ESCUELA POLITÉCNICA NACIONAL
ESCUELA DE FORMACIÓN DE TECNÓLOGOS
CAMPUS POLITÉCNICO "ING. JOSÉ RUBÉN ORELLANA"

INFORME PRÁCTICAS LABORATORIO

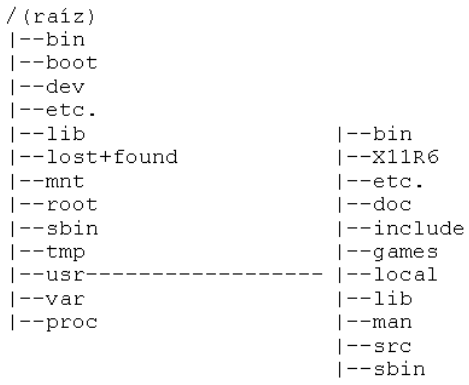
SISTEMAS OPERATIVOS 1

No. Práctica: 9

MANEJO DE PERMISO DE FICHEROS EN LINUX

El sistema de archivos de Linux está organizado en *archivos* y *directorios*. Un archivo es una colección de datos que se almacena en un medio físico y a la cual se le asigna un nombre. Los archivos, a su vez, están agrupados en conjuntos llamados directorios. Un directorio puede tener subdirectorios, formándose así una estructura jerárquica con la forma de un árbol invertido. El directorio inicial de esa jerarquía se denomina directorio *raíz* y se simboliza con una barra de división (/).

El sistema de archivos de un sistema Linux típico está formado por los siguientes directorios bajo el directorio raíz:



/bin Contiene los programas ejecutables que son parte del sistema operativo Linux. Muchos comandos de Linux como cat, cp, ls, more y tar están ubicados en este directorio.

/boot Contienen el kernel (o núcleo) de Linux y otros archivos necesarios para el administrador de inicio LILO, que realiza la carga inicial del sistema operativo cuando la computadora se enciende.

/dev Contienen todos los archivos de acceso a dispositivos. Linux trata cada dispositivo (terminales, discos, impresoras, etc.) como si fuera un archivo especial.

/etc. Contiene archivos de configuración del sistema y los programas de inicialización.

/home Contiene los directorios HOME de los usuarios. El directorio HOME el directorio inicial en el que se encuentra posicionado un usuario al ingresar al sistema, por lo que también se conoce como *directorio de logín* o *de conexión*.

/lib Contiene los archivos de biblioteca utilizados por las aplicaciones y utilidades del sistema, así también como las librerías pertenecientes a diferentes lenguajes de programación.

/lost+found Directorio para archivos recuperados por el proceso de reparación del sistema de archivos, que se ejecuta luego de una caída del sistema y asegura su integridad luego de que el equipo haya sido apagado de manera inapropiada.

/mnt Es un directorio vacío que se usa normalmente para montar dispositivos como disquetes y particiones temporales de disco.

/proc Contiene archivos con información sobre el estado de ejecución del sistema operativo y de los procesos.

/root Es el directorio HOME para el usuario root (administrador del sistema).

/sbin Contienen archivos ejecutables que son comandos que se usan normalmente para la administración del sistema.

/tmp Directorio temporal que puede usar cualquier usuario como directorio transitorio.

/usr Contiene archivos de programa, de datos y de librerías asociados con las actividades de los usuarios.

/var Contiene archivos temporales y de trabajo generados por programas del sistema. A diferencia de /tmp, los usuarios comunes no tienen permiso para utilizar los subdirectorios que contiene directamente, sino que deben hacerlo a través de aplicaciones y utilidades del sistema.

Linux proporciona cuentas para múltiples usuarios, asignando a cada cuenta un directorio hogar, un identificador numérico y un nombre (login) con los cuales obtiene acceso a la información ubicada en el directorio hogar. El esquema de seguridad de los archivos está estructurado en tres clases de usuarios.

