



MINISTERIO
DE EDUCACIÓN
Y CIENCIA

SECRETARÍA GENERAL
DE EDUCACIÓN
Y FORMACIÓN PROFESIONAL

DIRECCIÓN GENERAL
DE EDUCACIÓN,
FORMACIÓN PROFESIONAL
E INNOVACIÓN EDUCATIVA

CENTRO NACIONAL
DE INFORMACIÓN Y
COMUNICACIÓN EDUCATIVA

Redes de área local Aplicaciones y Servicios Linux

Usuarios del sistema Unix



SERVICIO DE
FORMACIÓN DEL
PROFESORADO

C/ TORRELAGUNA, 58
28027 - MADRID

Índice de contenido

Usuarios y grupos de usuarios en Unix.....	3
Cuentas de usuario.....	3
Grupos de usuarios.....	6
Usuario root.....	6
Administración de usuarios y grupos.....	7
Creación de usuarios.....	7
Comando adduser.....	7
Comando useradd.....	8
Modificación de usuarios.....	8
Eliminación de usuarios.....	8
Creación de grupos.....	9
Comando addgroup.....	9
Comando groupadd.....	9
Modificación de grupos.....	9
Eliminación de grupos.....	9
Añadir usuarios a un grupo.....	10
Quitar usuarios de un grupo.....	10
Herramienta gráfica de administración de usuarios.....	10
Permisos de archivos y carpetas.....	11
Usuario propietario y grupo propietario de un archivo.....	11
Tipos de permisos.....	12
Permiso de lectura.....	12
Permiso de escritura.....	12
Permiso de ejecución.....	13
¿A quién se puede otorgar permisos?.....	13
Visualizar los permisos de un archivo o carpeta.....	14
Cambio de permisos.....	15
Bits SUID y SGID.....	18
Máscaras.....	18
Grupos privados de usuario.....	19
Cambiar usuario propietario y grupo propietario.....	20

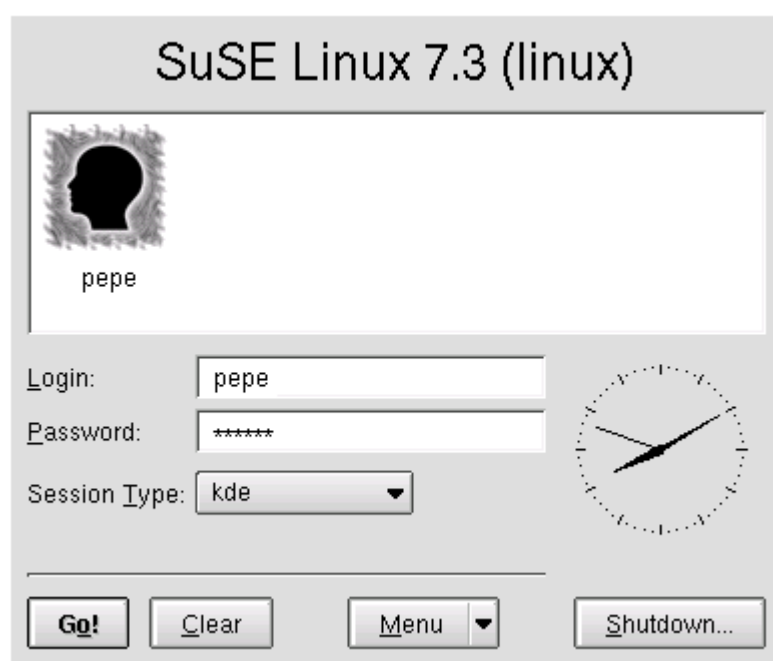
Usuarios y grupos de usuarios en Unix

El sistema Unix es un sistema operativo multiusuario. Linux está basado en el sistema Unix. Para que múltiples usuarios puedan hacer uso del sistema de una forma segura y ordenada, es necesario que el sistema disponga de mecanismos de administración y seguridad para proteger los datos de cada usuario, así como para proteger y asegurar el correcto funcionamiento del sistema.

Cuentas de usuario

Para poder utilizar el sistema operativo Unix es necesario disponer de una cuenta de usuario que se compone de **nombre de usuario (login)** y de **contraseña (password)**. Las cuentas de usuario son creadas por el administrador que en Unix es un usuario especial llamado **root** (ver más abajo). Los usuarios deberán pertenecer al menos a un grupo de usuarios ya que obligatoriamente deben tener asignado un grupo principal o grupo primario.

Cuando un usuario entra en un sistema Unix, debe identificarse indicando su **nombre** de usuario (en inglés '**login**') y su **contraseña** (en inglés '**password**'). Si se equivoca al introducir su nombre o su contraseña, el sistema le denegará el acceso y no podrá entrar.



Inicio de sesión en Linux

Una vez se haya identificado de forma satisfactoria, el usuario podrá utilizar el sistema y ejecutar todas las aplicaciones que le sean permitidas, así como leer, modificar o borrar aquellos archivos sobre los cuales tenga permiso.

Las cuentas de usuario no solo ofrecen al usuario un nombre y una contraseña, también le proporciona una **ruta para almacenar sus documentos** y su perfil generalmente dentro de la carpeta `/home/nombre-usuario` y comúnmente denominada **carpeta home del usuario** y un **intérprete de comandos (shell)** que le permitirá ejecutar aplicaciones.

Cuando el usuario ejecuta una aplicación, el sistema carga la aplicación en memoria y la ejecuta. En el argot informático a las aplicaciones que se están ejecutando en un momento determinado se les denomina **procesos**. Los procesos en ejecución pertenecen a algún usuario. El sistema asigna a los procesos el usuario que los ejecuta. Ejemplo, si el usuario "pepe" ejecuta la aplicación "konqueror", en la lista de procesos del sistema aparecerá un nuevo proceso llamado "konqueror" cuyo propietario es "pepe".

Obligatoriamente, todos los procesos del sistema pertenecen a algún usuario. Ejecutando el comando 'ps aux' podemos ver todos los procesos en ejecución. Si ejecutamos el comando 'top' lo veremos a tiempo real.

```

top - 10:59:08 up 41 min, 0 users, load average: 0.00, 0.00, 0.00
Tasks: 37 total, 2 running, 35 sleeping, 0 stopped, 0 zombie
Cpu(s):  0.7% user,  2.3% system,  0.0% nice, 97.0% idle
Mem:   191388k total, 178944k used, 12444k free,  6688k buffers
Swap:   0k total,  0k used,  0k free, 91456k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 347  root        16   0 19164  16m 1904  S   1.0   9.0   1:13.60 XFree86
 584  knoppix     12   0 18996  18m  15m  S   0.7   9.9   0:11.46 kdeinit
3110  root        15   0  1032 1032  844  R   0.7   0.5   0:00.30 top
   1  root         9   0    76   76   52  S   0.0   0.0   0:05.08 init
   2  root         9   0     0     0     0  S   0.0   0.0   0:00.01 keventd
   3  root        18  19     0     0     0  S   0.0   0.0   0:00.00 ksoftirqd_CPU0
   4  root         9   0     0     0     0  S   0.0   0.0   0:00.09 kswapd
   5  root         9   0     0     0     0  S   0.0   0.0   0:00.00 bdflush
   6  root         9   0     0     0     0  S   0.0   0.0   0:00.06 kupdated
  12  root         9   0     0     0     0  S   0.0   0.0   0:00.00 khubd
 223  root         9   0   696   696   592  S   0.0   0.4   0:00.00 pump
 277  root        15   0   636   636   628  S   0.0   0.3   0:03.48 automount
 282  root         8   0  2516  2516  1228  S   0.0   1.3   0:00.20 bash
 283  root         8   0  2516  2516  1228  S   0.0   1.3   0:00.15 bash
 284  root         8   0  2516  2516  1228  S   0.0   1.3   0:00.13 bash
 285  root         8   0  2516  2516  1228  S   0.0   1.3   0:00.11 bash
 336  root         9   0  1344  1344  1232  S   0.0   0.7   0:00.06 xsession
 345  root         9   0  1348  1348  1236  S   0.0   0.7   0:00.01 xsession
 359  knoppix      9   0  1372  1372   980  S   0.0   0.7   0:00.09 xinitrc

