Tema 3 - Gestión básica de grupos, usuarios y permisos

ÍNDICE

0. Introducción

Este tema trata sobre la gestión de usuarios y grupos de usuarios en sistemas Ubuntu. Pero la teoría y filosofía de trabajo es aplicable a cualquier sistema Linux. También describe quién es el usuario *root*.

En los sistemas Linux, todo lo que un usuario puede o no puede hacer con un fichero está determinado por sus permisos y los de sus grupos. Si se dispone de los permisos necesarios, o si se pertenece al grupo con permisos necesarios, se podrá acceder a un fichero o directorio.

1. El usuario root y comandos relacionados

El usuario ***root*** es un super-usuario que puede ejecutar cualquier operación en el sistema: **es el usuario con más privilegios**. Esto le permite poder mantener y administrar el sistema.

Normalmente, sólo el dueño de un fichero/directorio puede modificar los permisos del mismo. Y **normalmente un usuario sólo tiene total libertad dentro de su *home* o directorio de trabajo.**

Sin embargo, **el usuario *root* puede modificar los permisos de cualquier fichero del sistema, incluso aunque no sea el propietario.**

De igual forma que podemos acceder al sistema con nuestro usuario, también podemos acceder al sistema como *root*, para tener todos los privilegios.

Estudiaremos más adelante los comandos ***sudo***, **su** y **passwd** pero adelantaremos que si el usuario root no está activo entonces debemos ejecutar el comando `sudo -i` (Esto nos permitirá acceder como root, y una vez hecho ejecutar el comando `passwd` para cambiar su contraseña (password): sudo passwd root)

```
sudo -i

sudo passwd root
```

- Si hemos accedido al sistema como un usuario (cualquiera) pero queremos ejecutar comandos como superusuario podemos usar el comando **sudo**.
- Y si hemos accedido con otro usuario y queremos convertirnos en usuario *root*, usaremos el comado **su**.

El comando `su`

su es el acrónimo de «substitute user» (usuario sustituto). **Este comando nos permite cambiar de usuario sin cerrar la sesión del usuario actual.** Este comando se utiliza cuando tenemos que ejecutar comandos con la cuenta de otro usuario.

Hay dos formas de cambiar a la cuenta del usuario root con el comando `su`:

- Sin heredar las variables de entorno
- Heredando las variables de entorno del usuario que ejecuta el comando

Cambiar a la cuenta del usuario root sin heredar las variables de entorno del usuario que ejecuta el comando:

```
su -
```

Cambiamos al usuario root heredando las variables de entorno del usuario que ejecuta el comando:

```
su
```

También se usa `su` **para convertirnos en otro usuario existente en el sistema:**

```
su - lola
```

Para volver a la sesión de nuestro usuario basta con introducir el comando `exit`.

El comando `sudo`

Permite ejecutar comandos con permisos de *root* o super usuario. En Ubuntu, para que un usuario pueda ejecutar comandos con *sudo*, **debe pertenecer al grupo `sudo`.**

Sólo un usuario con privilegios puede añadir a otro usuario al grupo de **"sudoers"**. Para gestionar los usuarios que pueden usar *sudo*, se usa el comando `visudo`.

Este comando abre un editor (vi) **para editar el fichero */etc/sudoers*.** En el caso de Ubuntu se abre nano.

```
su -
# Añadir un usuario al grupo "sudoers"
usermod -aG sudo lola
```

El comando `passwd`

Podemos cambiar la contraseña de root utilizando el comando passwd.

Sintaxis: **passwd [opciones] [USUARIO]**

- -d: borra la contraseña
- -e: hace expirar la contraseña para que en el siguiente Loguin se solicite cambiarla

Una vez que estamos con el usuario root, podríamos cambiar su password utilizando:

```
passwd root
```

Gestion de usuarios "sudoers": `visudo`

Como root, ejecutamos:

```
visudo
```

Al usar *visudo*, se debe buscar una línea similar a la siguiente:

```
%sudo        ALL=(ALL)       ALL
```

Si la línea estuviese comentada (comienza por #), habrá que descomentarla.

