archivo o directorio. Para poder operar correctamente con este comando recordaremos que los permisos pueden ser "**r**" (lectura), "**w**" (escritura) y "**x**" (ejecución) y que además, para identificar al propietario usamos la **u**, al grupo propietario la **g** y a los demás usuarios la **o**. Si nos queremos referir a todos los usuarios se puede usar una **a**.

Para poder agregar un permiso se usa el modificador "+", para quitar el "-" y para asignar el "=". El uso del "=" sobrescribe todos los permisos que hayan sido declarados con anterioridad. Para especificar varios permisos consecutivos se debe separar mediante una coma ",".

Para determinar los permisos finales siempre se deben tener en cuenta los siguientes aspectos:

- Para poder realizar operaciones sobre cualquier directorio (lectura o escritura) será necesario siempre, tener otorgado además el permiso de ejecución (x).
- Para acceder a un recurso de cualquier forma (ejecución, lectura o escritura) se deben tener permisos de ejecución para todos los directorios que contienen al recurso directa e indirectamente (recordar que el sistema de archivos es jerárquico).

3.3.1 Sintaxis:

chmod -[OPCIONES] [PERMISOS] NOMBRE_ARCHIVO

3.3.2 Opciones:

-R Modo recursivo

-v Modo verbose, muestra mensajes.

-c Como -v pero solo muestra los archivos para los que hubo un cambio de permisos

3.3.3 Ejemplos:

chmod ugo +rwx carta	Agrega permisos de lectura, escritura y ejecución a todos (*user*,*group y others*) al archivo *carta*.
chmod u=rw carta	Aplica permisos de lectura y ejecución sobre el archivo carta para el usuario *owner (u)* y quita el permiso de ejecución (*x).*
chmod o-rwx carta	Quita el permiso de lectura, escritura y ejecución a los usuarios comunes (*others*) sobre el archivo carta.
chmod u=rwx,g=rw,o=r test	Asigna permisos de lectura, escritura y ejecución para el *owner*, lectura y escritura para el *grupo* y lectura para

	others.
Chmod –Rv u=rwx carpeta	Asigna permisos de lectura, escritura y ejecución sobre carpeta al propietario. Actúa en modo recursivo y muestra mensajes en pantalla.
chmod a=rw test	Aplica permisos de lectura y escritura para todos los usuarios del equipo, y quita el permiso de ejecución (*x*).

3.3.4 Método mediante el uso de números Binarios.

Existe otro método para asignar los permisos sobre un archivo, y es usando números binarios. Este método se aplica siguiendo estos pasos:

1. Se deben convertir los campos correspondientes a permisos (rwx) para las tres categorías (**u**ser, **gr**oup y **o**thers) a números binarios, reemplazando con un 1 cuando queramos activar un permiso o 0 cuando queramos desactivarlo.
2. Dividir el número obtenido en grupos de tres cifras cada uno (quedarán 3 grupos de números).
3. Convertir cada grupo de 3 números obtenidos a numeración octal.
4. Volver a reagrupar los nuevos números obtenidos, que será pasados como argumento al comando *chmod*.

3.3.5 Ejemplo:

Modo	Permisos (ugo)	Usuario	Grupo	Otros
	rwxrw-r--	rwx	rw-	r--
En binario	111110100	111	110	100
En octal		**7**	**6**	**4**

Por lo tanto para definir permisos *rwx* al propietario (*u)*, rw al grupo (g) y r a los demás (o) la sintaxis del comando será:

chmod 764 carta

Para simplificar la conversión puede ser útil la tabla siguiente.

Tabla de permisos.

Octal	**Binario**		

Numero	Numero	Permisos	Descripción
0	000	Nada	Sin permisos asignados
1	001	--x	Ejecución
2	010	-w-	Escritura
3	011	-wx	Escritura / Ejecución
4	100	r--	Lectura
5	101	r-x	Lectura / Ejecución
6	110	rw-	Lectura / Escritura
7	111	rwx	Lectura / Escritura / Ejecución

4 UMASK

Al generar un nuevo archivo o directorio el sistema le asigna un conjunto de permisos predeterminados. Por distintas circunstancias, tareas habituales de administración por ejemplo, puede surgir la necesidad de variar estos permisos predeterminados, y para ello nos valdremos del comando UMASK.