```

Mostrando procesos con top. La segunda columna indica el propietario

Cuando se crea un nuevo archivo, el propietario del archivo será el **usuario** que lo ha creado y el grupo del archivo será el **grupo principal** de dicho usuario. Ejemplo, si "pepe" cuyo grupo principal es "profes" crea un nuevo archivo llamado examen.txt, el propietario de examen.txt será "pepe" y el grupo propietario será "profes", o lo que es lo mismo, el archivo pertenecerá al usuario pepe y al grupo profes. **Obligatoriamente, todos los archivos del sistema pertenecen a algún usuario y a algún grupo.**

```

Bash
pepe@0[pepe]$ ls -l
-rw-rw-r-- 1 pepe  profes    11 Feb 12 18:28 examen.txt
pepe@0[pepe]$

```

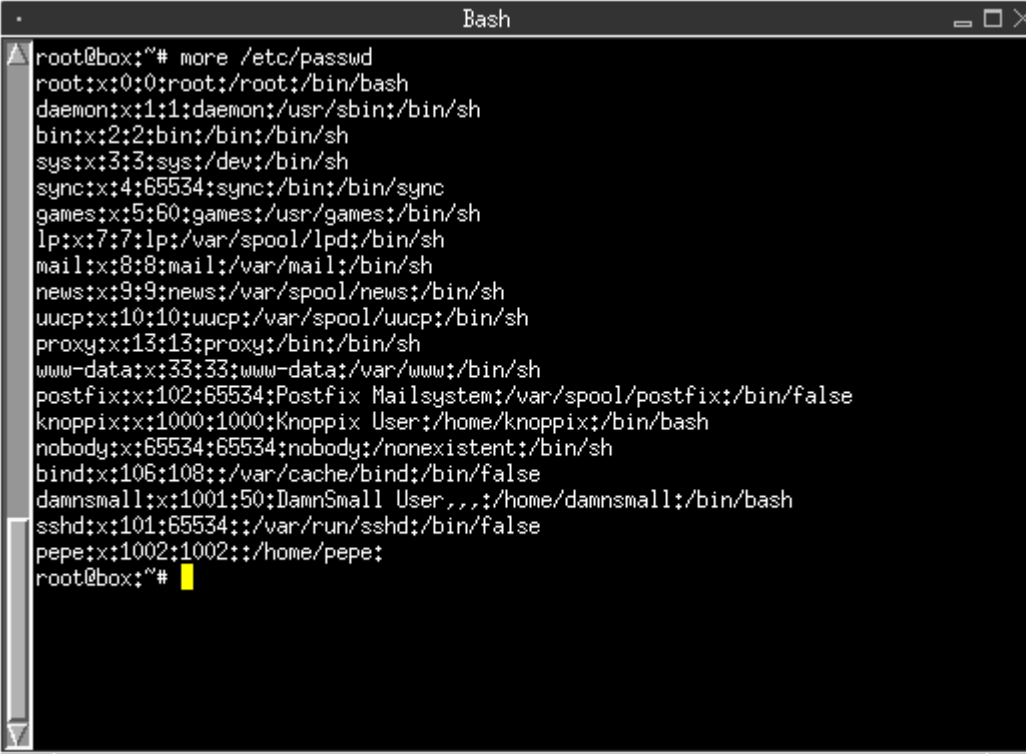
examen.txt pertenece al usuario pepe y al grupo profes

La cuenta de usuario le permite acceder al sistema tanto de forma presencial (sentado delante del ordenador) como de forma remota accediendo desde otro equipo por la red. Los permisos que tiene el usuario cuando utiliza el sistema presencialmente son los mismos que tiene cuando lo hace remotamente. Lo habitual es utilizar el sistema de forma remota ya que al ser Unix un sistema multiusuario, la única forma de que varios usuarios lo utilicen de forma simultánea es remotamente.

El sistema Unix codifica los usuarios con un número diferente a cada uno que es el **identificador de usuario (uid = User Identifier)**. Internamente el sistema trabaja con el uid, no con el nombre del usuario. Normalmente a los usuarios que creamos se les asignan uids desde 1000 en adelante. Los números uid menores que 100 se reservan para usuarios especiales del sistema.

En Unix por defecto, la información de los usuarios de un sistema se guarda en el archivo **/etc/passwd**. Es un archivo de texto que puede visualizarse con cualquier editor. Cada línea del archivo /etc/passwd

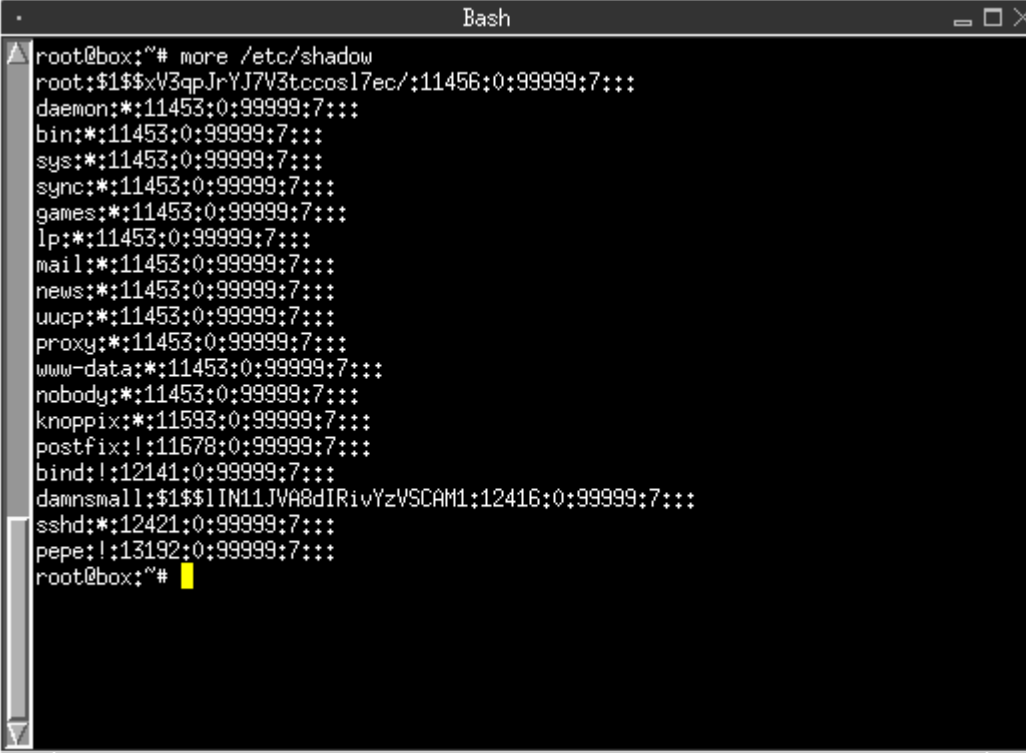
almacena los parámetros de un usuario. Solo puede modificarlo el administrador (root). A continuación mostramos el archivo passwd:

A terminal window titled 'Bash' showing the command 'more /etc/passwd' and its output. The output lists system users (root, daemon, bin, sys, sync, games, lp, mail, news, uucp, proxy, www-data, postfix, knoppix, nobody, bind, damnsnall) and regular users (sshd, pepe) with their respective IDs, groups, home directories, and shells. The terminal ends with a root prompt.

```
root@box:~# more /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
postfix:x:102:65534:Postfix Mailsystem:/var/spool/postfix:/bin/false
knoppix:x:1000:1000:Knoppix User:/home/knoppix:/bin/bash
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
bind:x:106:108::/var/cache/bind:/bin/false
damnsnall:x:1001:50:DamnSmall User,,,:/home/damnsnall:/bin/bash
sshd:x:101:65534::/var/run/sshd:/bin/false
pepe:x:1002:1002::/home/pepe:
root@box:~#
```

Volcado del archivo /etc/passwd

Las contraseñas de cada usuario se guardan encriptadas con un sistema de codificación irreversible, en el archivo **/etc/shadow** que también es un archivo de texto.

A terminal window titled 'Bash' showing the command 'more /etc/shadow' and its output. The output shows the encrypted password hashes for the same set of users as the /etc/passwd file, along with fields for password expiration and minimum age. The terminal ends with a root prompt.

```
root@box:~# more /etc/shadow
root:$1$XV3qpJrYJ7V3tccosl7ec/:11456:0:99999:7:::
daemon*:11453:0:99999:7:::
bin*:11453:0:99999:7:::
sys*:11453:0:99999:7:::
sync*:11453:0:99999:7:::
games*:11453:0:99999:7:::
lp*:11453:0:99999:7:::
mail*:11453:0:99999:7:::
news*:11453:0:99999:7:::
uucp*:11453:0:99999:7:::
proxy*:11453:0:99999:7:::
www-data*:11453:0:99999:7:::
nobody*:11453:0:99999:7:::
knoppix*:11593:0:99999:7:::
postfix:!:11678:0:99999:7:::
bind:!:12141:0:99999:7:::
damnsnall:$1$1IN11JVA8dIRivYzVSCAM1:12416:0:99999:7:::
sshd*:12421:0:99999:7:::
pepe:!:13192:0:99999:7:::
root@box:~#
```

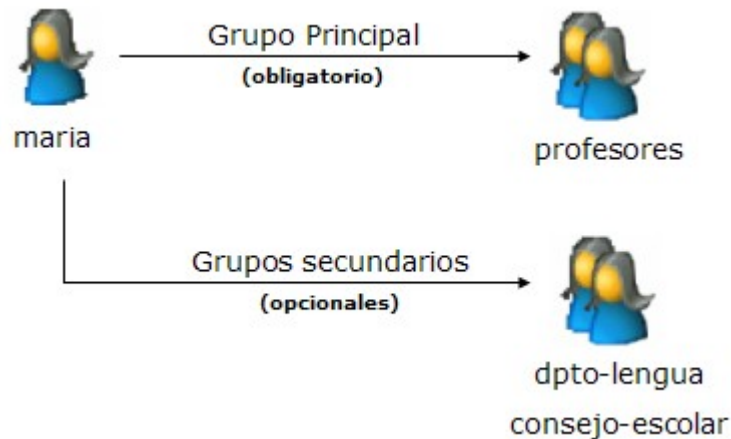
Volcado del archivo /etc/shadow

Grupos de usuarios

Para poder administrar los permisos de los usuarios de una forma más flexible, el sistema Unix permite la organización de usuarios en grupos y establecer permisos a los grupos.

Ejemplo, si en un centro educativo el grupo "profesores" tiene acceso a ciertas carpetas, cuando demos de alta un profesor nuevo, tan solo tendremos que añadirle al grupo "profesores" para que pueda acceder a todas esas carpetas. Es lo que se denomina administración de permisos por grupos.

Todos los usuarios pertenecen al menos a un grupo que es el **grupo principal del usuario**, también llamado grupo primario del usuario, pero pueden pertenecer a más grupos. En caso de que pertenezcan a más grupos, éstos serán **grupos secundarios**.



Todo usuario debe pertenecer a un grupo principal obligatoriamente

Los grupos pueden contener varios usuarios. Los grupos de usuarios solo pueden contener usuarios, nunca podrán contener a otros grupos.

El sistema Unix codifica los grupos de usuarios con un número diferente a cada uno que es el **identificador de grupo (gid = Group Identifier)**. Internamente el sistema trabaja con el gid, no con el nombre del grupo. Normalmente a los grupos que creamos se les asignan gids desde 1000 en adelante. Los números gid menores que 100 se reservan para grupos especiales del sistema.

En Unix por defecto, la información de los grupos de un sistema se guarda en el archivo **/etc/group**. Es un archivo de texto que puede visualizarse con cualquier editor. Cada línea del archivo **/etc/group** almacena los parámetros del grupo y los usuarios que contiene. Solo puede modificarlo el administrador (root). Las contraseñas de los grupos se guardan encriptadas con un sistema de codificación irreversible, en el archivo **/etc/gshadow** que también es un archivo de texto.

Usuario root

El usuario root, a veces llamado **superusuario**, es el usuario administrador del sistema. Está identificado con el **número de usuario cero (uid=0)** y tiene permisos sobre todo el sistema sin ningún tipo de restricción. El usuario root puede acceder a cualquier archivo, ejecutar, instalar y desinstalar cualquier aplicación, modificar los archivos de configuración del sistema y administrar usuarios. Si dispones de la contraseña de root tendrás control total sobre todo el sistema.

Administración de usuarios y grupos

En nuestro centro tanto el profesorado como el alumnado acostumbran a personalizar su escritorio, incluyendo accesos directos, carpetas y demás vínculos que considera importantes y cómodos en su tarea diaria. Pero todo esto incomoda, en muchas ocasiones, a otros usuarios que también utilizan el mismo equipo. Esta situación es especialmente patente en las aulas de informática, donde parece que el alumnado compite en crear ámbitos de trabajo extravagantes e inútiles, con la pérdida de tiempo en la carga del sistema operativo y que posteriormente repercute en nuestro trabajo al tener que estar reinstalando equipos, eliminando programas, etc.

Para resolver el problema anterior es necesario disponer de una base de datos de usuarios y grupos donde poder asignar o denegar permisos de acceso a los recursos autenticados de nuestro centro. Ello permitirá personalizar los entornos de trabajo de cada usuario, dando solución al problema planteado tanto por el profesorado como por el alumnado.