Para que los cambios tengan efecto, se debe salir de la sesión y volver a entrar, o también se puede abrir una nueva *shell* con el usuario.

Otra forma alternativa de añadir un usuario a /etc/sudoers

En lugar de añadir el usuario al grupo *sudo* también se puede añadir el usuario usando visudo, e insertando una línea similar a esta:

```
elenagg        ALL=(ALL)       ALL
```

Algunos comandos importantes sobre el "usaurio"

Antes de nada, como usuario de un sistema Linux: ¿quién soy yo? ¿a qué grupos pertenezco? Ambas preguntas se responden con estos comandos:

```
# ¿Qué usuario soy yo?
whoami

# ¿A qué grupos pertenezco?
groups

# Obteniendo toda la información a la vez
id

# UID
id -u

# Nombre del usuario (loguin)
id -un

# GID
id -g

# Nombre del grupo
id -gn

man id
```

2. Gestión de "grupos" en Linux

Ficheros de Grupos: /etc/group y /etc/gshadow

Para permitir una administración flexible de los permisos de los usuarios, **Linux permite organizar los usuarios en grupos y de esta forma asignar los permisos de ficheros y directorios a un grupo.**

La organización de usuarios en grupos de usuarios permite otorgar diferentes permisos a cada grupo, y de esta forma gestionar “en bloque” qué grupos tienen permiso y para qué.

Un grupo es un conjunto de usuarios que comparten los mismos permisos.

Hay dos tipos de grupos:

- Primario
- Secundarios

Todos los usuarios deben pertenecer a un grupo principal o primario (obligatoriamente), pero pueden pertenecer a otros grupos, esos son considerados secundarios. Los grupos están formados sólo por usuarios, es decir, no se admite que un grupo contenga otros grupos.

Cada grupo de usuarios en Linux, es identificado con un numero diferente. E**sto es conocido como identificador de grupo o GID o Group IDentifier.**.

En Linux, la información relativa a los grupos está en los ficheros `/etc/group` y `/etc/gshadow`.

/etc/group

Contiene las caracterísitcas de todos los grupos del sistema. Cada grupo está definido en una línea del fichero */etc/group*. Cada línea consta de 4 campos (separados por `:`)

```
cat /etc/group | grep elenagg

# El grupo elenagg aparece en varias líneas:

adm:x:4:syslog,elenagg
cdrom:x:24:elenagg
sudo:x:27:elenagg
dip:x:30:elenagg
plugdev:x:46:elenagg
lpadmin:x:120:elenagg
lxd:x:131:elenagg
elenagg:x:1000:
sambashare:x:132:elenagg
vboxsf:x:998:elenagg

 1     2  3   4
```

1. **Nombre del grupo**.

2. **Contraseña cifrada** (Si está vacío no tiene contrasña. Una **x** indica que se gestiona mediante una contraseña guardada en el fichero /etc/gpaswd).
3. **GID**.
4. Lista de **miembros** (separados por `,`).

En el ejemplo anterior se puede ver que un usuario como **elenagg** pertenece a varios grupos.

/etc/gshadow

Almacena los passwords encriptadas de los grupos de ususario aunque generalmente no se suele usar.