- El dueño (u) del archivo,
- El grupo (g) al que pertenece el dueño,
- Y los otros(o) usuarios que no son el dueño o no pertenecen a su grupo.

(La letra "a" se utiliza para representar a todos los usuarios: dueño, grupo y otros). Cada archivo en GNU Linux posee un atributo para identificar el dueño y el grupo. Además, posee una serie de bits (9 en total) para definir el permiso logıa de lectura escritura y ejecución del archivo. Estos bits están organizados como se muestra en la figura.

Figura 1: Organización de los bits de permisos

Con ayuda de esta estructura se puede definir, para cada archivo, una combinación de permisos para que los usuarios del sistema tengan el acceso adecuado al archivo. El comando ls -l visualiza el estado actual de los permisos de un archivo. A continuaci´on se muestra un ejemplo: % ls -l

- -rw-r--r-- 1 visel cecalc 12601 Nov 5 14:41 VIS3.txt
- -rw-r--r-- 1 visel cecalc 17066 Nov 4 16:05 VIS4.txt
- -rw-r--r-- 1 visel cecalc 14829 Nov 6 11:06 VIS5.txt

La primera columna de información está conformada por diez caracteres. El primero es una identificación del tipo de archivo y el resto corresponde a los permisos organizados de la manera en que se muestra en la figura. Para modificar los permisos de un archivo se utiliza el comando chmod y su sintaxis es como sigue:

chmod permisos archivos

Existen dos nomenclaturas para construir el argumentos "permisos" del comando chmod. La primera de ellas consiste en generar un decimal de tres dígitos a partir de la transformación de los tres octetos que conforman los bits de permisos. Cada grupo de tres bits representa un número binario en el rango comprendido entre cero y siete. Como base de esta primera forma de construir el argumento de permisos se asume que un uno (1) implica asignar el permiso y un cero (0) significa negarlo. Si se toma un octeto cualquiera, el del dueño por ejemplo, y se le asigna permiso de lectura, escritura y se le niega el de ejecución, se tiene el número binario 110, lo cual representa al seis en decimal. El mismo procedimiento se aplica a los otros dos octetos. Así, se puede obtener el número decimal de tres dígitos que se necesita en este caso.

Lectura	Escritura	Ejecución	Valor
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Numero	Letras	Significado
0	---	Sin acceso
1	--x	Ejecución
2	-w-	Escritura
3	-wx	Escritura y ejecución
4	r--	Lectura
5	r-x	Lectura y ejecución
6	rw-	Lectura y escritura
7	rwx	Lectura/Escritura/Ejecución

Ejemplo:

Asignar todos los permisos para el dueño y el grupo, y solo lectura para el resto de los usuarios de un archivo particular. En el octeto del dueño se tiene 111 lo que es igual a 7. Para el octeto del grupo se tiene el mismo valor 111, es decir, 7. Por ´último, en el octeto de los otros tenemos 100, es decir, 4. Entonces el comando queda de la siguiente forma:

% chmod 774 archivo

chmod g+w miArchivo Asigna permiso de escritura al grupo

chmod go+rx miArchivo Asigna permiso de lectura y ejecución al grupo y otros

chmod go-r miArchivo Retira permiso de lectura al grupo y otros

chmod go= miArchivo Retira todos los permisos de grupo y otros

Ordenamiento de Archivos

sort [-t separador] [-i] archivo ...

El comando sort sirve para ordenar el contenido de un archivo. También tiene la capacidad de fusionar diferentes archivos en uno solo, manteniendo cierto orden en los registros.