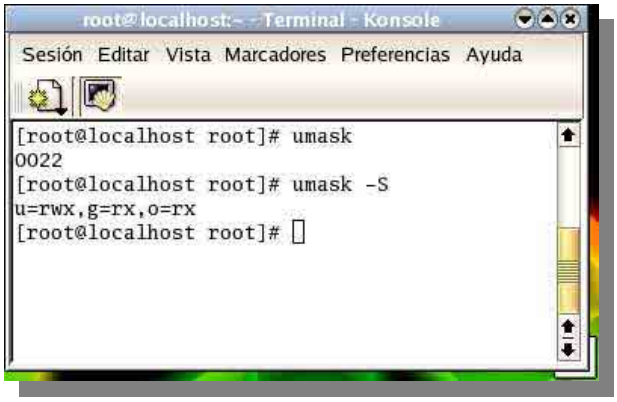
UMASK (*User Mask – Mascara de Usuario)* es un filtro que nos permitirá a partir de los permisos máximos disponibles en el sistema filtrar aquellos que no necesitemos de forma tal que la resultante sea la definición exacta que deseamos que se aplique en la generación de dichos archivo y directorio.

Umask funciona como una mascara, es decir que resta a los permisos predeterminados del sistema a aquellos que Umask determina que hay que filtrar.

4.1 EJEMPLO: UMASK APLICADO A UN DIRECTORIO:

Permisos Máximos:		777	(rwx,rwx,rwx)
Umask:	—	755	(rwx,r-x,r-x)
Permisos Resultantes:		022	(----r--r-)

Para simplificar usaremos el comando seguido del modificador *–S*. De este modo nos devolverá los permisos de la forma UGO, tal como puede verse en la figura siguiente.



Para modificar el valor de la máscara, usaremos el comando agregando los mismos modificadores que usamos con *chmod*. Por lo tanto podemos usar el signo "=" para asignar,"-"para quitar y "+" para agregar.

4.2 EJEMPLOS:

umask u=rwx,g=r,o=r	Asignará como mascara ***rwx*** para el *owner*, y ***r*** para grupo y otros.
umask o-rx	Quitará los permisos ***r*** y ***x*** a otros.
umask u=rwx,g-x,o-r	Asigna permisos ***rwx*** al ***owner***, quita ***x*** al grupo ***y*** r a otros

5 LINKS (*ENLACES)*

5.1 INODO

Los inodos son estructuras de datos que contienen a los ficheros y a toda su información dentro de los sistemas de archivos de UNIX.

Cada fichero es contenido en uno o varios inodos.

Los inodos a su vez son identificados por un número dentro del sistema de archivos en el que residen, y contienen información sobre los permisos, la propiedad, los modos de acceso, el tamaño, etc.

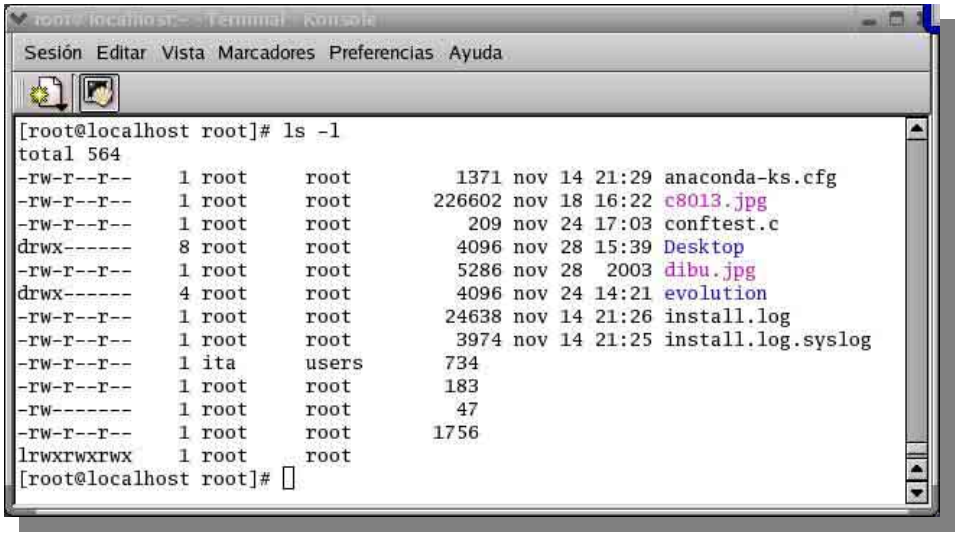
El comando ***ls*** seguido del modificador ***–i*** es permite la visualización de los inodos (***ls –i***).

5.2 INTRODUCCIÓN A LOS ENLACES

Un enlace o link es un vínculo hacia un archivo que se utiliza para referenciar al mismo desde distintas partes del sistema o incluso desde diferentes medios de almacenamiento.

Para hacer una analogía con algo que a todos nos resultará familiar, dentro de Windows es más que habitual trabajar con los llamados accesos directos, los cuales nos permiten mantener la información centraliza en un punto y luego desde diferentes lugares generar acceso a ellas.