El primer campo es el nombre del grupo y el último campo identifica los miembros del grupo separados por comas.

```
sudo cat /etc/gshadow

# Se muestra parte del fichero:

root:*::
daemon:*::
bin:*::
sys:*::
adm:*::syslog,elenagg
cdrom:*::elenagg
sudo:*::elenagg,LRSO
vboxsf:*::elenagg
```

Añadir grupos `groupadd`

Comando para añadir un grupo.

```
sudo groupadd developers

# Lo verificamos
cat /etc/group | grep developers

man groupadd
```

Modificar grupos `groupmod`

Comando para modificar un grupo. Podemos modificar el GID y el nombre.

```
# Se puede modificar el GID
sudo groupmod -g 1003 developers
```

```
# Se puede modificar el nombre
sudo groupmod -n nuevo_nombre_gr antiguo_nombre_gr
sudo groupmod -n desarrolladores developers

man groupmod
```

Eliminar grupos `groupdel`

Comando para eliminar un grupo. Un grupo no se podrá eliminar si tiene algún miembro.

```
sudo groupdel developers

man groupdel
```

Añadir password a un grupo `gpasswd`

No es común añadir passwd a un grupo porque tendría que ser compartida por todos los usuarios del grupo. No obstante la sitanxis sería:

```
sudo gpasswd developers

# Se puede comprobar que ahora el grupo tiene passwd:

sudo cat /etc/gshadow | grep developers
```

3. Gestión de "usuarios" en Linux

El concepto de usuario en Linux permite separar entornos de ejecución para diferentes propósitos permitiendo trabajar dos usuarios de manera simultánea en el mismo sistema, teniendo cada uno un usuario diferente y un directorio personal(home) diferente.

En los sistemas o plataformas multiusuario es muy común que muchos servicios internos del sistema tengan su propio usuario para restringir el acceso de ese servicio como mecanismo de seguridad.

Ficheros de Usuarios: /etc/passwd y /etc/shadow

la configuración de los usuarios en linux está en los ficheros `/etc/passwd y /etc/shadow`

/etc/passwd

Cuando se crea un usuario en el sistema, se alamcena la siguiente información en el fichero `/etc/passwd`. Esta información son 7 campos, separados por caracteres (`:`).

```
username:x:UID:GID:comment:home_directory:login_shell
    1    2  3   4     5        6               7

root:x:0:0:root:/root:/bin/bash
elenagg:x:1000:1000:Elena GG,,,:/home/elenagg:/bin/bash
```

Los campos son:

1. **Id. de usuario**: login del usuario. Se almacena en las variables $USER o $LOGNAME de la *shell*.
2. **Contraseña** cifrada (o una x que indica que se usa `/etc/shadow`)
3. **UID** (código numérico del usuario)
4. **GID** (código numérico del grupo primario) - Aunque un usuario puede pertenecer a más de un grupo.
5. **Comentarios**: por ejemplo el nombre completo, o el nombre de la empresa, etc.
6. **Directorio home** (ruta absoluta): normalmente */home/$USER*.
7. ***Shell*** del usuario cuando hace *login*: por ejemplo, */bin/bash*.

Normalmente, el fichero */etc/passwd* puede ser leido por todos los usuarios, pero sólo el usuario *root* puede modificar su contenido.

Si el usuario tiene una **x**, quiere decir que su contraseña cifrada se encuentra almacenada en el fichero */etc/shadow*. Este fichero sólo puede ser leído y modificado por *root*.

/etc/shadow

Este fichero contiene información sobre contraseñas de los usuarios que hay en el fichero /etc/passwd, y las almacena de manera cifrada. Para visualizar el contenido de este archivo tendremos que tener más privilegios que un usuario normal, por esto ejecutaremos:

```
sudo cat /etc/shadow
root:$y$j9T$EaaXbnqASlnnHzBix8duq/$uLZly31tANV.oG.83fGeahjQ4FJXSUMZYQ1AwbyF9wC:194
06:0:99999:7:::
bin:*:19213:0:99999:7:::
sys:*:19213:0:99999:7:::
sync:*:19213:0:99999:7:::
egg:$y$j9T$nSmR7xIMNGcuLOqE7uWd7.$o5Lq7ZhCQIDe0PEaFBS7QN/1ayLdTc8bFAzpZD8w3j.:1940
1:0:99999:7:::
```

Los campos son:

1. **Id. de usuario**: nombre del usuario
2. **Contraseña**: Password encriptado.
3. **lastmod**: Tiempo transcurrido desde el último cambio de clave.
4. **Min**: Número mínimo de días hasta que se puede volver a cambiar la contraseña.
5. **Max**: Número máximo de días hasta que el sistema obliga a cambiar la contraseña del usuario.
6. **Aviso**: Número de días previos al Max en los que el usuario es avisado de su obligado cambio de contraseña
7. **Inactividad**: Número de días entre el vencimiento de la contraseña y el bloqueo de la cuenta.
8. **Expiración**: Fecha en la que la cuenta se deshabilita. Si se deja en blanco, la cuenta nunca expira.
9. **Reservado**: Campo reservado para futuros usos.

Sobre el campo "contraseña":

- "*" se usa cuando la cuenta de usuario nunca ha tenido un password.
- "!" significa que la cuenta ha sido deshabilitada para loguearse mediante password.

Añadir usuarios `adduser`

Al crear un usuario en Linux, este recibe un UID (Identificador de Usuario Único) **y a su vez **se crea un grupo para ese usuario con su GID** (Identificador Único de Grupo) para ese grupo del nuevo usuario. El grupo que se crea es el grupo primario del usuario.

Al añadir un usuario **se crea su directorio personal en ***/home/***. Por ejemplo, para la usuaria *peppa*, se creará el directorio */home/peppa*.

Para cada nuevo usuario se copiarán dentro de su espacio de trabajo los ficheros que se encuentran en el directorio */etc/skel*. Es el directorio que se encarga de contener la "plantilla" del home de cada nuevo usuario que creamos. Podemos observar su contenido con:

```
ls -la /etc/skel
```

`adduser` Sintaxis básica:

sudo adduser [--home DIR] [--shell INTÉRPRETE] [--no-create-home] [--uid ID][--ingroup GRUPO | --gid ID] USUARIO

El fichero de configuración **/etc/adduser.conf** establece las directivas que tiene que seguir el comando adduser cada vez que se crea un nuevo usuario. Es decir, **tiene los valores de configuración por defecto cuanto creamos un usuario.**

```
# Crea el usuario y el grupo primario de dicho usuario, con la shell por defecto y con el directorio del usuario por defecto en /home/<usuario>/
sudo adduser peppa

# Crea un usuario y lo añade al grupo developers (tiene que existir)
# "ingroup" es el grupo por defecto que se crea cuando creas un usuario. Si no se especifica con esta opción el grupo que se crea toma el mismo nombre que el usuario
sudo adduser --ingroup developers peppa

# Para ver los grupos de un usuario:
groups peppa


# Crea un usuario indicando cual sera su directorio de trabajo
sudo adduser --home /home/usu_peppa peppa

# Añade un usuario existente a un grupo ya existente
# Por ejemplo, para un grupo llamado developers se haría:
sudo adduser peppa developers

# Para no crear un directorio de usuario en /home (--no-create-home)
sudo adduser --no-create-home peppa

# Para crear un usuario o cuenta de sistema (--system)
# Por omisión, los usuarios del sistema se añaden al grupo nogroup.
# Para añadir el nuevo usuario del sistema a un grupo existente, usar las opciones --gid [GID] o --ingroup [GRUPO].
# Para añadir el nuevo usuario del sistema a un grupo con su mismo ID, use la opción --group.
sudo adduser --system peppa


man adduser / adduser --help
```

Cambiar la contraseña `passwd`

Ya vimos el comando passwd para cambiar la contraseña de root. Cualquier usuario puede **modificar su contraseña usando el comando** `passwd`.