Búsqueda de Cadenas de Caracteres en Archivos

Para buscar una cadena de caracteres dentro de uno o varios archivos se utiliza el comando grep

- grep cadena arch1 Muestra las líneas del archivo arch1 que contienen la palabra cadena.
- grep -i cadena arch1 Muestra las líneas del archivo arch1 que contienen la palabra cadena, pero sin distinguir entre mayúsculas y minúsculas.
- grep -n cadena arch1 Muestra las líneas del archivo arch1 que contienen la palabra cadena, pero añade el número de la línea al principio

Ejemplo: % grep slovaca secuencias.genebank

Cortar y Pegar Archivos

Existen comandos para extraer información desde archivos que se encuentren estructurados de forma particular. También en GNU Linux está presente un comando para poder unir información de manera sistematizada proveniente de archivos. El primero de los comandos es cut el cual es capaz de cortar trozos de archivos según un patrón específico.

- cut -cl1-l2,l3-l4,...,ln-lm archs Este comando extrae de los archivos archs la informaci´on de cada l´ınea comprendida entre los caracteres l1 y l2, l3 y l4 y asi sucesivamente. l1,l2,l3...lm son las posiciones de los caracteres en cada l´ınea.
- cut -d"sepf1,2,..,n archs Este comando extrae de los archivos archs las columnas 1,2,..,n las cuales se encuentran separadas por el caracter sep.

Ejemplos:

- % cut -c1-10,20-30 /etc/passwd
- % cut -d":f1,6 /etc/passwd

El comando para pegar información proveniente de archivos diferentes es paste. Para explicar cómo funciona este comando supongamos que se tienen dos archivos, cada uno de los cuales contiene una columna de datos. Supongamos que el primero de estos archivos contiene las coordenadas X de cierta ubicación espacial. Ahora supongamos que el segundo archivo contiene las coordenadas Y se desea mostrar por pantalla una columna al lado de la otra, entonces debemos ejecutar el comando paste como sigue: % paste arch1 arch2 Donde arch1 y arch2 son los archivos que contienen la información. Este comando contiene otras opciones interesantes, revíselas con el comando man.

Comparación de Archivos

El comando diff se usa para comparar dos archivos de texto. Su función es comparar línea a línea el contenido de los dos archivos y dar como salida aquellos registros que son distintos. La sintaxis general de este comando es como se muestra a continuación: % diff arch1 arch2 También puede usarse el comando sdiff que cumple la misma función que diff pero presenta la diferencia en forma horizontal: % sdiff arch1 arch2

Comparación de Directorios

Este comando permite comparar el contenido de dos directorios y genera información tabulada con el resultado de la comparación. La salida de la comparación que se realiza lista el contenido de cada uno de los directorios comparados, y luego las diferencias entre el contenido de tales subdirectorios. La sintaxis de este comando es como se muestra a continuación: % dircmp [-d] arch1 arch2 La opción demuestra el contenido donde difieren los archivos.

Estableciendo permisos desde el administrador de archivos

Bits SUID y SGID

El bit SUID es una extensión del permiso de ejecución. Se utiliza en escasas ocasiones y sirve para que cuando un usuario ejecute una aplicación, ésta se ejecute con permisos del usuario propietario en lugar de hacerlo con los del usuario que ejecuta la aplicación, es decir, es equivalente a que sea ejecutada por el propietario.

Para activar el bit SUID, se puede ejecutar el comando chmod u+s nombre_archivo o sumar 4000 al número en octal si utilizamos dicho sistema. También se puede hacer lo mismo para el grupo, es el denominado bit SGID sumando 2000 al número en octal. Activar los bits SUID ó SGID puede ocasionar problemas de seguridad sobre todo si el propietario es root.

Si aplicamos el bit SGID a una carpeta, todas las subcarpetas y archivos creados dentro de dicha carpeta tendrán como grupo propietario el grupo propietario de la carpeta en lugar del grupo primario del usuario que ha creado el archivo. Es una ventaja cuando varias personas pertenecientes a un mismo grupo, trabajan juntas con archivos almacenados en una misma carpeta. Si otorgamos permisos de lectura y escritura al grupo, los archivos podrán ser modificados por todos los miembros del grupo y cuando cualquiera de ellos cree un archivo, éste pertenecerá al grupo.

