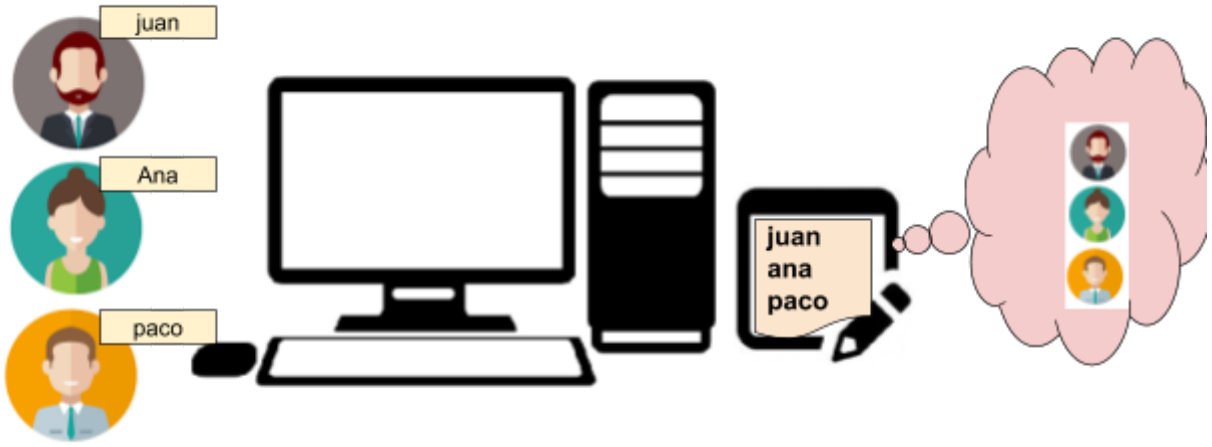
USUARIOS Y GRUPOS

Introducción

Linux es un sistema operativo multiusuario y multitarea, esto es: más de un usuario puede trabajar en el sistema de forma simultánea con otros, ejecutando una o más tareas a la vez.



Linux distingue entre diferentes usuarios, los registra y mantiene una supervisión sobre lo que intentan realizar en todo momento. Existe una especie de "base de datos" de usuarios.

> Ejercicio: Usa el comando "getent passwd" para ver la lista de usuarios registrados en tu equipo. Si observas usuarios "extraños" aguarda tu curiosidad hasta el punto "usuarios de sistema" un poco más adelante.

Una función básica de los sistemas Linux es acreditar a los usuarios que tienen acceso al computador. Es decir, verificar que el usuario que está intentando usar el ordenador es alguien registrado en esa "base de datos". La acreditación suele hacerse cuando el usuario desea iniciar una sesión con el ordenador. Si no se puede acreditar, no podrá usar ese ordenador.

> Acreditarse es lo que el lobo consiguió hacer después de dos intentos fallidos, entrando a la casa de los cabritillos y dándose un buen festín porque mamá cabrita no hubo implantado un mejor sistema de acreditación.

Pero esta gestión de los usuarios, va mucho más allá de la acreditación. El usuario se convierte en un dato que se registra en otros lugares del sistema, por ejemplo:

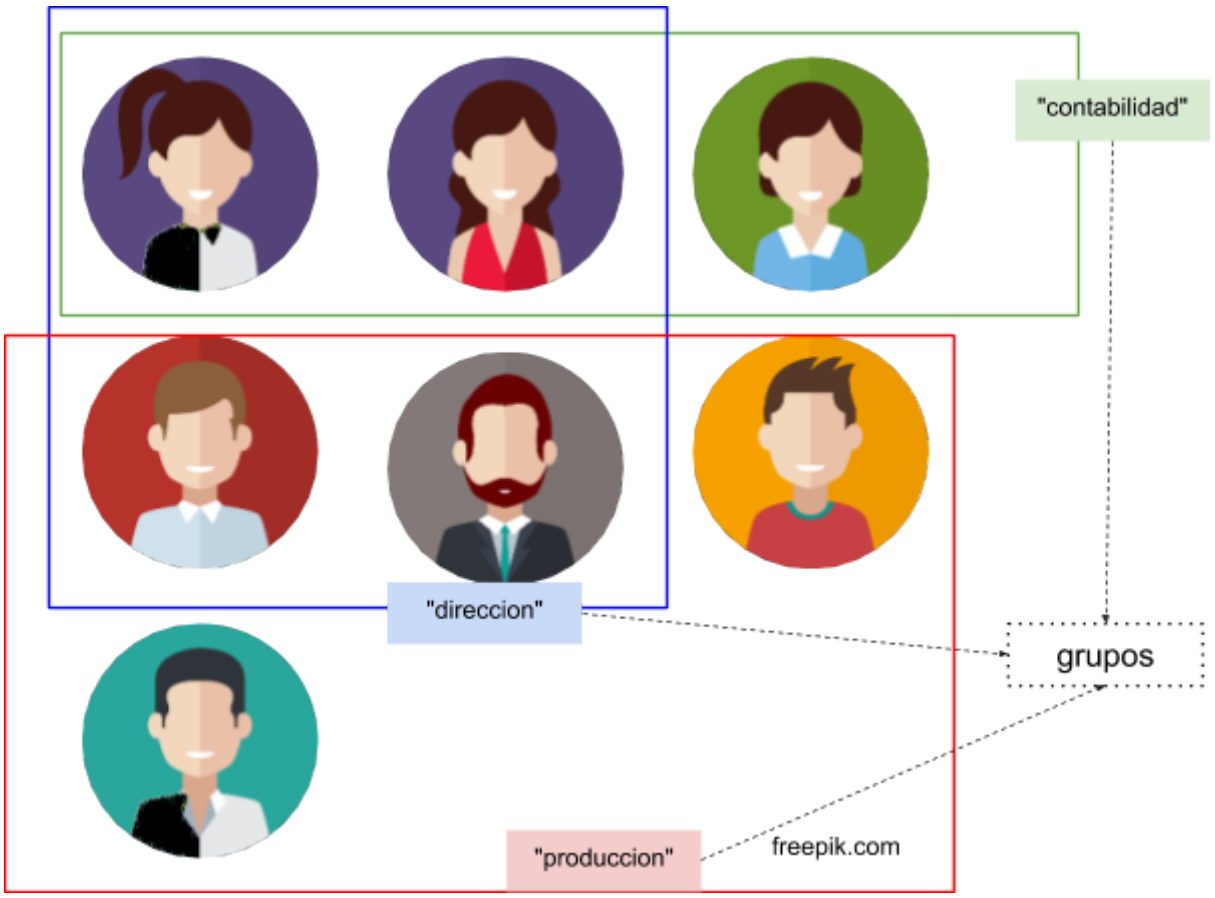
- **Ficheros**. Cada fichero pertenece a un usuario que figura como un atributo más del fichero.
- **Procesos**. Todo proceso en ejecución pertenece a un usuario que es el que lo ha lanzado a ejecución

Ejercicio: Comprobar qué dueños tienen diferentes ficheros del sistema. Ejercicio: lanza un proceso en background y averiguar quien es el dueño del proceso.