Recordemos la sintaxis: **passwd [opciones] [USUARId maO]**

- "-d": borra la contraseña
- "-e": hace expirar la contraseña para que en el siguiente loguin el usuario solicite cambiar la password

```
# Para cambiar la contraseña de un usuario en particular, se ejecuta:
sudo passwd elenagg

# Se puede borrar la contraseña de un usuario
sudo passwd -d peppa

# Hace expirar la contraseña de un usuario
sudo passwd -e peppa

man passwd / passwd --help
man shadow
```

Eliminar usuarios `userdel`

```
# Elimina un usuario, pero conserva sus ficheros en /home/peppa
sudo userdel peppa

# Elimina un usuario, y elimina además el directorio /home/peppa. También se
elimina al usuario de los grupos a los que pertenecía.
sudo userdel -r peppa


man userdel / userdel --help
```

Añadir un usuario a un grupo `usermod`

Sintaxis:

usermod [opciones] USUARIO

Permite hacer modificaciones en la configuración del usuario: modificar el nombre de la cuenta, el directorio de trabajo, agrega el usuario a un grupo o más, modifica su grupo primario, permite añadir el usuario a varios grupos, modificar la fecha de expiración de la cuenta, etc..

Permite también cambiar la contraseña del usuario (opción -p) pero entonces queda en claro en el fichero /etc/shadow por eso el comando a utilizar para modificar la contraseña debe ser passwd.

Normalmente, sólo un usuario con privilegios puede hacer las modificaciones.

```
su -

# Modifica el nombre de la cuenta del usuario:
sudo usermod -l nombre_nuevo nombre_antiguo

# Añade un usuario a dos grupos (opción -a: append. opción -G: varios gruipos
```

```
separados por ,)
sudo usermod -aG grupo1,grupo2 usuario
# Ejemplo:
sudo usermod -aG developers,testers elenagg

# Usaremos la opción -G para determinar los grupos a los que queremos que
pertenezca un usuario.
# Sirve también para eliminar un usuario de un grupo.
# Si el usuario elenagg pertenece a los grupos "elenagg", "developers" y "testers"
y queremos elimiarlo del grupo developers, haremos:
sudo usermod -G elenagg,testers elenagg


# Modifica el directorio de trabajo de un usuario
sudo usermod -d /home/nuevodir usuario
# Ejemplo:
sudo usermod -d /home/new_peppa peppa


# Modificar el UID de un usuario
sudo usermod -u 1007 LRSO
```

Para modificar los grupos de un usuario también se puede utilizar el comando `gpasswd`

```
# Opción -a (append): Añade un usuario (user1) a un grupo (ungrupo1)
gpasswd -a user1 grupo1
# Ejemplo:
gpasswd elenagg developers

# Opción -M (members): Añadir varios usuarios a un grupo
gpasswd -M user1,user2,user3 grupo1

# Opción -d: Elimina un usuario de un grupo
gpasswd -d user2 grupo1


man usermod
man gpasswd
```

4. Permisos de ficheros y directorios en Linux

En Linux, **los permisos o privilegios que los usuarios pueden tener sobre determinados archivos se establecen en tres niveles claramente diferenciados.** Estos tres niveles son los siguientes:

- **Permisos del propietario (owner).** El propietario es aquel usuario que genera o crea un archivo/directorio dentro de su directorio de trabajo (HOME), o en algún otro directorio sobre el que tenga permiso.
- **Permisos del grupo (group).** Como hemos visto en los apartados previos, cada usuario pertenece a un grupo de trabajo. De esta forma, cuando se gestiona un grupo, se gestionan todos los usuarios que pertenecen a dicho grupo. Es decir, es más fácil integrar varios usuarios en un grupo al que se le conceden determinados privilegios en el sistema, que asignar los privilegios de forma independiente a cada usuario.
- **Permisos del resto de usuarios, o también llamados "otros" (others).** Por último, también los privilegios de los archivos o recursos, pueden tenerlos otros usuarios que no pertenezcan al grupo de trabajo del usuario propietario, sino que pertenecen a otros grupos. A estos se les denomina "resto de usuarios del sistema".

