Usuarios, grupos y permisos

Arquitectura y Sistemas Operativos.
Tecnicatura Superior en Programación.
UTN-FRA

Autores: *Prof. Martín Isusi Seff*

Revisores: *Prof. Marcos Pablo Russo*

Versión: 1

Los sistemas operativos basados en Linux y Unix trabajan de una manera distinta al resto de los sistemas operativos en términos de "multitarea", aunque presentan, de todas maneras, algunas diferencias. La principal diferencia radica en que, mientras que la mayoría de sistemas operativos solo permiten que un solo usuario esté trabajando al mismo tiempo, Linux fue diseñado para permitir múltiples usuarios al mismo tiempo. Para que estos usuarios no afecten las sesiones del resto de los usuarios, se tuvo que desarrollar un complejo modelo de permisos.

Permisos de lectura, escritura y ejecución

Los permisos son los "derechos" que tiene un usuario o grupo sobre un archivo o directorio.
Los permisos básicos son *lectura, escritura y ejecución.*

- *Permiso de lectura (read).* Este permiso hace que el contenido de un archivo sea visible, o, en el caso de los directorios, el que contenido del mismo sea visible.
- *Permiso de escritura (write)*. Este permiso hace que el contenido de un archivo pueda ser modificado, mientras que para directorios permite realizar acciones sobre los archivos que contiene (borrarlos, agregar nuevos archivos, etc.)
- *Permiso de ejecución (execute)*. En archivos, este permiso hará posible la ejecución del mismo. Para esto, el archivo deberá ser un *script* o un programa.

Listar los permisos de un archivo o directorio

Para listar los permisos de un archivo o directorio, basta con ejecutar el comando ***ls***, junto con la opción ***-l***. El resultado de la ejecución de dicho comando mostrará el contenido de un directorio de la siguiente manera:

```
-rw-r--r-- 1 root root 1031 Nov 18 09:22 /etc/passwd
```

Los primeros diez caracteres del ejemplo representan los permisos. El primer carácter (que en este caso es un -) representa el tipo de archivo que está listando. Los tipos de archivo son:

- **d** representa directorios
- **s** representa un archivo especial
- **-** representa un archivo regular

Los siguientes nueve caracteres representan puntualmente qué permisos tienen el propietario, los permisos que tienen aquellos usuarios que pertenezcan al mismo grupo que el propietario, y los permisos que tienen aquellos usuarios que no pertenecen al mismo grupo que el propietario. El siguiente cuadro muestra cómo se descomponen los diez caracteres, utilizando el ejemplo anterior:

-	rw-	r--	r--
Tipo de archivo	Permisos del dueño	Permisos del grupo del dueño	Permisos del resto de los usuarios

Podemos observar que los permisos del propietario son **rw-**. Esto quiere decir que el usuario

tiene permisos de lectura (***r***), y permisos de escritura (***w***). Los permisos de ejecución, si los tuviera, estarían representados con una ***x*** luego del permiso de escritura. En este caso, como no posee permisos de ejecución, solo vemos un **-**.

Siguiendo con la misma lógica, podemos decir que tanto los usuarios del grupo del propietario, como el resto de los usuarios, solo poseen permisos de lectura.

Administrando cuentas de usuario

El comando para crear un nuevo usuario estándar es ***useradd***. La sintaxis es la siguiente:

```
useradd <nombre>
```

Las opciones que podemos utilizar son las siguientes:

Opción	Descripción
-d <home_dir>	Nos permite especificar cuál será el directorio principal del usuario que estamos creando.
-e <fecha>	Permite especificar una fecha en la que expirará la cuenta del nuevo usuario.
-s	Especifica el intérprete de línea de comandos que usará por defecto el nuevo usuario. Por ejemplo: ***useradd nuevousuario -s /bin/bash***

Una vez creado el nuevo usuario, es necesario configurar un password para el mismo. Esto se hace utilizando el comando ***passwd***. La sintaxis es la siguiente:

```
passwd <usuario>
```

Para poder ejecutar el comando, es necesario tener permisos de superusuario.

Otra manera de crear usuarios es utilizar el comando ***adduser***. El mismo no se encuentra siempre instalado, por lo que hay que instalarlo como cualquier otro paquete. Este comando permitirá crear usuarios de una manera más simple ya que automáticamente creará el directorio principal, asignará un intérprete de línea de comandos y configurará el password.

Para remover una cuenta de usuario, el comando a utilizar es ***userdel***. La sintaxis del mismo es:

```
userdel <usuario>
```

Se debe tener en cuenta que ejecutar el comando como se muestra en el ejemplo anterior solo eliminará la cuenta, pero los archivos y el directorio principal no serán borrados. Para poder borrar un usuario completamente, el comando deberá ser ejecutado utilizando la opción ***-r***.

```
userdel -r <usuario>
```

Entendiendo *sudo*

El usuario *root*, es el usuario que tiene los permisos para hacer cualquier cosa en el sistema. Es por eso que, por motivos de seguridad, se suele utilizar ***sudo***. Este comando permite a

usuarios y grupos, tener acceso a comandos que no podría acceder normalmente. Esto quiere decir que aquellos usuarios ***sudo*** podrán ejecutar comandos como si fuesen *root* sin realmente estar logueados como usuarios *root*.

El paquete ***sudo*** no está presente en todas las distribuciones de GNU/Linux, por lo tanto, en muchos casos será necesario instalarlo.