Es un poco pronto para entender la relación entre usuarios y permisos todavía.

Grupos.

Existen varios motivos por los que se hace necesario agrupar diferentes usuarios y administrar sus permisos de forma global. La solución pasa por agruparlos bajo un "grupo". Cada usuario puede estar en diferentes grupos dentro de un sistema Linux.

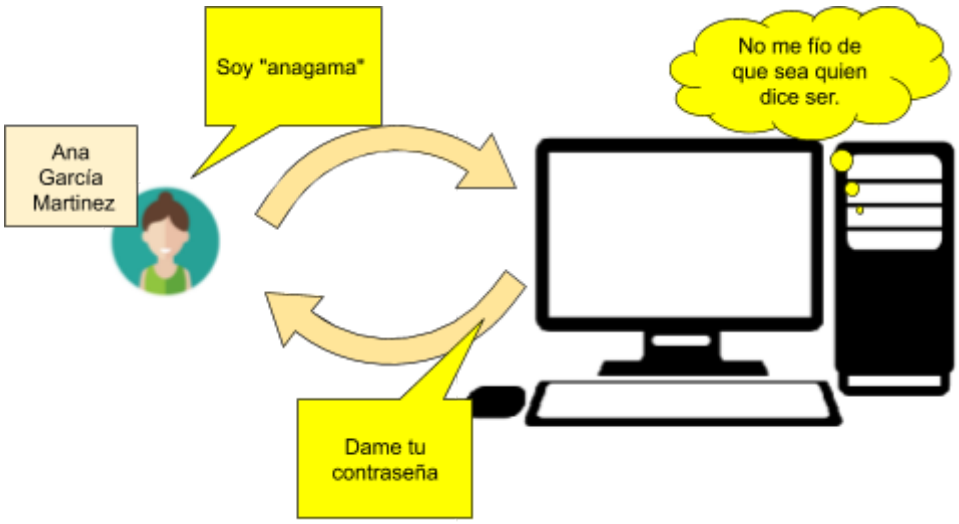


Los grupos no son más que conjuntos de usuarios.

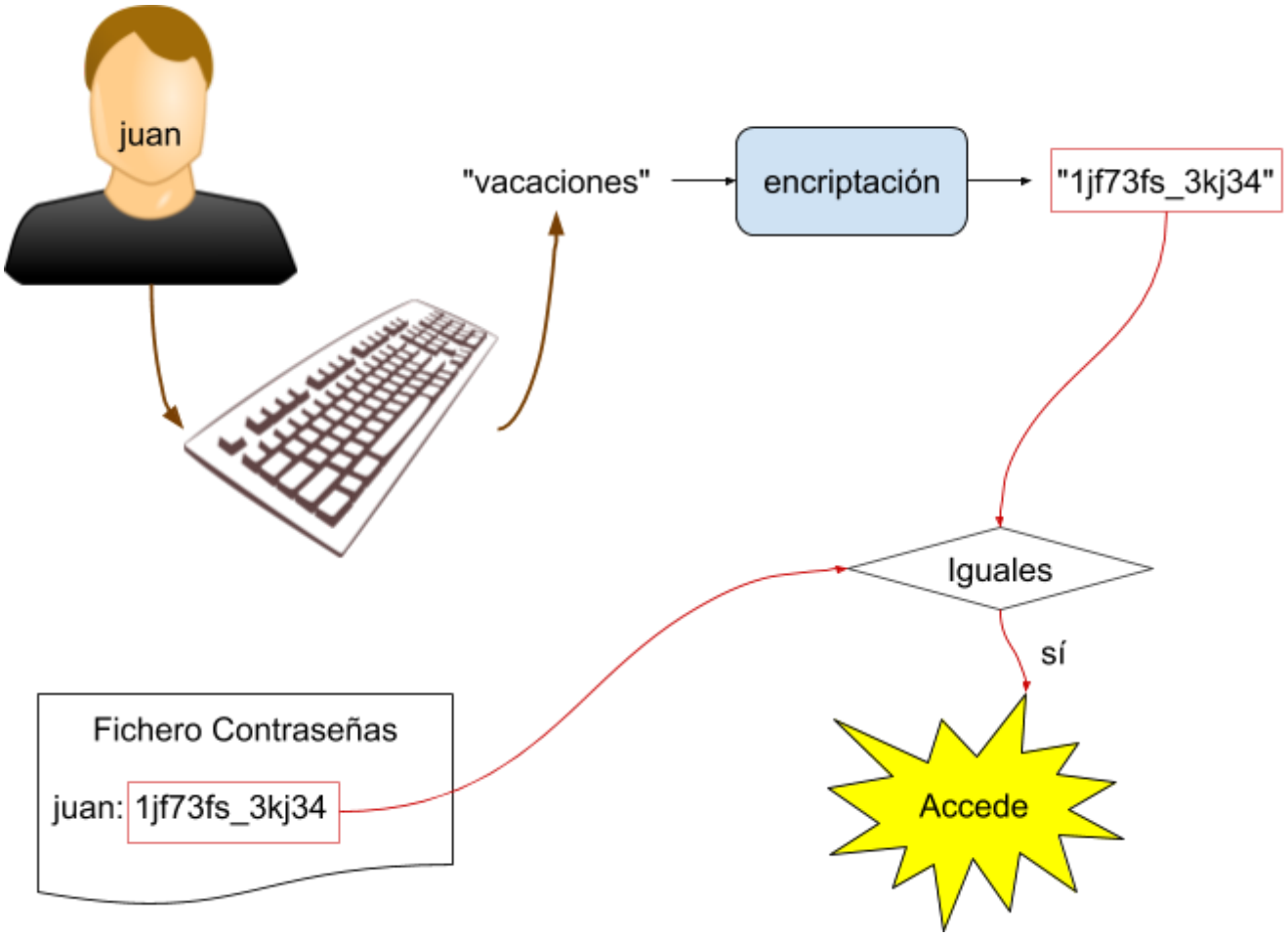
Ejercicio: Usa el comando "getent group" para ver la lista de grupos existentes en tu sistema.

Acreditación y Contraseñas

Cuando se ingresa al sistema, es necesario que un usuario se identifique con un login y una contraseña. El login es un nombre que identifica de forma única al usuario. La contraseña puede ser, por ejemplo, una combinación de letras, números y caracteres especiales.