La administración de usuarios y grupos **solamente puede realizarlas el usuario root** y comprende las siguientes tareas:

- Creación de usuarios
- Modificación de usuarios
- Eliminación de usuarios
- Creación de grupos
- Modificación de grupos
- Eliminación de grupos
- Añadir usuarios a un grupo
- Quitar usuarios de un grupo

Para realizar estas tareas, el administrador dispone de una serie de comandos y también de alguna aplicación gráfica que veremos a continuación.

Creación de usuarios

Para la creación de un usuario se dispone de dos comandos que son `adduser` y `useradd`.

Comando `adduser`

El comando `adduser` permite agregar usuarios de una forma interactiva. Tan solo hay que ejecutar el comando e ir respondiendo a las preguntas. Veamos un ejemplo de utilización:

// Ejemplo de utilización de `adduser`

```
root@knoppix37:~# adduser
Ingrese un nombre de usuario a añadir: luis
Añadiendo usuario luis...
Adding new group `luis' (1007).
Adding new user `luis' (1007) with group `luis'.
Creando el directorio home /home/luis.
Copiando archivos desde /etc/skel
Enter new UNIX password: // Contraseña del usuario
Retype new UNIX password: // Repetir la contraseña
passwd: password updated successfully
Cambiando la información de usuario para luis
Introduzca el nuevo valor,: o presione ENTER para el predeterminado
Nombre completo []: Luis Ruiz
Número de habitación []: // Intro
Teléfono del trabajo []: // Intro
Teléfono de casa []: // Intro
Otro []: // Intro
¿Es correcta la información? [y/N] y
root@knoppix37:~#
```

En la secuencia anterior, al proporcionar el nombre de usuario vemos que ha creado un grupo llamado igual y ha añadido al usuario a dicho grupo. Es lo que se denomina grupos privados de usuario cuya utilidad analizaremos posteriormente. Vemos también que ha copiado una serie de carpetas y archivos de la carpeta `/etc/skel` que es donde se almacena el esqueleto del perfil de usuario. Nos ha pedido dos veces la contraseña (por si nos equivocamos) y nos ha hecho preguntas de información adicional no obligatoria.

Se recomienda que el nombre de usuario sea en minúsculas y además de letras también puede contener números y algún signo como guiones normales y guiones bajos. Debemos recordar que unix distingue entre mayúsculas y minúsculas, es decir, Pepe es distinto de pepe.

El comando `adduser` se utiliza cuando añadimos usuarios de forma esporádica y de uno en uno.

Comando `useradd`

El comando `useradd` permite añadir un usuario indicando como parámetros la información particular para crear el usuario en la misma línea de comandos. La sintaxis es:

```
#useradd [opciones] nombre-usuario
```

Entre las opciones más destacables tenemos:

- `-g`: Grupo principal que queremos tenga el usuario (debe existir)
- `-d`: Carpeta home del usuario. Suele ser `/home/nombre-usuario`
- `-m`: Crear carpeta home si es que no existe.
- `-s`: Intérprete de comandos (shell) del usuario. Suele ser `/bin/bash`

Ejemplo, si deseamos crear un usuario llamado 'pedro' cuyo grupo principal sea 'profesores', cuya carpeta home sea `/home/pedro` y su intérprete de comandos sea `/bin/bash`, ejecutaremos el siguiente comando:

```
// Crear un usuario
```

```
# useradd -g profesores -d /home/pedro -m -s /bin/bash pedro
```

De ésta manera habremos creado al usuario pedro y su carpeta home. Tan solo nos quedará establecer su contraseña con el comando `passwd`:

```
// Establecer la contraseña del usuario
```

```
# passwd pedro
```

Entonces el sistema nos preguntará dos veces la contraseña que queremos asignar a pedro.

El comando `useradd` se suele utilizar cuando queremos crear muchos usuarios automáticamente mediante **archivos de comandos (scripts)**.

Modificación de usuarios

Se utiliza el comando `usermod` y permite cambiar el nombre del usuario, su carpeta home, su intérprete de comandos y algunos otros parámetros.

```
// Cambiar el home de un usuario
```

```
# usermod -d /home/carpeta_pedro pedro
```

Eliminación de usuarios

Se realiza con el comando `userdel` seguido del nombre del usuario. Con la opción `-r` eliminará también su carpeta home, ejemplo:

```
// Eliminación de un usuario
```

```
# userdel -r pedro
```


Eliminaría el usuario pedro y su carpeta home.

Creación de grupos

Para la creación de un grupo se dispone de dos comandos que son `addgroup` y `groupadd`.

Comando `addgroup`

Permite agregar grupos de una forma interactiva. Tan solo hay que ejecutar el comando y nos preguntará el nombre del grupo. Veamos un ejemplo de utilización:

// Creación de un grupo de usuarios

```
root@knoppix37:~# addgroup
Ingrese un nombre de grupo a añadir: alumnos
Adding group `alumnos' (1010)...
Hecho.
root@knoppix37:~#
```

El nombre del grupo debe ser en minúsculas y además de letras también puede contener números y algún signo como guiones normales y guiones bajos.

Comando `groupadd`

Permite añadir un grupo indicando como parámetro el nombre del grupo. Ejemplo, si deseamos crear un grupo llamado 'alumnos' ejecutaremos:

// Añadir un grupo

```
# groupadd alumnos
```

El comando `groupadd` se suele utilizar cuando queremos crear muchos grupos automáticamente mediante archivos de comandos (scripts).

Modificación de grupos

El comando `groupmod` permite modificar el nombre de un grupo o el gid del mismo. La sintaxis es:

```
# groupmod [-g nuevo-gid] [-n nuevo-nombre] nombre-grupo
```

// Cambiar el gid del grupo profesores

```
# groupmod -g 2000 profesores
```

Eliminación de grupos

Se realiza con el comando `groupdel` seguido del nombre del grupo, ejemplo:

// Eliminación de un grupo

```
# groupdel profesores
```

Eliminaría el grupo profesores. Si algún usuario tuviera dicho grupo como grupo primario, el comando `groupdel` **no** eliminará el grupo.

Añadir usuarios a un grupo

Se utiliza nuevamente el comando **adduser** seguido del nombre del usuario y del nombre del grupo al que queremos añadirle, ejemplo:

```
// Añadir un usuario a un grupo
```

```
# adduser juan profesores // añade juan al grupo profesores
```

Quitar usuarios de un grupo

Se utiliza el comando **deluser** seguido del nombre del usuario y del nombre del grupo del que queremos quitarle, ejemplo:

```
// Quitar a un usuario de un grupo
```