Para que un usuario determinado sea pueda utilizar ***sudo***, será necesario que el mismo esté presente en el archivo ***sudoers***. Este archivo se encuentra ubicado dentro del directorio ***/etc.***

Es importante tener en cuenta que, para agregar un usuario a la lista de ***sudoers*** (agregarlo al archivo), es necesario tener permisos *root*.

Trabajando con grupos

Los grupos en Linux son una manera de organizar los usuarios, principalmente como una medida de seguridad. El control de los grupos se administra a través del archivo ***/etc/group***, que contiene la lista de grupos y sus miembros. Cada usuario tiene un grupo primario por defecto. Cuando un usuario inicia sesión, se asocia al mismo con el grupo primario al que pertenece. Esto quiere decir que cada vez que el usuario cree un archivo, o ejecute un programa, esto quedará asociado al grupo primario. Para cambiar el grupo con el que un usuario está asociado en un momento determinado, se utiliza el comando ***chgrp***.

Cambiando permisos de archivos y directorios

Suponiendo que ejecutamos el comando ***ls -s***, si la salida es la siguiente:

```
-rw-r--r-- 1 root root 1031 Nov 18 09:22 /etc/passwd
```

Analicemos la información. Los primeros diez caracteres sabemos que representan los permisos que distintos grupos, y el propietario, tienen sobre el archivo o directorio. Luego, el número que se muestra representa, en caso de archivos, la cantidad de inodos que utiliza, y en en directorios, representa la cantidad de archivos y subdirectorios que contiene. Luego tendremos el propietario, seguido por el grupo del propietario (al momento que creó el archivo o directorio). En este caso, tanto el propietario, como el grupo, son *root*.

Comando *chmod*

El comando ***chmod*** nos permitirá cambiar permisos en archivos o directorios. Existen dos maneras de de especificar nuevos permisos al utilizar el comando ***chmod***, con letras o números (en base octal).

Para utilizar el comando ***chmod*** con letras, hay que tener en cuenta lo siguiente: El signo **+** agregará permisos y el signo **–** quitará permisos. Por otro lado, las letras que podemos utilizar y su significado son las siguientes:

- ***r*** representa el permiso de lectura.
- ***w*** representa el permiso de escritura.

- ***x*** representa el permiso de ejecución.
- ***X*** representa el permiso de ejecución (sólo aplica a directorios).

Por ejemplo, el comando

```
chmod +r <archivo>
```

Nos permitirá agrega permisos de escritura tanto al propietario como a su grupo y al resto de los grupos.

En el caso de utilizar el modo *octal* para definir los permisos, se deberá "calcular" el valor de los permisos tanto para el propietario, como para su grupo y para el resto de los grupos. Este cálculo se basa en la siguiente tabla:

Valor octal	Lectura (r)	Escritura (w)	Ejecución (x)
7	r	w	x
6	r	w	-
5	r	-	x
4	r	-	-
3	-	w	x
2	-	w	-
1	-	-	x
0	-	-	-

Esto se puede traducir en el siguiente ejemplo: Supóngase que se desea agregar los siguientes permisos sobre un archivo llamado ***archivo.txt:*** permisos de lectura, escritura y ejecución (rwx) al propietario, lectura y escritura (rw-) al grupo del propietario y solo lectura al resto de los usuarios (r--). El comando para realizar esto sería:

```
chmod 764 archivo.txt
```

Permisos adicionales en archivos

Además de los permisos comunes, lectura, escritura y ejecución, hay algunos permisos adicionales que también pueden resultar útiles.

Estos permisos adicionales son el ***sticky bit (t)*** y ***setuid bit (s)***. Estos dos permisos describen el comportamiento de los archivos y ejecutables en situaciones "multiusuario".

Cuando se define en un archivo o directorio, el modo ***sticky bit***, significa que solo el propietario (o *root*) pueden eliminar el archivo, independientemente de los permisos que tengan el resto de los usuarios. Para configurar el permiso ***sticky bit*** sobre un archivo ejecutamos los siguiente:

```
chmod +t <archivo>
```

El permiso ***setuid*** se puede explicar mediante el siguiente ejemplo: Supóngase el usuario *usuario1* que es propietario de *programa.sh.* El usuario *usuario1*, pertenece al grupo *grupo1*.

Suponiendo que el resto de los usuarios pertenecientes al *grupo1* tienen permisos de ejecución sobre el archivo *archivo1.sh*, éstos podrán ejecutar el archivo como si fueran el *usuario1*. Para aquellos que estén acostumbrados al sistema operativo Windows, este comportamiento es similar a la opción "ejecutar como", que permite ejecutar un programa como si fuera un usuario distinto.

Cambiando el propietario de un archivo

Por defecto, el propietario de un archivo es aquel que crea el mismo, y por defecto, el grupo asociado es grupo en que esté registrado el usuario al momento de crear el archivo. Para cambiar el propietario de un archivo, se utiliza el comando ***chown***. Por ejemplo, se tiene el archivo *lista.txt* con los siguientes detalles:

```
-rw-r--r-- 1 administrador1 grupoadmin 1031 Nov 18 09:22 /home/admin/lista.txt
```

Si se quisiera cambiar el propietario de tal archivo (que actualmente es *administrador1*, y está bajo el grupo *grupoadmin*) por *administrador2*, el comando a ejecutar sería el siguiente:

```
chown administrador2:grupoadmin /home/admin/lista.txt
```