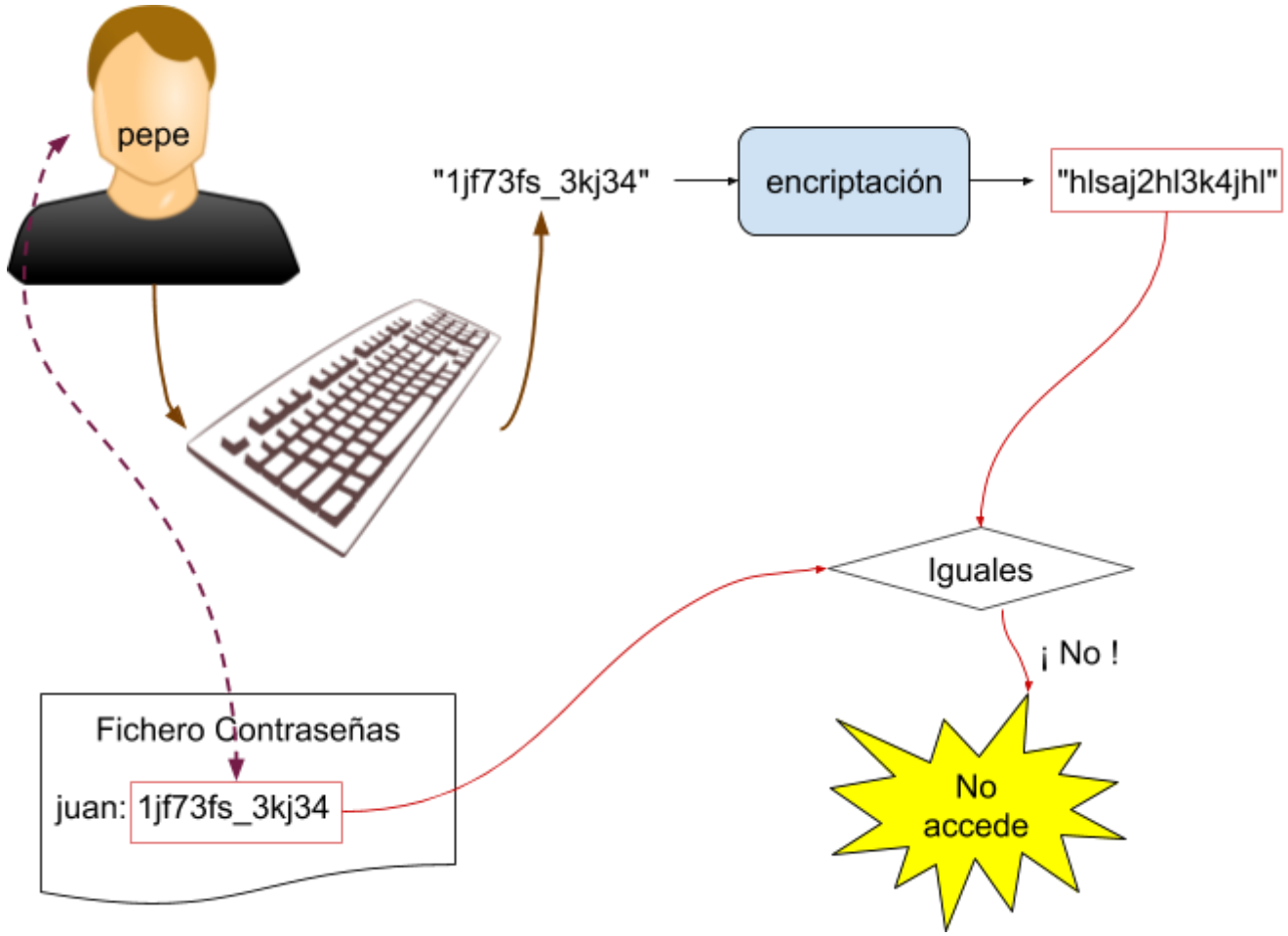


El sistema de contraseñas en Linux es de tipo unidireccional. Esto quiere decir que nuestra contraseña no es almacenada como texto, sino que es cifrada y guardada así. Cuando entramos en el sistema y escribimos nuestra contraseña, ésta es cifrada igualmente y comparada con la que está almacenada (también cifrada). Si coinciden, la identificación es positiva, si el resultado de la encriptación es distinto, no hay identificación.



Conocer la contraseña cifrada no te lleva fácilmente a descubrir la contraseña sin cifrar (pero aún así, se ocultan en la actualidad). Veamos un ejemplo:

Supón que pepe descubre y lee el fichero de contraseñas y concretamente la contraseña cifrada de juan (que es "1jf73fs_3kj34"). Cuando intente ingresar y acreditarse, el malévolo pepe escribirá esa contraseña en el teclado. Pero a continuación, ésta se encriptará (como se hace con todas) produciendo un resultado muy distinto (en el ejemplo "hlsaj2hl3k4jhl") . El sistema comparará esta encriptación con la contraseña original y la rechazará.



Actualmente, en los sistemas GNU/Linux se puede escoger dos tipos de cifrado posibles para las contraseñas de usuario: 3DES que se viene usando desde los inicios de UNIX, tiene el inconveniente que sólo permite contraseñas de 8 caracteres, si se escriben más se ignoran, el otro tipo es MD5 con el que podemos usar contraseñas de la longitud que deseamos, por seguridad se recomienda utilizar el tipo MD5.

Saber la contraseña encriptada y averiguar la contraseña real, es algo así como probar una paella y descubrir sólo por el sabor/textura los detalles de cómo ha sido cocinada. Si no ves cómo se cocina, no lo averiguarás por mucha paella que comas.

Los usuarios en GNU/Linux

Existen 3 tipos de usuarios en una instalación normal:

- **Usuario corriente**, (regular user) es un individuo particular que puede entrar en el sistema y utilizarlo de forma normal con unos privilegios limitados. También se les conoce como usuarios de login. Estos usuarios corresponden, pues, con una persona.
- **"root" (superusuario)**, todo sistema operativo GNU/Linux cuenta con un superusuario, "root" o usuario administrador, que tiene los máximos privilegios que le permitirán efectuar cualquier operación sobre el sistema, su existencia es imprescindible ya que se encarga de gestionar los servicios, resto de usuarios y grupos, etc.
- **Usuarios de Sistema**, Son realmente usuarios virtuales o que no corresponden con personas, pero necesarios por algún motivo. Vienen preinstalados con el sistema o junto con algún paquete y están vinculados a las tareas que debe realizar el sistema operativo, este tipo de usuario no puede ingresar al sistema con un login normal. Ejemplo: mail, ftp, bin, sys, proxy, etc. También se le conoce como usuarios sin login

Ejercicio: Usa el comando "getent passwd" para ver la lista de usuarios registrados en tu equipo y diferenciar los tipos de usuario anteriores. No tomes conclusiones certeras todavía.