Máscaras

Cuando se crea un archivo, los permisos originales por defecto son 666 y cuando se crea una carpeta, los permisos por defecto son 777. Dichos permisos por defecto pueden modificarse con el comando umask.

Con umask podemos definir la máscara de permisos, cuyo valor original es 000. El permiso por defecto será el resultado de restar del permiso original, el valor de la máscara. Si deseamos que los archivos se creen con permisos 644 (lo más habitual), pondremos máscara 022 ya que 666-022=644. En el caso de las carpetas, el permiso efectivo será 755 ya que 777-022=755. Si analizamos el valor de la máscara en binario, cada bit a '1' desactiva un permiso y cada bit a '0' lo activa, es decir, si tiene un valor 022 (000 010 010) cuando creemos una carpeta, tendrá permisos rwxr-xr-x y cuando creemos un archivo tendrá permisos rw-r--r-- ya que el permiso de ejecución para archivos hay que fijarle con chmod al tener los archivos el permiso original 666.

Cada usuario tiene su máscara. Se puede fijar la máscara por defecto para todos los usuarios en el archivo /etc/profile o para cada usuario en el archivo /home/usuario/.bashrc

// Ejemplo de uso de umask

visel@ubuntu:/tmp$ umask

La modificación con umask de la máscara por defecto no afecta a los archivos y carpetas existentes sino solo a los nuevos que cree ese usuario a partir de ese momento.

Grupos privados de usuario

Para hacer más flexible el esquema de permisos Unix, se recomienda utilizar grupos privados de usuario. Consiste en crear un nuevo grupo con el mismo nombre del usuario, cada vez que se crea un nuevo usuario y hacer que el grupo principal del nuevo usuario sea el nuevo grupo.

Ejemplo, si creamos un usuario visel, crearemos también un grupo llamado visel y haremos que el grupo primario del usuario visel sea el grupo visel.

En el siguiente ejemplo observamos que el UID del usuario visel es 1002 y que su grupo principal es el 1003 que corresponde al GID del grupo visel. También vemos que si creamos un nuevo archivo, pertenecerá al usuario visel y al grupo visel.

// Ejemplo: Usuario visel y grupo visel

visel@ubuntu:/tmp$ more /etc/passwd |grep visel

visel:x:1002:1003::/home/visel:

visel@ubuntu:/tmp$ more /etc/group |grep visel

visel:x:1003:

visel@ubuntu:/tmp$ ls > archivo.txt

visel@ubuntu:/tmp$ ls -l

-rw-rw-r-- 1 visel visel 12 Feb 12 20:17 archivo.txt

visel@ubuntu:/tmp$

Aunque parezca inservible, la creación de un grupo personal para cada usuario, permitirá crear otros grupos mediante los cuales, diferentes personas puedan trabajar de forma colaborativa sobre los archivos dentro de una carpeta concreta. Veámoslo mejor con un ejemplo:

Supongamos que creamos una carpeta llamada 'exámenes' que pertenezca al grupo profesores. Si establecemos el bit SGID en dicha carpeta con el comando 'chmod g+s exámenes', todos los archivos que se creen dentro de dicha carpeta tendrán como grupo propietario el grupo profesores. Si todos los usuarios utilizan máscara 002, los permisos de los archivos serán 664 con lo cual, cualquier integrante del grupo profesores podrá visualizar y modificar los archivos.

El problema de usar la máscara 002 es que cualquiera que pertenezca al grupo principal de un usuario, tendría acceso de escritura sobre sus archivos, pero esto no sucederá nunca ya que cada usuario tiene su propio grupo principal y nadie más pertenece a él.