Para visualizar el numero de links que posee algún archivo se usa el comando ***ls*** con el modificador ***–l (ls –l)***. De esta forma veremos en el segundo campo, el número de enlaces que un archivo posee.



Al momento de eliminar un link, debe tenerse en cuenta que solo se eliminará al enlace en sí, quedando intactos los archivos que referencia el mismo. Por lo tanto para eliminar definitivamente un archivo, deberán borrarse todos y cada uno de los enlaces que este posea.

5.2.1 Comando relacionado

El comando utilizado para crear enlaces o links es ***ln***.

5.2.2 Sintaxis

ln [OPCIONES] [NOMBRE1] [NOMBRE2]

5.2.3 Opciones

-s crea enlaces blandos o soft links.

-v modo verbose, muestra un mensaje por cada archivo enlazado.

-f fuerza el borrado de archivos existentes.

5.3 ENLACES DUROS O HARD LINKS

Los enlaces duros o ***Hard Links*** son los creados de forma predeterminada por el comando *ln*, usado sin ningún modificador.

Al crear un hard link se crea un nuevo archivo con un nombre diferente pero que apunta al mismo inodo. La limitación de este tipo de enlace es que ambos ficheros deben residir en el mismo sistema de archivos.

5.3.1 Ejemplo

ln test copia

Este comando crea un enlace duro al archivo *test* llamado *copia*.

5.4 ENLACES BLANDOS O SOFT LINKS

Este tipo de enlace es lo más parecido que se puede encontrar dentro de un sistema Unix de un Acceso directo de los utilizados en Windows. Las diferencias son que el link que se genera reside en un inodo diferente al del archivo original y además, que no se aplican permisos sobre el enlace, sino que se utilizan como permisos efectivos los del fichero al que este apunta.

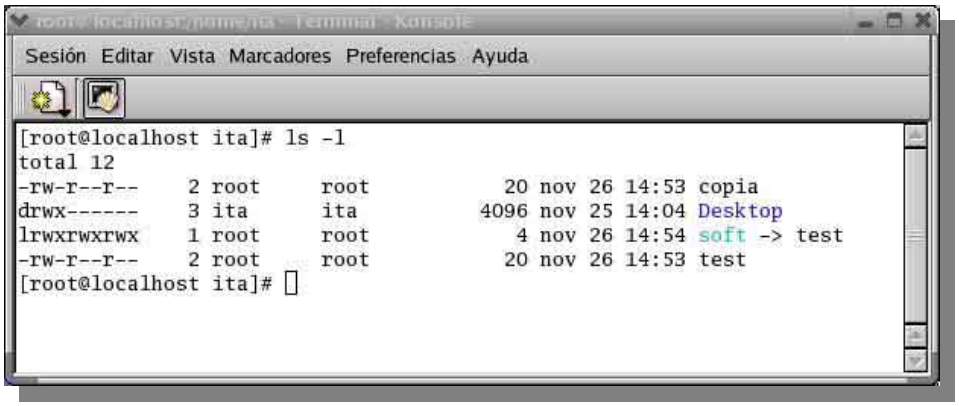
Para generar enlaces simbólicos se usa el comando *ln* seguido del modificador *–s*.

5.4.1 Ejemplo

ln –s test soft

Al ejecutar este comando se creará un soft link al archivo *test* llamado *soft*.

Como vemos en la figura siguiente el archivo *test* tiene 2 enlaces, el archivo original *test* y un enlace duro llamado *copia*. Además existe un tercer archivo llamado *soft*, que en realidad es un enlace simbólico a *test* (ver el nombre al final *soft -> test*).



Puede verse en la figura el tema de los permisos de los enlaces, mientras que *copia* posee permisos propios, *soft* en realidad usa los de *test*, como se aprecia en la primera columna (la primera l significa que *soft* es un link). Los permisos asociados son de acceso total para todos (*rwxrwxrwx*), permitiendo de este modo que se usen como efectivos los permisos que tiene aplicados el archivo *test*.

NOTAS

CUESTIONARIO CAPITULO 33

1.- El usuario Root acaba de crear un directorio en donde ha colocado 20 archivos ¿Qué comando debe utilizar para que el usuario Sergio sea el propietario del directorio y de los archivos contenidos en él?

2.- ¿Cómo cambiaría los permisos del archivo oemlogo.bmp para que el propietario acceda con escritura, lectura y ejecución, el grupo solo lectura y ejecución y otros solo lectura?

3.- ¿Qué diferencias encuentra entre los enlaces blandos y los duros?

4.- ¿Cómo haría para quitarle el permiso de ejecución al grupo y a otros sobre el archivo carta.txt?