Base de datos de usuarios

Toda la información de los usuarios y grupos se guarda en los siguientes archivos:

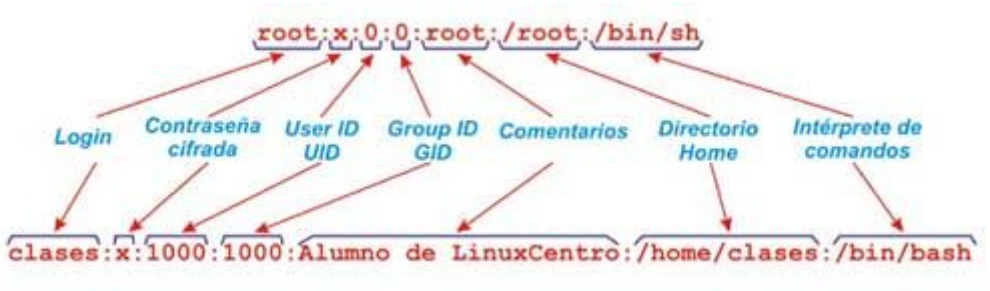
- **`/etc/passwd`**: guarda información de los usuarios del sistema como: nombres, directorio home, shell.
- **`/etc/group`**: almacena la información sobre los grupos existentes en el sistema.
- **`/etc/shadow`**: contiene las contraseñas cifradas de los usuarios además de otros datos para su validación.

Archivo /etc/passwd

El archivo `passwd` almacena los usuarios creados en el sistema y tiene el siguiente formato:

```
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
clases:x:1000:1000:LinuxCentro:/home/clases:/bin/bash
```

Cada línea está separada en campos, el separador de campo son los dos puntos (:), y cada campo representa lo siguiente:



- ***Login:*** el nombre del usuario. No puede haber dos nombres iguales.
- ***Contraseña cifrada:*** si no se utiliza el fichero de shadow, las contraseñas cifradas se almacenan en este campo. Si utilizamos el fichero de shadow, todos los usuarios existentes en este fichero deben existir también en el de shadow y en este campo se pone el carácter "x".
- ***User ID:*** número de identificación del usuario. Es el número con el cual el sistema identifica al usuario. El 0 es el único que está reservado para el root.
- ***Group ID:*** el número de grupo al cual pertenece el usuario. Como un usuario puede pertenecer a más de un grupo, este GID es del grupo primario.
- ***Comentarios:*** campo reservado para introducir los comentarios que queramos sobre el usuario. Se suele utilizar para poner el nombre completo o algún tipo de identificación personal.
- ***Directorio home:*** el directorio home del usuario es donde éste podrá guardar todos sus ficheros, generalmente se encuentran dentro del directorio /home y el nombre de cada directorio es similar al de cada usuario.
- ***Intérprete de comandos:*** un intérprete de comandos (shell) es un programa que se encarga de leer todo lo que escribimos en el teclado y ejecutar los programas o comandos que le indiquemos. Hay decenas de ellos, aunque el más utilizado es, sin duda, el bash (GNU Bourne-Again SHell). Si en este campo está: /bin/false ó /bin/nologin el usuario no podrá tener acceso a su shell y no podrá ejecutar comandos.

Actividad. Observa el fichero /etc/passwd intentando descubrir las características y diferencias de los usuarios normales, administrador (root) y de sistema.

Los grupos en Linux

Los grupos se utilizan para agrupar diferentes usuarios y establecer permisos o privilegios sobre todos ellos a la vez simplemente administrando el grupo.

- Los grupos tienen un nombre y un número que los identifica.

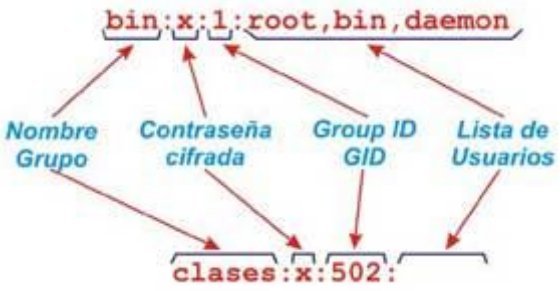
- Algunos de los elementos del sistema (ficheros, procesos, etc) están caracterizados por un usuario "dueño" y también por un grupo al que se asigna dicho elemento
- Los grupos pueden tener un administrador, que es un usuario con privilegios para añadir o eliminar usuarios del grupo administrado
- Solo root puede establecer un administrador en un grupo
- Los grupos pueden tener una contraseña, que se usa para que un usuario que no es del grupo, pueda pasar por ser de este grupo durante una sesión.
- …..FALTA

Archivo /etc/group

El archivo group almacena la información de los grupos del sistema, y tiene el siguiente formato:

```
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
clases:x:502:
```

Al igual que el archivo anterior cada línea está separada en campos el separador de campo son los dos puntos (:), y cada campo representa lo siguiente:



- **Nombre del grupo**: Por defecto con los comandos habituales se crea un grupo con el mismo nombre que el usuario creado, aunque pueden existir otros grupos con nombres específicos.
- **Contraseña cifrada**: la contraseña de un grupo se utiliza para permitir que los usuarios de un determinado grupo se puedan cambiar a otro o para ejecutar algunos programas con permisos de otro grupo, siempre que se disponga de la contraseña.
- **Group ID**: número de identificación del grupo. Es el número con el cual el sistema identifica internamente a los grupos. El 0 es el único que está reservado para el grupo del root (los administradores).
- **Lista de usuarios**: los nombres de los usuarios que pertenecen al grupo, separados por comas. Aunque todos los usuarios deben pertenecer a un determinado grupo (especificado en el cuarto campo del fichero de passwd), este campo se puede

utilizar para que usuarios de otros grupos también dispongan de los mismos permisos que tiene el que se está referenciando.