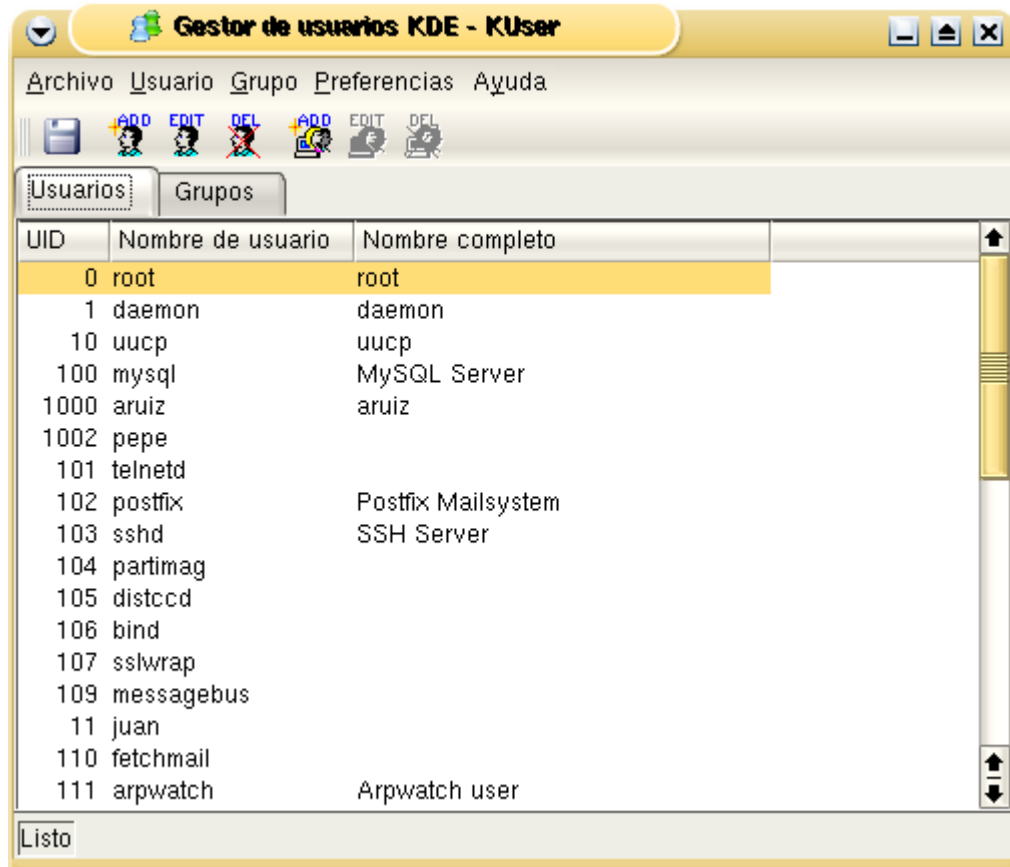
```
# deluser juan profesores // quita a juan del grupo profesores
```

Para más información de todos estos comandos se puede consultar la ayuda del manual ejecutando *man* seguido del nombre del comando, ejemplo *man adduser*.

Herramienta gráfica de administración de usuarios

Knoppix dispone de una herramienta gráfica de administración de usuarios que es la aplicación 'kuser'. Para ejecutarla podemos abrir una consola de root y ejecutar **kuser** o si hemos iniciado sesión como root, podemos pulsar **Alt+F2** y ejecutar **kuser**. También podemos ir a **K > Sistema > Administrador de usuarios**.

La primera vez que ejecutamos **kuser** nos advertirá que no están configurados los archivos que almacenan la información de los usuarios y los grupos pero por defecto se autoconfigura con los archivos **/etc/passwd** y **/etc/group**. Después aparecerá la pantalla principal de **kuser**:



Se trata de una herramienta bastante intuitiva que dispone de una pestaña para administrar usuarios y otra para administrar grupos. Con los botones de la barra de herramientas se pueden realizar prácticamente todas las acciones que permite el programa. El significado de dichos botones se puede observar en la siguiente figura.



Permisos de archivos y carpetas

Usuario propietario y grupo propietario de un archivo

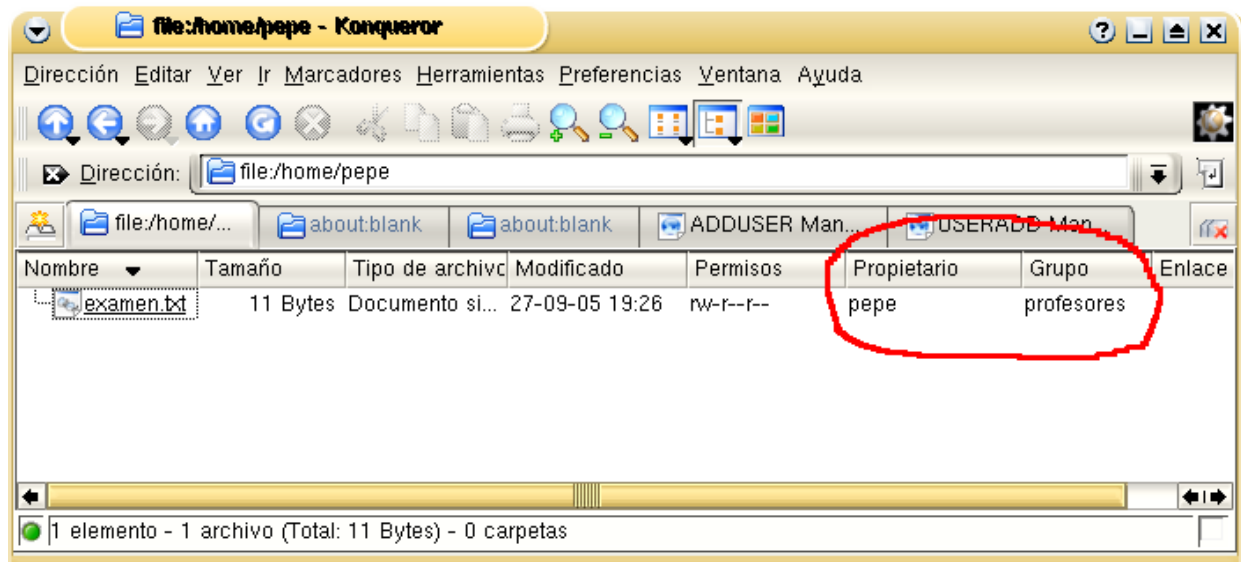
Anteriormente se ha comentado que en Unix todos los archivos pertenecen obligatoriamente a un usuario y a un grupo. Cuando un usuario crea un nuevo archivo, el propietario del archivo será el usuario que lo ha creado y el grupo del archivo será el grupo principal de dicho usuario.

Ejemplo, si un usuario llamado 'pepe' cuyo grupo principal es el grupo 'profesores' crea un nuevo archivo, el propietario del archivo será 'pepe' y el grupo propietario del archivo será 'profesores', o lo que es lo mismo, el archivo pertenecerá al usuario pepe y al grupo profesores. Obligatoriamente, todos los archivos del sistema pertenecen a algún usuario y a algún grupo.