Cambiar usuario propietario y grupo propietario

Para poder cambiar el usuario propietario y el grupo propietario de un archivo o carpeta se utiliza el comando chown (change owner). Para ello hay que disponer de permisos de escritura sobre el archivo o carpeta. La sintaxis del comando es:

chown nuevo_usuario[.nuevo_grupo] nombre_archivo

En el siguiente ejemplo vemos una secuencia de comandos en la que inicialmente comprobamos que el archivo 'examen.txt' pertenece al usuario visel y al grupo profesores. Posteriormente hacemos que pertenezca al usuario luis y luego hacemos que pertenezca al usuario pedro y al grupo alumnos:

// Cambiar propietario y grupo propietario

root@ubuntu:/tmp# ls -l

total 4

-rw-rw-r-- 1 visel profesores 11 2016-06-23 20:15 examen.txt

root@ubuntu:/tmp# chown luis examen.txt

root@ubuntu:/tmp# ls -l

total 4

-rw-rw-r-- 1 luis profesores 11 2016-06-23 20:15 examen.txt

root@ubuntu:/tmp# chown pedro.alumnos examen.txt

root@ubuntu:/tmp# ls -l

total 4

-rw-rw-r-- 1 pedro alumnos 11 2016-06-23 20:15 examen.txt

root@ubuntu:/tmp#

ACTIVIDADES PROPUESTAS

// Dar permiso de escritura al usuario propietario sobre el archivo 'examen.txt'

// Quitar permiso de escritura al resto de usuarios sobre el archivo 'examen.txt'

// Dar permiso de ejecución al grupo propietario sobre el archivo '/usr/bin/games/tetris'

// Dar permiso de lectura al grupo propietario sobre el archivo 'examen.txt'

// Poner varios permisos juntos separados por comas

// Poner varios usuarios juntos

// Dar todos los permisos al usuario y ninguno ni al grupo ni al resto

// Dar al usuario y al grupo permisos de lectura y ejecución y ninguno al resto

// Dar todos los permisos al usuario y lectura y ejecución al grupo y al resto

// Dar todos los permisos al usuario y de lectura al resto, sobre todos los archivos

// Cambiar permisos a todos los archivos incluyendo subcarpetas

Actividad

Situado en el home del usuario del sistema con el cual se ingresó a Ubuntu, realice las siguientes actividades.

1-Cree un nuevo directorio llamado **PRACTICA2**

a) Al interior de este directorio cree dos nuevos llamados:

PROCESO1 **PROCESO2**

b) Dentro del directorio PROCESO1 cree dos archivos llamados:

PRACX1.TXT **PRACX2.TXT**

c) Copie los dos archivos al interior de PROCESO2 pero renómbrelos con el nombre original seguido por la palabra "nuevo".

d) Renombre los archivos del directorio **PROCESO1** con el nombre original por la palabra "antiguo".

e) Retire todos los permisos del usuario sobre el directorio **PROCESO1**.

-Trate de ingresar al directorio. Qué ocurre?.

-Vuelva a asignarle al usuario, únicamente el permiso de lectura del directorio. Qué ocurre?.

f) Cambie los permisos de los archivos al interior del directorio **PROCESO2**, solo permitiendo al grupo y a otros leer y ejecutar pero no escribirlos.

g) Cambie el permiso del directorio PROCESO2 para que todos puedan escribir. Liste el resultado , ocurrió algo en la forma de presentar el directorio?.

h) Nuevamente cambie los permisos de usuario, grupo y otros para el PROCESO2 de manera recursiva. Emplee la notación de texto equivalente para 755. Ocurrió algún cambio?.

2- Retorne al directorio HOME del usuario y cambio los permisos de manera recursiva del directorio PRACTICA2, impidiendo que el usuario pueda escribir dentro del mismo.

a) Cree un nuevo directorio llamado PROCESO3 . Que ocurrió?.
b) Es necesario cambiar algún permiso para crear este nuevo directorio?.
c) Cambie el permiso del directorio PROCESO3 a 555.
d) Copie el contenido del directorio PROCESO2 a PROCESO3. Qué ocurre?.
e) Es necesario modificar algún permiso para completar el punto d?