Archivo /etc/shadow

El archivo shadow se encarga de almacenar las contraseñas cifradas del usuario, y tienen el siguiente formato:

```
root:$1$qvZCDFha$8CsNHHB/QDYlx3wDnZWzp/:12829:0:99999:7:::
bin:*:12829:0:99999:7:::
daemon:*:12829:0:99999:7:::
clases:$1$8Ne4Ij4r$th9obKXkR7iTZGj26jGUc/:12831:0:99999:7:::
```

Igual que los archivos anteriores cada línea está separada en campos el separador de campo son los dos puntos (:), y cada campo representa lo siguiente:

- **Login**: debe ser el mismo nombre que se utiliza en el fichero de passwd.
- **Contraseña cifrada**.
- Días que han pasado, desde el 1 de enero de 1970, hasta que la contraseña ha sido cambiada por última vez.
- Días que deben pasar hasta que la contraseña pueda ser cambiada.
- Días que han de pasar hasta que la contraseña deba ser cambiada.
- Días antes de caducar la contraseña en el que se avisará al usuario de que debe cambiarla.
- Días que pueden pasar después de que la contraseña caduque, antes de deshabilitar la cuenta del usuario (si no se cambia la contraseña).
- Días, desde el 1 de enero de 1970, desde que la cuenta está deshabilitada.
- Campo reservado.

En sistemas UNIX es muy común representar las fechas a partir del número de segundos transcurridos desde el 1 de enero de 1970.

En sistemas donde hay muchos usuarios y se desea restringir el espacio de disco que utilizan se puede utilizar quotas.

Actividad. Averigua en internet, cuál es la historia de este archivo y comprueba sus permisos.

Directorio /etc/skel

Cada usuario, al ser creado, debe tener un directorio de inicio, y en él, suelen haber ficheros de inicialización y configuración. Este directorio contiene los ficheros que se copian a un directorio e inicio en el momento de la creación de un nuevo usuario.

Archivo /etc/gshadow

El archivo gshadow guarda configuracion de administración de los grupos. Incluyendo el usuario responsable de administrar el grupo y la contraseña necesaria para pasar a pertenecer a este grupo durante una sesión. Info: Un usuario no puede por sí mismo añadirse a un grupo, incluso si sabe esta contraseña, pero durante una sesión, sí que puede pasar a pertenecer a este grupo (provisionalmente) si sabe la contraseña. Este es el principal uso de la contraseña de grupo.

COMANDOS PARA GESTIONAR USUARIOS

A partir de aquí todo se realiza en una actividad práctica que está en el anexo.

Para crear usuarios existen dos comandos. Uno está enfocado a facilitar la creación de un usuario mediante la introducción interactiva de sus datos y la creación automatizada de todos los elementos necesarios (directorio de inicio, grupos, etc) . El otro está orientado a ser usado desde un script y realiza sólo funciones básicas

Creación de usuarios: Adduser y useradd

Cuando se da de alta un nuevo usuario hay diversos elementos posibles que modificar/añadir/configurar (/etc/passwd, directorio home, intérprete de comandos, etc). La acción de crear el usuario debe establecer estos parámetros, carpetas, etc. Pese a que todo puede hacerse manualmente por el administrador, existen comandos que realizan esta tarea completamente

adduser : Este comando pregunta interactivamente los datos del nuevo usuario y crea los ficheros y directorios necesarios. Observa la siguiente captura de pantalla que muestra el momento en el que realiza la primera pregunta

```
2DAMASIX / # adduser pepe
Adding user `pepe' ...
Adding new group `pepe' (1003) ...
Adding new user `pepe' (1003) with group `pepe' ...
Creating home directory `/home/pepe' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
```

¿Ves el orden en el que se crean las cosas ?

useradd : permite añadir nuevos usuarios al sistema, también permite establecer la información por defecto de los nuevos usuarios. Este comando obtiene la información de los usuarios por línea de argumentos.

En algunas distribuciones se encuentra enlazado simbólicamente por el comando **adduser**, ambos nombres se pueden emplear indistintamente para las mismas acciones. Sin embargo, en Ubuntu, son comandos diferentes y useradd tan sólo modifica `/etc/passwd`, `/etc/group` y `/etc/shadow`, pero no crea directorios de inicio, copia ficheros de /etc/skel ni obliga a crear una contraseña. Este comando es útil para crear usuarios de sistema

Algunas opciones:

-u : permite especificar el UID.
-c : añade los valores a la sección de comentarios.
-d : permite especificar el directorio de trabajo, no creará automáticamente el directorio señalado, se supone que existe, pero si no existe se crea igual el usuario.
-s : permite establecer el shell.

Ejemplos:

`# useradd raul`	# crea el usuario raul con las propiedades por defecto
`# useradd -u 500 carlos`	# crea el usuario carlos con su UID 500
`# useradd -c 'Juan Perez' juan`	#crea el usuario juan rellenando el comentario con "Juan Perez"
`# useradd -d /home/soft msantos`	# crea el usuario msantos con su directorio de trabajo "soft"
`# useradd -s /bin/false operador1`	# crea el usuario operador1 desactivando la posibilidad de ejecutar un shell

`# useradd -D`
 	
`GROUP=100`
`HOME=/home`
`INACTIVE=-1`
	# muestra las propiedades por defecto de los nuevos usuarios que se añadan
	# grupo primario (no se emplea)

`EXPIRE=`
`SHELL=/bin/bash`
`SKEL=/etc/skel`
	# directorio base de todos los usuarios
	# número de días entre que el password expire y la cuenta se deshabilite
	# fecha (YYYY-MM-DD) en que la cuenta expira
	# shell que empleará el usuario para interactuar con el sistema
	# directorio cuyo contenido se copiará en el directorio de cada usuario

`# useradd -D -b /users`	# cambia el directorio base por defecto de los nuevos usuarios a /users. El directorio /users debe existir previamente

Valores por defecto: son valores por defecto que serán utilizados para crear a los nuevos usuarios:

En RedHat o derivados, los valores por defecto están en el archivo /etc/default/useradd.

En Debian y derivados, los valores por defecto para añadir nuevos usuarios estan en el archivo /etc/adduser.conf

Ejemplo

```
useradd –m –b "/home/pepe" –d "/home/pepe" –u 100001 –s /bin/bash –U pepe
```

Ejercicio: Revisa en la descripción anterior de los comandos adduser y useradd, qué modificaciones se realizan en el sistema al crear un usuario. Constátalo verificando el sistema antes y después de que crees un usuario.

Deberás saber qué cambios provoca la creación correcta y completa de un usuario.

Comando passwd

`Comando passwd` : permite establecer y/o cambiar la contraseña de un usuario. También puede bloquear, desbloquear y deshabilitar una cuenta. Si se invoca sin argumentos se asume que se está cambiando la contraseña del usuario actual.

Algunas Opciones:

-d : deshabilita la necesidad de contraseña del usuario.
-l : bloquea la cuenta de un usuario añadiendo un signo de admiración (!) delante de su contraseña en el archivo /etc/shadow.
-u : desbloquea la cuenta de un usuario bloqueado.