Con el comando **ls** añadiendo la opción **-l** (formato largo) podemos visualizar el usuario propietario y el grupo propietario del archivo, ejemplo:

```
pepe@knoppix37:~$ ls -l
total 4
-rw-r--r-- 1 pepe profesores 11 2005-09-27 19:26 examen.txt
pepe@knoppix37:~$
```

Comprobamos que el usuario propietario es pepe y el grupo propietario es profesores. La misma información podemos verla desde el administrador de archivos 'konqueror' si vamos a la carpeta /home/pepe y activamos la opción 'vista en árbol' yendo a **Ver > Modo de vista:**



Tipos de permisos

En los Sistemas Unix, la gestión de los permisos que los usuarios y los grupos de usuarios tienen sobre los archivos y las carpetas, se realiza mediante un sencillo esquema de tres tipos de permisos que son:

- Permiso de **lectura**
- Permiso de **escritura**
- Permiso de **ejecución**

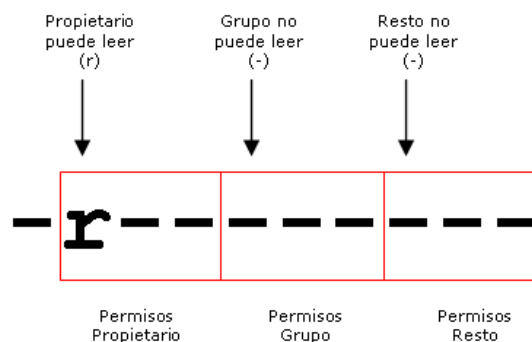
El significado de éstos permisos difiere si se tienen sobre archivos o sobre carpetas. A continuación veremos el significado para cada uno de los casos:

Permiso de lectura

Cuando un usuario tiene **permiso de lectura de un archivo** significa que puede **leerlo** o **visualizarlo**, bien sea con una aplicación o mediante comandos. Ejemplo, si tenemos permiso de lectura sobre el archivo `examen.txt`, significa que podemos ver el contenido del archivo. Si el usuario no tiene permiso de lectura, no podrá ver el contenido del archivo.

Cuando un usuario tiene **permiso de lectura de una carpeta**, significa que puede visualizar el contenido de la carpeta, es decir, puede ver los archivos y carpetas que contiene, bien sea con el comando `'ls'` o con un explorador de archivos como Konqueror. Si el usuario no tiene permiso de lectura sobre la carpeta, no podrá ver lo que contiene.

El permiso de lectura se simboliza con la letra 'r' del inglés 'read'.



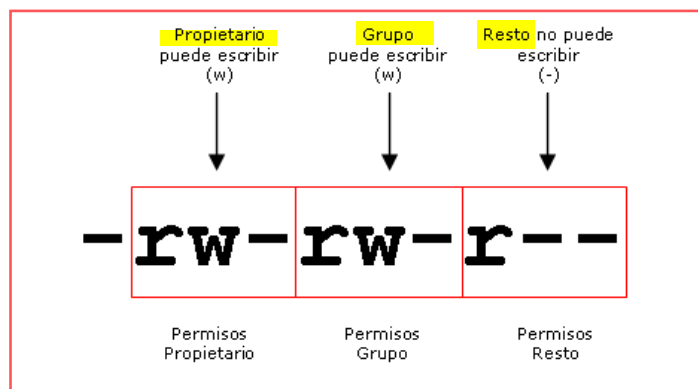
Permiso de escritura

Cuando un usuario tiene permiso de escritura sobre un archivo **significa** que **puede modificar su contenido**, e **incluso borrarlo**. También le da derecho a **cambiar los permisos del archivo** mediante el comando `chmod` así

como **cambiar su propietario y el grupo propietario** mediante el comando **chown**. Si el usuario no tienen permiso de escritura, no podrá modificar el contenido del archivo.

Cuando un usuario tiene permiso de escritura sobre una carpeta, significa que puede modificar el contenido de la carpeta, es decir, puede crear y eliminar archivos y otras carpetas dentro de ella. Si el usuario no tiene permiso de escritura sobre la carpeta, no podrá crear ni eliminar archivos ni carpetas dentro de ella.

El permiso de **escritura** se simboliza con la letra '**w**' del inglés '**write**'.



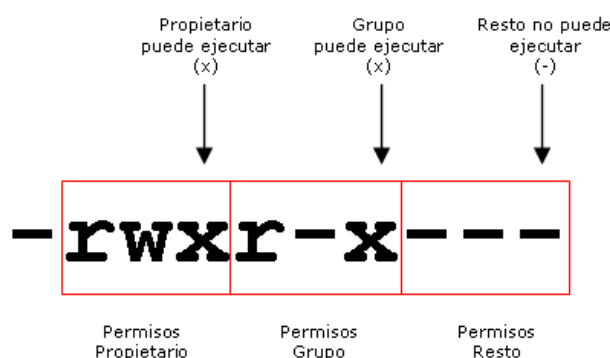
Permiso de ejecución

Cuando un usuario tiene permiso de ejecución de un archivo **significa que puede ejecutarlo**. Si el usuario no dispone de permiso de ejecución, no podrá ejecutarlo aunque sea una aplicación.

Los únicos archivos ejecutables son las aplicaciones y los archivos de comandos (scripts). Si tratamos de ejecutar un archivo no ejecutable, dará errores.

Cuando un usuario tiene permiso de ejecución sobre una carpeta, significa que puede entrar en ella, bien sea con el comando 'cd' o con un explorador de archivos como Konqueror. Si no dispone del permiso de ejecución significa que no puede ir a dicha carpeta.

El permiso de **ejecución** se simboliza con la letra '**x**' del inglés '**eXecute**'.



¿A quién se puede otorgar permisos?

Los permisos solamente pueden ser otorgados a **tres tipos o grupos de usuarios**:

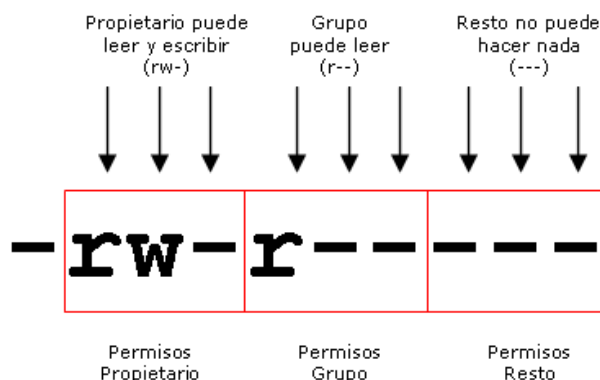
- Al **usuario propietario** del archivo
- Al **grupo propietario** del archivo
- Al **resto de usuarios** del sistema (todos menos el usuario propietario y los usuarios que estén en el grupo propietario)

Se pueden dar permisos de lectura, escritura, ejecución ó combinación de ambos al usuario propietario del archivo, al grupo propietario del archivo o al resto de usuarios del sistema. En Unix no existe la posibilidad de asignar permisos a usuarios concretos ni a grupos concretos, tan solo se puede asignar permisos al usuario propietario, al grupo propietario o al resto de usuarios.

Ejemplo, si disponemos de un archivo llamado '**examen.tx**' cuyo propietario es '**pepe**' y cuyo grupo propietario es '**profesores**', se pueden dar permisos de lectura, escritura, ejecución ó combinación de ambos

al usuario 'pepe', al grupo 'profesores' y al resto de usuarios, pero no podríamos dar permisos a otros usuarios distintos de pepe (juan, luis, pedro,...) ni a otros grupos (alumnos, directivos, personal,...) ya que el esquema Unix no lo permite.

Supongamos que la siguiente figura representa los permisos de examen.txt:



El usuario propietario (pepe) podrá leer y escribir en el documento. Los pertenecientes al grupo profesores podrán leerlo y el resto no podrá hacer nada.

Si deseo que otros usuarios tengan algún permiso sobre el archivo 'examen.txt', no me quedará más remedio que incluirlos en el grupo profesores u otorgar el permiso al resto de usuarios pero si hago esto último, absolutamente todos los usuarios del sistema gozarán del permiso, por eso no se recomienda salvo que eso sea nuestra intención.