En los sistemas multiusuario siempre existe la figura del administrador, super usuario o root. Este administrador es el encargado de crear y dar de baja a usuarios, así como también, de establecer los privilegios (permisos o derechos) que cada uno de ellos tendrá en el sistema.

Para verlo de forma práctica veamos el listado de algunos ficheros y directorios:

```
ls -l /etc/passwd
-rw-r--r-- 1 root root 1395 sep 22 01:14 /etc/passwd

ls -ld /etc/
drwxr-xr-x 102 root root 8192 oct 13 05:21 /etc/
```

Los listados anteriores muestran información sobre el propietario y los permisos de los ficheros. Hagamos un repaso:

```
drwxr-xr-x 102 root  root    8192 oct 13 05:21  /etc/
__________     ____  _____
     |           |     |
     |           |     -> Grupo con permisos sobre el fichero
     |           |
     |           -> Usuario propietario del fichero
     |
     |
     -> Permisos del fichero:
       - Tipo de fichero (regular, directorio, enlace, etc).
       - Permisos del usuario/propietario ( r w x )
       - Permisos del grupo ( r - x )
```

```
        - Permisos para el resto de usuarios ( r - x )
```

Así que los permisos se agrupan para Usuario, Grupo y Otros usuarios:

- **owner/user** (propietario/usuario): Es el usuario que creó y posee un archivo/directorio.
- **group** (grupo): Todos los usuarios que son miembros del mismo grupo.
- **others** (otros): Todos los demás usuarios del sistema que no son propietarios ni miembros del grupo.

```
-  r w x     r - x     r - x
   _____     _____     _____
     |         |         |
     |         |          -> Permisos para Otros usuarios del sistema
     |         |
     |         -> Permisos para el Grupo propietario
     |
     -> Permisos para el Usuario propietario
```

Podemos pensar en represenar cada uno de estos permisos con un digito 0 ó 1, de forma que el si se tiene el permiso lo podríamos representar con un 1 y si no se tiene se podría representar con un 0.

Por ejemplo:

```
 -  r w x     r - x     r - x
    1 1 1     1 0 1     1 0 1
```

Según esto, podemos representar los permisos del usuario (r w x) con 3 digitos. Los permisos del grupo (r w x) con otros 3 digitos y para los permisos del resto de usuarios podemos utilizar también 3 digitos.

Normalmente, estos permisos se agrupan en 3 grupos de 3 digitos y se representan como 3 números en base octal. Para el ejemplo anterior, los permisos serían 755:

```
-  r w x     r - x     r - x
   1 1 1     1 0 1     1 0 1
   \___/     \___/     \___/
     7         5         5

Otro ejemplo:

-  r w -     r - -     r - -
   1 1 0     1 0 0     1 0 0
   \___/     \___/     \___/
     6         4         4
```

A modo de recordatorio, la siguiente tabla muestra la equivalencia entre valores expresados en base 2 (binarios) y valores expresados en base 8 (octal):

Octal	Binario	Permisos
0	0 0 0	- - -
1	0 0 1	- - x
2	0 1 0	- w -
3	0 1 1	- w x
4	1 0 0	r - -
5	1 0 1	r - x
6	1 1 0	r w -
7	1 1 1	r w x

-

Hay una pequeña diferencia entre el significado de los permisos para ficheros y directorios:

- **Ficheros**
    - **r**: El fichero puede ser leído (se lee su contenido).
    - **w**: El fichero puede ser escrito/modificado (se escribe/modifica su contenido).
    - **x**: El fichero puede ser ejecutado (el fichero es un programa o *script*).
- **Directorios**
    - **r**: Se puede listar el contenido del directorio (hacer un *ls*, por ejemplo).
    - **w**: Se pueden crear o eliminar ficheros dentro del directorio.
    - **x**: Se puede acceder al directorio, y a sus subdirectorios (hacer un *cd*).