Ejemplos:

`# passwd raul`
`New UNIX password:`
`Retype new UNIX password:`
`passwd: all authentication tokens updated successfully`
	# estableciendo una contraseña para raul

# passwd -d raul	# deshabilita la cuenta del usuario raul eliminando su password
# passwd -l raul	# bloquea la cuenta del usuario raul poniendo un signo ! delante de su password en el archivo /etc/shadow
# passwd -u raul	# desbloquea la cuenta del usuario raul

Recuerda: Los parámetros asociados a la contraseña del usuario así como su situación de bloqueo están registrados en el fichero /etc/shadow

Comandos userdel, deluser y usermod

- **Comando deluser** : permite eliminar definitivamente un usuario del sistema.
- **Comando userdel** : permite eliminar definitivamente un usuario del sistema.

Ejemplos:

`# userdel raul`	# elimina el usuario raul manteniendo su directorio de datos.
`# userdel -r raul`	# elimina al usuario raul y borra su directorio base.

- **Comando usermod** : se emplea para modificar algunas propiedades de los usuarios como: el login, el directorio base, el shell que se inicia al conectarse, los grupos a los que pertenece, la fecha de expiración de la cuenta, etc. También bloquea y

desbloquea una cuenta. Como opciones utiliza las disponibles en el comando useradd.

Ejemplos:

`# usermod -s /bin/csh pepe`	# coloca el shell csh para el usuario pepe
`# usermod -G users,disk pepe`	# señala como grupos secundarios de pepe a users y disk
`# usermod -e 2005-10-20 pepe`	# indica que la cuenta de pepe expirará el 20 de octubre del 2005

Comandos chfn, chage y chsh

Comando chfn : permite cambiar la información de contacto de un usuario. Esta incluye aspectos como: el nombre completo, la oficina de trabajo y los teléfonos. Se almacena en el archivo de usuarios /etc/passwd en la sección de comentarios.

Ejemplos:

# chfn pepe Changing finger information for pepe. Name []: pepe Davis Office []: Informática Office Phone []: 22-0909 Home Phone []: 44-3025	# Cambia información del usuario pepe
# chfn Changing finger information for root. Name []: Office []: Office Phone []: Home Phone []:	# cambia la información de contacto del usuario actua

Comando chsh: cambia el shell del usuario especificado.

Ejemplos:

```
# chsh pepe
```

Cambiando la shell de acceso para pepe
Introduzca el nuevo valor, o presione enter para el predeterminado
Shell de acceso [/bin/bash]:

```
# chsh
```

Cambiando la shell de acceso para root
Introduzca el nuevo valor, o presione enter para el predeterminado
Shell de acceso [/bin/bash]:

Comando chage : permite cambiar el password y los datos del usuario.

Algunas Opciones:
-d [dias] : Cuenta el número de días (desde 01-01-1970) transcurridos desde que cambió la contraseña por última vez
-E [fecha] : Modifica la fecha en que la cuenta del usuario expirará y será bloqueada
-I [dias] : Modifica cuantos días puede permanecer una cuenta con una contraseña expirada antes de ser bloqueada
-M [dias] : Modifica el número máximo de días durante los que es válida la contraseña de usuario. Pasados los días, el usuario deberá de modificarla
-m [dias] : Modifica el número mínimo de días entre cambio de contraseña
-W [dias] : Modifica el número de días que se avisará al usuario antes de cambiar la contraseña

Ejemplos:

`# chage -E 2005-06-15 pepe`	# la cuenta del usuario pepe expirará el 15.Junio.2005
`# chage -I 7 pepe`	# la cuenta del usuario pepe tendrá 7 días de comunicaciones antes de ser bloqueada
`# chage -M 7 pepe`	# se da 7 dias al usuario pepe para que pueda modificarla su contraseña, luego del cual deberá modificarla en forma obligatoria

COMANDOS PARA GESTIONAR GRUPOS

- **Comando groupadd** : permite añadir un grupo al sistema.

Ejemplos:

`# groupadd admin`	
`# groupadd -g 601 supervisor`	# añade un grupo supervisor con GID 601

- **Comando groupdel** : permite eliminar un grupo del sistema, el grupo no podrá ser eliminado si este es el grupo primario de un usuario.

  Ejemplo:
  # groupdel admin
- **Comando groupmod** : permite modificar el nombre o GID de un grupo.

  Ejemplos:

`# groupmod -g 701 supervisor`	# cambia el GID a 601 del grupo supervisor
`# groupmod -n manager supervisor`	# cambia el nombre del grupo supervisor a manager

- **Comando gpasswd** : permite administrar los grupos. Se puede utilizar para añadir y eliminar usuarios, señalar un administrador e indicar un password para el grupo.

  Ejemplos:

`# gpasswd -A raul admin`	# señala como administrador del grupo admin al usuario raul
`# gpasswd admin`	# cambia el passwd del grupo admin
`# gpasswd -a juan admin`	# añade el usuario juan al grupo admin
`# gpasswd -d juan admin`	# elimina al usuario juan del grupo admin

El comando adduser/deluser también se puede usar para añadir/eliminar un usuario a un grupo

- Ejemplos:

`# adduser raul contables`	# añade a raul al grupo de contables
`# deluser raul contables`	# quita a raul del grupo de contables.

COMANDOS ADICIONALES

Comando	**Descripción**

# whoami	# nos muestra que usuario somos
# groups [usuario]	# nos sirve para saber a qué grupos pertenecemos
# id [usuario]	# nos mostrará nuestro uid, gid, yid el id del grupo al que pertenecemos
# su [usuario]	# sirve para convertirnos en otro usuario sin tener que salir de la sesión, así como para cambiar a ser un superusuario
# newgrp [grupo]	# sirve para cambiar de grupo principal de un usuario. Puede incluso utilizarse cambiar a un grupo que no pertenecemos, para ello entramos en una nueva sesión en la que todo sigue igual menos el grupo primario.
# who	# nos muestra la lista de usuarios dentro del sistema
# w [usuario]	# nos muestra la lista de usuarios dentro del sistema y también lo que están haciendo
# write [usuario]	# comando que se utiliza para comunicarse entre los usuarios del sistema
# wall [mensaje]	# permite enviar un mensaje a todos los terminales de los usuarios dentro sistema
# mesg [y|n]	#permite activar o desactivar la opción de recibir mensajes

Cambios de grupo.

Observa que un usuario puede pertenecer a varios grupos a la vez. En muchas ocasiones esto le otorga un privilegio especial, que es la suma de todos los privilegios de los distintos grupos. Sin embargo, en otras acciones, el usuario ejecuta acciones para las cuales debe establecer o participar con un sólo grupo, el resto de grupos no se tienen en cuenta.