Para poder cambiar permisos sobre un archivo, es necesario poseer el permiso de escritura sobre el mismo. El usuario root puede modificar los permisos de cualquier archivo ya que tiene acceso total sin restricciones a la administración del sistema.

Visualizar los permisos de un archivo o carpeta

Con el comando **ls -l** podemos visualizar los permisos de los archivos o carpetas. Al ejecutar el comando aparecen todos los archivos, uno por línea. El bloque de 10 caracteres del principio simboliza el tipo de archivo y los permisos.

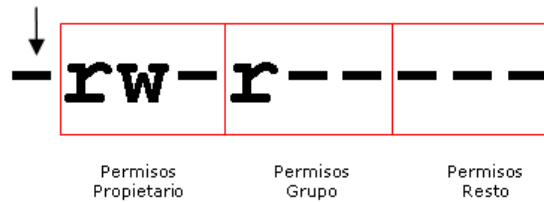
```
Bash
pepe@pepe$ ls -l
-rw-r----- 1 pepe  profes  11 Feb 12 18:57 examen-fisica.txt
-rw-r----- 1 pepe  profes  11 Feb 12 18:57 examen-musica.txt
pepe@pepe$
```

El primer carácter indica de qué **tipo de archivo** se trata. Si es un guión '-' significa que se trata de un **archivo normal**, la letra 'd' significa que se trata de una **carpeta** (directory), la letra 'l' significa que se trata de un **enlace** (link). Otros valores son s, p, b que se refieren a sockets, tuberías (pipe) y dispositivos de bloque respectivamente.

Los 9 caracteres siguientes simbolizan los **permisos del usuario propietario** (3 caracteres), los **permisos del grupo propietario** (3 caracteres) y los **permisos del resto de usuarios** (3 caracteres). Vienen codificados con las letras r, w y x que se refieren a los permisos de lectura, escritura y ejecución. Si en lugar de aparecer dichas letras aparecen guiones significa que se carece de dicho permiso. Ejemplo, si los diez primeros caracteres son -rw-r----- significa que es un archivo normal, que el usuario propietario dispone de permisos de lectura y escritura pero no de ejecución, que el grupo propietario dispone tan solo de permiso de lectura y el resto de usuarios no dispone de ningún permiso. Veámoslo en la siguiente imagen:

Tipo de archivo:

(-) para archivos normales
(d) para carpetas (directory)
(l) para enlaces (link)
(s)=socket, (p)=tubería (pipe), (b)=dispositivo de bloque.



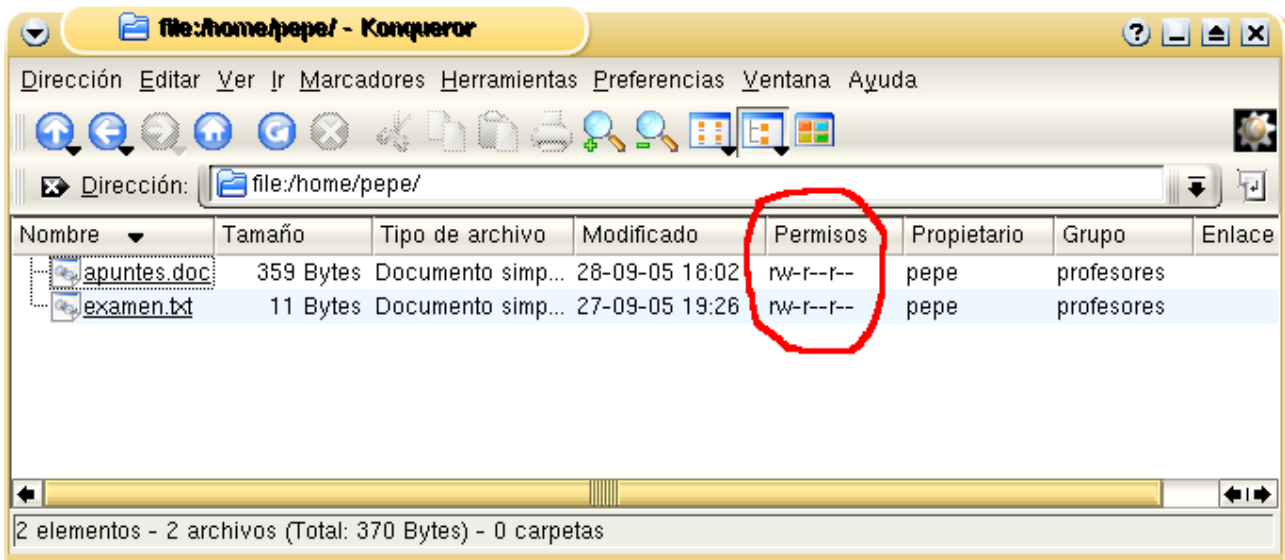
Permisos de lectura y escritura para el propietario y lectura para el grupo

En el siguiente ejemplo vemos que pepe tiene permiso de lectura y escritura y que el resto solo tiene permiso de lectura tanto sobre el archivo 'apuntes.doc' como sobre el archivo 'examen.txt'.

// Visualización de permisos

```
root@knoppix37:/home/pepe# ls -l
total 8
-rw-r--r--  1 pepe profesores 359 2005-09-28 18:02 apuntes.doc
-rw-r--r--  1 pepe profesores  11 2005-09-27 19:26 examen.txt
```

La misma información podemos verla desde el administrador de archivos 'konqueror' si vamos a la carpeta /home/pepe y activamos la opción 'vista en árbol' yendo a **Ver > Modo de vista**:



Permisos de archivo desde konqueror

Cambio de permisos

Para cambiar los permisos de un archivo o una carpeta es necesario disponer del permiso de escritura (w) sobre dicho archivo o carpeta. Para hacerlo, se utiliza el comando **chmod**. La sintaxis del comando chmod es la siguiente:

```
#chmod [opciones] permiso nombre_archivo_o_carpeta
```

Los permisos se pueden representar de dos formas. La primera es mediante las iniciales de a quién va dirigido el permiso (**usuario=u**, **grupo=g**, **resto=o** (other)), seguido de un signo **+** si se quiere añadir permiso o un signo **-** si se quiere quitar y seguido del tipo de permiso (lectura=r, escritura=w y ejecución=x).

Ejemplos:

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

```
# chmod u+w examen.txt
```

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

```
# chmod o-w examen.txt
```

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

```
# chmod g+x /usr/bin/games/tetris
```

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

```
# chmod g+r examen.txt
```

```
// Se pueden poner varios permisos juntos separados por comas
```

```
# chmod u+w,g-r,o-r examen.txt
```

```
// Se pueden poner varios usuarios juntos
```

```
# chmod ug+w examen.txt
```

La segunda forma de representar los permisos es mediante un código numérico cuya transformación al binario representaría la activación o desactivación de los permisos. El código numérico está compuesto por tres cifras entre 0 y 7. La primera de ellas representaría los permisos del usuario propietario, la segunda los del grupo propietario y la tercera los del resto de usuarios.