Hay que tener en cuenta que, cuando no se tiene permiso de lectura de un directorio, no se pueden mostrar (listar) sus contenidos, pero sí se puede, por ejemplo, leer un fichero que esté dentro: tendríamos que usar la ruta absoluta para leer un fichero dentro de un diretcorio que no podemos listar. Esto implica que se debe conocer a priori el nombre completo del contenido de un directorio.

Además, aunque no se pueda leer (**r**) o escribir (**w**) en un directorio, siempre que se pueda acceder a él (**x**), se podrá trabajar con los ficheros que contiene. Dicho de otra manera, los permisos de un directorio no son los permisos de los ficheros (y sub-directorios) que contiene.

Recordatorio: al hacer un `ls -l` podemos ver de qué tipo es un fichero:

Código	Tipo de objeto
-	Fichero regular.
d	Directorio.
l	Enlace simbólico.

Código	Tipo de objeto
c	Dispositivo (especial) de carácter.
b	Dispositivo (especial) de bloque.
p	FIFO.
s	Socket.

Cambiar los permisos de ficheros y directorios

Comando `chmod`

Se usa la herramienta **`chmod`** (*change file mode bits*) para cambiar los permisos. Los argumentos con los que se invoca a *chmod* son:

- ¿Para quién se cambian los permisos?: usuario, grupo, otros, todos. [**ugoa**].
- ¿Se conceden permisos **[+]** o se revocan permisos **[-]**?
- ¿Qué permiso se configura?: lectura, escritura y/o ejecución [**rwx**]

A continuación se muestran algunos ejemplos de permisos para ficheros:

```
cd
mkdir permisos
cd permisos
echo "Este es un fichero para jugar con chmod" > permisos.txt

# Vamos a cambiar algunos permisos

ls -l
# -rw-rw-r-- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod o-r permisos.txt
ls -l
# -rw-rw---- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod g+x permisos.txt
ls -l
# -rw-rwx--- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod u-r permisos.txt
ls -l
# --w-rwx--- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt


# También podemos cambiar varios permisos a la vez

chmod u+rx permisos.txt
ls -l
# -rwxrwx--- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod go-wx permisos.txt
```

```
ls -l
# -rwxr----- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod a-x permisos.txt
ls -l
# -rw-r----- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod a+rw permisos.txt
ls -l
# -rw-rw-rw- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt
```

También es común usar los valores expresados en base octal para cambiar los los permisos:

```
chmod 755 permisos.txt
ls -l
# -rwxr-xr-x 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod 750 permisos.txt
ls -l
# -rwxr-x--- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod 644 permisos.txt
ls -l
# -rw-r--r-- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod 650 permisos.txt
ls -l
# -rw-r-x--- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

chmod 666 permisos.txt
ls -l
# -rw-rw-rw- 1 elenagg elenagg 40 nov 10 11:26 permisos.txt
```

Y para comprender mejor cómo funcionan los permisos en los directorios, veamos algunos ejemplos:

```
cd
ls -l permisos
# -rwxr-xr-x 1 elenagg elenagg 40 nov 10 11:26 permisos.txt

ls -ld permisos
# drwxrwxr-x 2 elenagg elenagg 4096 nov 11 20:34 permisos/

chmod 400 permisos
ls -ld permisos
# Modificamos el permiso del directorio ("r") y vemos que podemos hacer
# dr-------- 2 elenagg elenagg 4096 nov 11 20:34 permisos/
```

```
cd permisos/
# No puedo acceder al directorio (no hay permiso "x" para el usuario)
# bash: cd: permisos/: Permiso denegado

ls permisos/
# Sí Puedo listar su contenido
# ls: no se puede acceder a permisos/permisos.txt: Permiso denegado
# permisos.txt

chmod 100 permisos
ls -ld permisos
# Modificamos ahora el directorio y le damos permiso "x". Vemos que pasa...
# d--x------ 2 elenagg elenagg 4096 nov 11 20:34 permisos/

ls permisos/
# No puedo listar su contenido
# ls: no se puede abrir el directorio permisos/: Permiso denegado

cd permisos
# Sí puedo acceder al directorio pero no puedo listar el contenido

ls
# ls: no se puede abrir el directorio .: Permiso denegado

pwd
#/home/elenagg/MisPruebas/permisos
# Puedo accder a sus ficheros siempre que conozca el nombre completo
cat permisos.txt
# Este es un fichero para jugar con chmod


man chmod
```

Propietarios de un fichero o un directorio

Recordemos todo lo que hemos aprendido:

- **Un fichero pertenece a un usuario** y a un **grupo**. Los usuarios y los grupos tienen asociados un identificador numérico, llamados *Id. de usuario* (**UID**) e *Id. de grupo* (**GID**).
- **Los usuarios del sistema están definidos en los ficheros** `/etc/passwd` y `/etc/shadow`. En este último estan encriptadas las contraseñas de los usuarios y su periodo de vigencia y sólo tiene acceso el usuario root.
- **Los grupos están definidos en los ficheros** `/etc/group` y `/etc/gshadow`. En este último se guardan los usuarios que están dados de alta dentro de cada grupo de usuarios asi como los admisnitradores de cada grupo y sólo tiene permiso de acceso el usuario root.

Veamos los siguientes ejemplos para introducri un nuevo comando:

```
# Buscamos el usuario en el fichero de usuarios
cat /etc/passwd | grep elenagg
# El usuario elenagg aparece en una línea

elenagg:x:1000:1000:Elena GG,,,:/home/elenagg:/bin/bash
```

```
# Buscamos el grupo elenagg en el fichero de grupos
cat /etc/group | grep elenagg

# El usuario elenagg aparece en varias líneas, en una de las líneas aparece como
grupo y en el resto como usuario que pertenece a otros grupos.

adm:x:4:syslog,elenagg
cdrom:x:24:elenagg
sudo:x:27:elenagg
dip:x:30:elenagg
plugdev:x:46:elenagg
lpadmin:x:120:elenagg
lxd:x:131:elenagg
elenagg:x:1000:
sambashare:x:132:elenagg
vboxsf:x:998:elenagg
```

- En el primero ejemplo, se puede ver que el usuario *elenagg* tiene *UID = 1000* y *GID = 1000*.
- En el segundo ejemplo se puede ver que el grupo *vboxsf* tiene *GID = 998* y que el usuario *elenagg* pertenece al dicho grupo *vboxsf*.

Cambiar el propietario y el grupo de un fichero o directorio

Comando `chown`

Sintaxis: chown [owner][:group] [file name]

```
su -

# Crearemos antes un grupo para poder ver los ejemplos (el comando de creación de
grupos se ve más adelante)
groupadd developers

# Cambia el usuario y el grupo de un fichero o directorio
chown elenagg:developers /home/elenagg/license.txt
chown elenagg:developers /home/elenagg/projects/

# Cambia solo el grupo
```

```
chown :developers /home/elenagg/license.txt

# Cambia solo el usuario
chown lola /home/elenagg/license.txt

# Cambia el usuario y el grupo de un directorio y todo su contenido
chown -R elenagg:developers /home/elenagg/projects
```

Ejercicios

Actividades de clase.

Para ampliar

El tema de gestión de usuarios da para varias asignaturas, pero tras esta primera introducción, se puede buscar información o profundizar más acerca de **suid**, **sgid**, **sticky bit**, **ficheros inmutables**.

Mucha de la configuración relativa a usuarios y grupos se guarda en el fichero `/etc/login.defs` 😀

`umask` El comando umask, es la abreviatura de user file-creation mode mask, y sirve para establecer los permisos por defecto que tendrán los nuevos ficheros y directorios que creemos.

Algunos recursos útiles

- Permissions - Ryans Tutorials
- Permissions - IBM Developer