En binario, las combinaciones representan el tipo de permisos. El bit más a la derecha (menos significativo) se refiere al permiso de ejecución (1=activar y 0=desactivar). El bit central se refiere al permiso de escritura y el bit más a la izquierda se refiere al permiso de lectura. La siguiente tabla muestra las 8 combinaciones posibles:

Código	Binario	Permisos efectivos
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

Si deseamos otorgar sólo permiso de lectura, el código a utilizar es el 4. Si deseamos otorgar sólo permiso de lectura y ejecución, el código es el 5. Si deseamos otorgar sólo permiso de lectura y escritura, el código es el 6. Si deseamos otorgar todos los permisos, el código es el 7. Si deseamos quitar todos los permisos, el

código es el 0. Ejemplos:

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

```
# chmod 700 examen.txt
```

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

```
# chmod 550 examen.txt
```

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

```
# chmod 755 /usr/bin/games/tetris
```

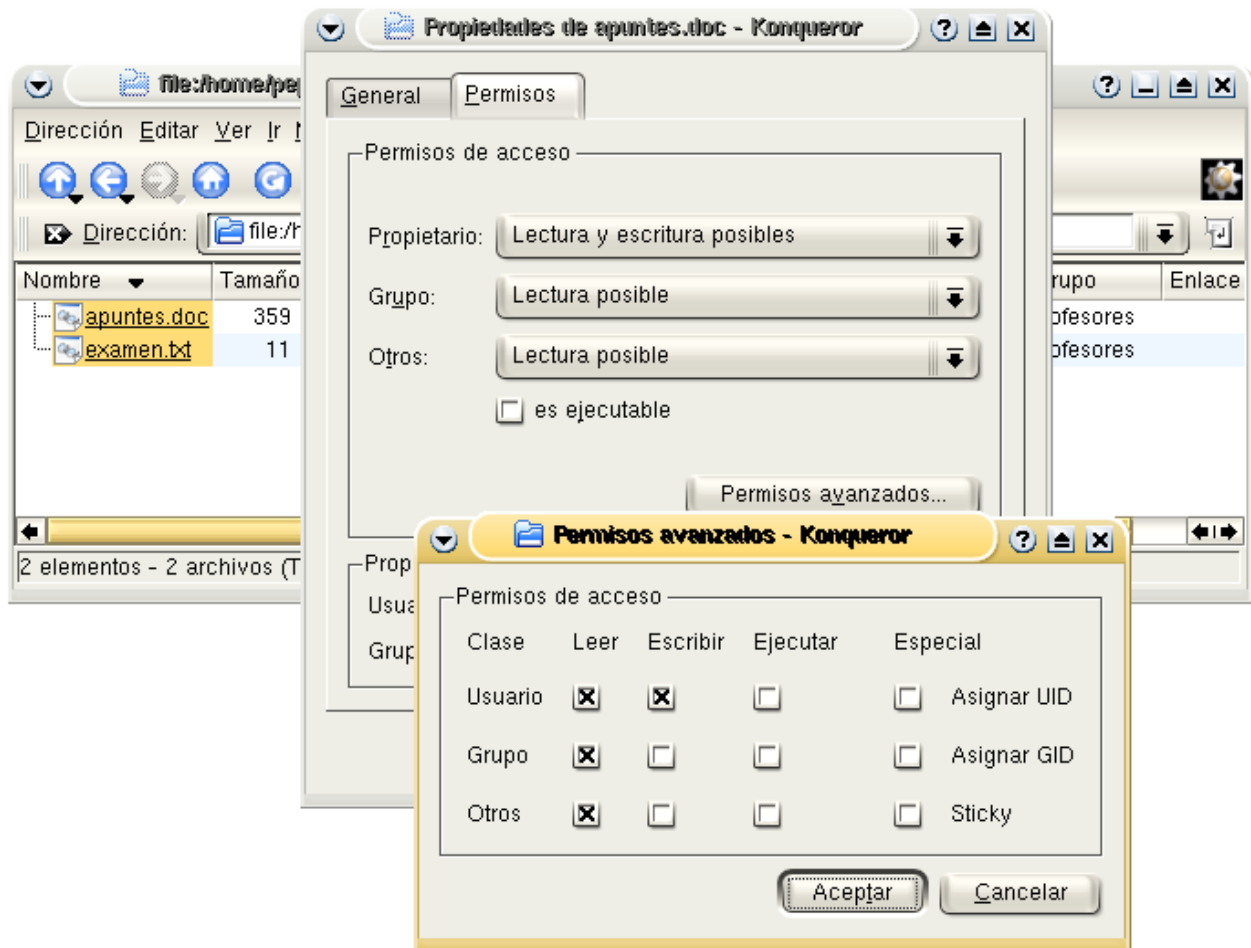
```
// Dar todos los permisos al usuario y de lectura al resto, sobre todos los  
archivos
```

```
# chmod 744 *
```

```
// Cambiar permisos a todos los archivos incluyendo subcarpetas
```

```
# chmod -R 744 *
```

Existe la posibilidad de cambiar los permisos utilizando el explorador de archivos **Konqueror**. Para ello tan solo hay que seleccionar los archivos o carpetas y haciendo clic sobre la selección con el botón derecho del ratón > **Propiedades**, nos aparecerá la ventana de propiedades. Haciendo clic en la pestaña **Permisos** podremos establecer los permisos de una forma sencilla y haciendo clic en 'Permisos avanzados' de una forma avanzada.



Estableciendo permisos desde Konqueror

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 creamos una carpeta, tendrá permisos `rw-r--r--` y cuando creamos 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`

```
pepe@3[pruebas]$ umask
0002
pepe@3[pruebas]$ mkdir nueva-carpeta
pepe@3[pruebas]$ ls -l
drwxrwxr-x    2 pepe      profes      1024 Feb 12 19:46 nueva-carpeta
pepe@3[pruebas]$ umask 022
pepe@3[pruebas]$ mkdir otra-carpeta
pepe@3[pruebas]$ ls -l
drwxrwxr-x    2 pepe      profes      1024 Feb 12 19:46 nueva-carpeta
drwxr-xr-x    2 pepe      profes      1024 Feb 12 19:46 otra-carpeta
pepe@3[pruebas]$
```

La modificación con `umask` de la máscara por defecto no afecta a los archivos y carpetas existentes si no **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 pepe**, crearemos también un grupo llamado `pepe` y haremos que el grupo primario del usuario `pepe` sea el **grupo pepe**.

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

// Ejemplo: Usuario `pepe` y grupo `pepe`

```
pepe@3[pruebas]$ more /etc/passwd |grep pepe
pepe:x:1002:1003::/home/pepe:
pepe@3[pruebas]$ more /etc/group |grep pepe
pepe:x:1003:
pepe@3[pruebas]$ ls > archivo.txt
pepe@3[pruebas]$ ls -l
-rw-rw-r--    1 pepe      pepe          12 Feb 12 20:17 archivo.txt
pepe@3[pruebas]$
```

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 pepe 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@knoppix37:/home/pepe# ls -l
total 4
-rw-rw-r-- 1 pepe profesores 11 2005-09-28 20:15 examen.txt
root@knoppix37:/home/pepe# chown luis examen.txt
root@knoppix37:/home/pepe# ls -l
total 4
-rw-rw-r-- 1 luis profesores 11 2005-09-28 20:15 examen.txt
root@knoppix37:/home/pepe# chown pedro.alumnos examen.txt
root@knoppix37:/home/pepe# ls -l
total 4
-rw-rw-r-- 1 pedro alumnos 11 2005-09-28 20:15 examen.txt
root@knoppix37:/home/pepe#
```