Por ejemplo, cuando el usuario cree un fichero, éste tendrá como dueño al usuario que lo ha creado (¡obvio!) Pero el fichero también tendrá un grupo. La pregunta es: ¿Si un usuario pertenece a muchos grupos cuál será el grupo del fichero que cree este usuario?

De todos los grupos a los que pertence el usuario, hay uno que llamamos primario, éste es el que se hace valer para casos como los de crear un fichero. Y por ello, al crear un usuario nuevo en el sistema se crea y se le asigna un grupo, para cubrir esta necesidad.

Pero un usuario puede cambiar su grupo primario, tanto permanentemente como en medio de una sesión. Para esto último se utiliza el comando newgrp. Así, el usuario puede seleccionar qué grupo será el primario.

Explica qué pasa aquí:

```
maria@2DAMASIX:~ > id
uid=1004(maria) gid=1004(maria) groups=1004(maria)
maria@2DAMASIX:~ > newgrp contables
Password:
maria@2DAMASIX:~ > id
uid=1004(maria) gid=1006(contables) groups=1004(maria),1006(contables)
maria@2DAMASIX:~ >
```

<falta poner :_La siguiente captura de pantalla explica el uso de newgrp:>

Permisos sobre ficheros.

Todos los archivos y directorios en Linux llevan una información sobre permisos que establecen quién puede hacer o no alguna acción con él. Cada fichero tiene unos permisos para:

- El **dueño** del fichero,
- Los usuarios del **grupo** (del fichero). ¡ Sí, así es ! cada fichero pertenece también a un grupo.
- Los **demás usuarios** (los que no son ni el dueño ni pertencen al grupo del fichero).

Así, el dueño puede hacer algunas operaciones con un fichero (o directorio), los usuarios de su grupo tienen capacidad para realizar otras operaciones y los demás tienen otro conjunto de permisos.

¿Cuáles son los permisos?

Pero *¿Qué se puede hacer con un fichero que deba impedirse o permitirse?*

Sobre ficheros:

- **r: read** (lectura): Se puede leer su contenido
- **w: write** (escritura):Se puede alterar su contenido (no significa que se pueda eliminar el fichero).
- **x: execute** (ejecución): El fichero se puede ejecutar (es un programa o un script).

Sobre directorios:

Tenga en cuenta que los nombres de tipos de permisos se mantienen pero su significado cambia al ser aplicados sobre directorios.

- **r: read** (lectura): Se puede listar el contenido y ver los ficheros que en él residen (comando '`ls`' funciona)
- **w: write** (escritura): Se puede crear y borrar ficheros dentro de él. No es preciso tener permiso de lectura para ello
- **x: execute** (ejecución): El directorio podrá ser usado en la formación de rutas hacia directorios interiores, esto es: podrá ser atravesado

Visualización de los permisos

Para poder ver los permisos de los archivos y directorios es necesario ejecutar el siguiente comando:

```
$ ls -l
```

Este comando nos dará una salida similar a la siguiente:

```
drwxr-xr-x 3 raul raul 4096   2005-02-16   14:47 Desktop
drwxr-xr-x 5 raul raul 4096   2005-02-16   12:42 GNUstep
-rw-r--r-- 1 raul raul 246417 2005-03-03   13:13 foto1.png
-rw-r--r-- 1 raul raul 232505 2005-03-03   13:14 carta2.abw
-rw-r--r-- 1 raul raul 239618 2005-03-03   13:14 informe.abw
drwxr-xr-x 2 raul raul 4096   2005-02-16   12:42 tmp
```

Ahora describamos la salida que hemos obtenido:

Con la siguiente línea interpretamos la información así:

```
- rw- r-- r-- 1 raul raul 246417 2005-03-03 13:13 foto1.png
↑ ↑  ↑  ↑  ↑ ↑ ↑    ↑      ↑         ↑          ↑     ↑
1 2  3  4  5 6 7    8      9         10         11
```

1. **Tipo de archivo** = es un archivo regular
2. **Permisos** = los permisos para el propietario son de lectura y escritura
3. **Permisos** = el grupo tiene permiso de sólo lectura
4. **Permisos** = los otros usuarios tienen el permiso de sólo lectura
5. **Enlace Físico** = tiene un enlace físico
6. **Propietario** = el usuario raul es el propietario de este archivo
7. **Grupo** = este archivo pertenece al grupo raul
8. **Tamaño** = su tamaño es de 246417 bytes
9. **Fecha** = fue creado el 03 de marzo de 2005
10. **Hora** = a 13:13 horas
11. **Nombre** = el archivo se llama "foto1.png"

Los permisos están asignados en 3 grupos de 3 caracteres:

3 grupos (dueño, grupo y otros) de tres carácteres (r, w, x)

En la siguiente imagen resaltamos nuevamente la ubicación de los permisos en caso no lo hayamos notado:

Cambiar permisos de un fichero

Mediante el comando **chmod** indicando los permisos y el nombre del fichero:

Para cambiar los permisos se puede hacer de 2 maneras:

1. Utilizando taquigrafía basada en caracteres, o
2. Utilizando números.

Cambio de permisos utilizando taquigrafía de caracteres:

Mediante una combinación de caracteres quitamos o ponemos permisos a uno o varios ficheros y su dueño, grupo u otros. En la siguiente tabla se especifica el significado de cada carácter usado en en este modo.

	Símbolo	Descripción
Identidades	**u**	Es el usuario propietario del archivo o directorio
	g	Es el grupo al que pertenece el archivo o directorio
	o	Otros usuarios, el resto del mundo, ni el propietario ni su grupo
	a	Todo el mundo – propietario, grupo y otros
Permisos	**r**	Acceso de lectura
	w	Acceso de escritura
	x	Acceso de ejecución
Acciones	**+**	Añade los permisos
	-	Elimina los permisos
	=	el único permiso. Asigna esos permisos exactamente

Vamos a practicar con el comando chmod, para ello lo primero que haremos será crear el archivo `foto1.png` para ver los cambios de permisos, así que les recomiendo seguir la secuencia siguiente:

Ejemplo	Descripción	Resultado
`touch foto1.png`	creamos el archivo foto1.png	`foto1.png`
`chmod a-rwx foto1.png`	le quitamos todos los permisos	`---------`
`chmod u+rwx foto1.png`	añadimos todos los permisos para el propietario	`rwx------`
`chmod g+x foto1.png`	añadimos el permiso de ejecución para el grupo	`rwx—x---`
`chmod o+r foto1.png`	añadimos el permiso de lectura para los otros usuarios	`rwx—xr--`
`chmod u-rw foto1.png`	eliminamos los permisos de lectura y escritura para el propietario	`--x—xr--`
`chmod a=r foto1.png`	establecemos como único permiso de lectura para los 3 grupos	`r—r—r--`
`chmod a=rx foto1.png`	establecemos los permisos de lectura y ejecución para los 3 grupos	`r-xr-xr-x`
`chmod a=- foto1.png`	quitamos todos los permisos	`----------`
`chmod u+rx,o+x foto1.png`	añadimos los permisos de lectura y ejecución al propietario y ejecución a otros	`r-x-----x`
`chmod g+rx,o-x foto1.png`	añadimos permiso de lectura y ejecución al grupo y eliminamos permiso de ejecución a otros	`r-xr-x---`
`chmod ug+wx,o-x foto1.png`	añadimos permiso de escritura y ejecución al propietarifo y grupo, y eliminamos permiso de ejecución a otros	`rwxrwx---`
`$ chmod a=rw foto1.png`	permite a cualquiera modificar el contenido e incluso eliminar el archivo	`rw-rw-rw-`

Si cambiamos los permisos a un directorio y deseamos que estos permisos tengan efecto sobre todos sus subdirectorios y archivos sólo deberemos añadir la opción `-R`. Ejemplo:

```
$ chmod a=rw DIRECTORIO -R
```

Cambio de permisos utilizando números.

Los permisos se traducen en números. Ciertos números están asignados a ciertas combinaciones de permisos, pero estas combinaciones siguen una lógica binaria sencilla. Suponemos que cada permiso individual tiene una posición y un 1 en esa posición da como resultado un número.

```
r = 4 (100) (lectura)
w = 2 (010) (escritura)
x = 1 (001) (ejecucion)
- = 0 (000) (sin permisos)
```

Cuando asignamos los permisos utilizando números debemos tener en cuenta que primero se sumarán los valores y dicho resultado será el que se coloque, aquí una tabla que muestra dichos valores:

Valor	Binario	Permisos	Descripción
0	000	`---`	El valor cero significa que no se han asignado permisos
1	001	`--x`	sólo se ha asignado el de ejecución
2	010	`-w-`	sólo permiso de escritura
3	011	`-wx`	permisos de escritura y ejecución
4	100	`r--`	sólo permiso de lectura
5	101	`r-x`	permisos de lectura y ejecución
6	110	`rw-`	permisos de lectura y escritura
7	111	`rwx`	permisos: lectura, escritura y ejecución

Los permisos por números se asignan en grupos de 3, es decir, para el propietario-grupo-otros, no es factible asignar solo para uno o dos de ellos.
Ejemplos:

- **`rw------- (600)`** – Sólo el propietario tiene el derecho de leer y escribir.
- **`rw-r--r-- (644)`** – Sólo el propietario tiene los permisos de leer y escribir; el grupo y los demás sólo pueden leer.
- **`rwx------ (700)`** – Sólo el propietario tiene los derechos de leer, escribir y ejecutar el archivo.
- **`rwxr-xr-x (755)`** – El propietario tiene los derechos de leer, escribir y ejecutar; el grupo y los demás sólo pueden leer y ejecutar.
- **`rwx--x--x (711)`** – El propietario tiene los derechos de lectura, escritura y ejecución; el grupo y los demás sólo pueden ejecutar.
- **`rw-rw-rw- (666)`** – Todo el mundo puede leer y escribir en el archivo. ¡No es una buena elección!

- **rwxrwxrwx (777)** – Todo el mundo puede leer, escribir y ejecutar. ¡Tampoco es buena elección!

Utilizaremos el mismo ejercicio anterior para poder practicar con los permisos utilizando números, el único cambio que haremos será utilizar otro archivo llamado **foto2.png**:

Ejemplo	Descripción	Resultado
`$ touch foto2.png`	creamos el archivo foto2.png	`foto2.png`
`$ chmod 000 foto2.png`	quitamos todos los permisos al archivo foto2.png	`----------`
`$ chmod 700 foto2.png`	añadimos todos los permisos para el propietario	`-rwx------`
`$ chmod 710 foto2.png`	añadimos el permiso de ejecución para el grupo	`-rwx--x---`
`$ chmod 714 foto2.png`	añadimos el permiso de lectura para los otros usuarios	`-rwx--xr--`
`$ chmod 114 foto2.png`	eliminamos los permisos de lectura y escritura para el propietario	`---x--xr--`
`$ chmod 444 foto2.png`	establecemos como unico permiso de lectura para los 3 grupos	`-r--r--r--`
`$ chmod 555 foto2.png`	establecemos los permisos de lectura y ejecución para los 3 grupos	`-r-xr-xr-x`
`$ chmod 000 foto2.png`	quitamos todos los permisos	`----------`
`$ chmod 501 foto2.png`	añadimos los permisos de lectura y ejecución al propietario y ejecución a otros	`-r-x-----x`
`$ chmod 550 foto2.png`	añadimos permiso de lectura y ejecución al grupo y eliminamos permiso de ejecución a otros	`-r-xr-x---`
`$ chmod 770 foto2.png`	añadimos permiso de escritura y ejecución al propietario y grupo, y eliminamos permiso de ejecución a otros	`-rwxrwx---`
`$ chmod 666 foto2.png`	permite a cualquiera modificar el contenido e incluso eliminar el archivo	`-rw-rw-rw-`

CAMBIANDO PROPIETARIOS Y GRUPOS

Otro de los puntos a la hora de establecer permisos es la necesidad de poder cambiar el propietario y grupo del archivo o directorio, para hacer esta operación debe estar como root y los usuarios y grupos que utilizará deben haber sido creados previamente.

Cambiando el propietario

Utilizamos el comando chown explicado líneas arriba:

`# chown clases foto1.png`	# estamos cambiando el propietario del archivo, ahora el usuario clases será el propietario del archivo foto1.png

`# chown raul foto2.png`	# el usuario raul será el propietario del archivo foto2.png
`# chown clases datos/ –R`	Si vamos a cambiar el propietario de un directorio y con todos sus subdirectorios y archivos en forma **recursiva** utilizaremos la opción –R: # el usuario clases será el nuevo propietario de todos los archivos y subdirectorios que estén dentro del directorio datos/

¿Quién puede realizar un cambio de dueño sobre un fichero?
Sólo el **superusuario** (root) puede realizar un cambio de dueño

Cambiando el grupo

Utilizamos el comando chgrp explicado líneas arriba:

```
# chgrp clases foto1.png   # cambio el grupo del archivo, ahora el archivo foto1.png será del grupo clases
# chgrp raul foto2.png     # el archivo foto2.png será del grupo raul
Si vamos a cambiar el grupo de un directorio y con todos sus subdirectorios y archivos en forma recursiva utilizaremos la opción –R:
# chgrp clases datos/ –R   # todos los archivos y sub directorios del directorio datos/ serán del grupo clases
```

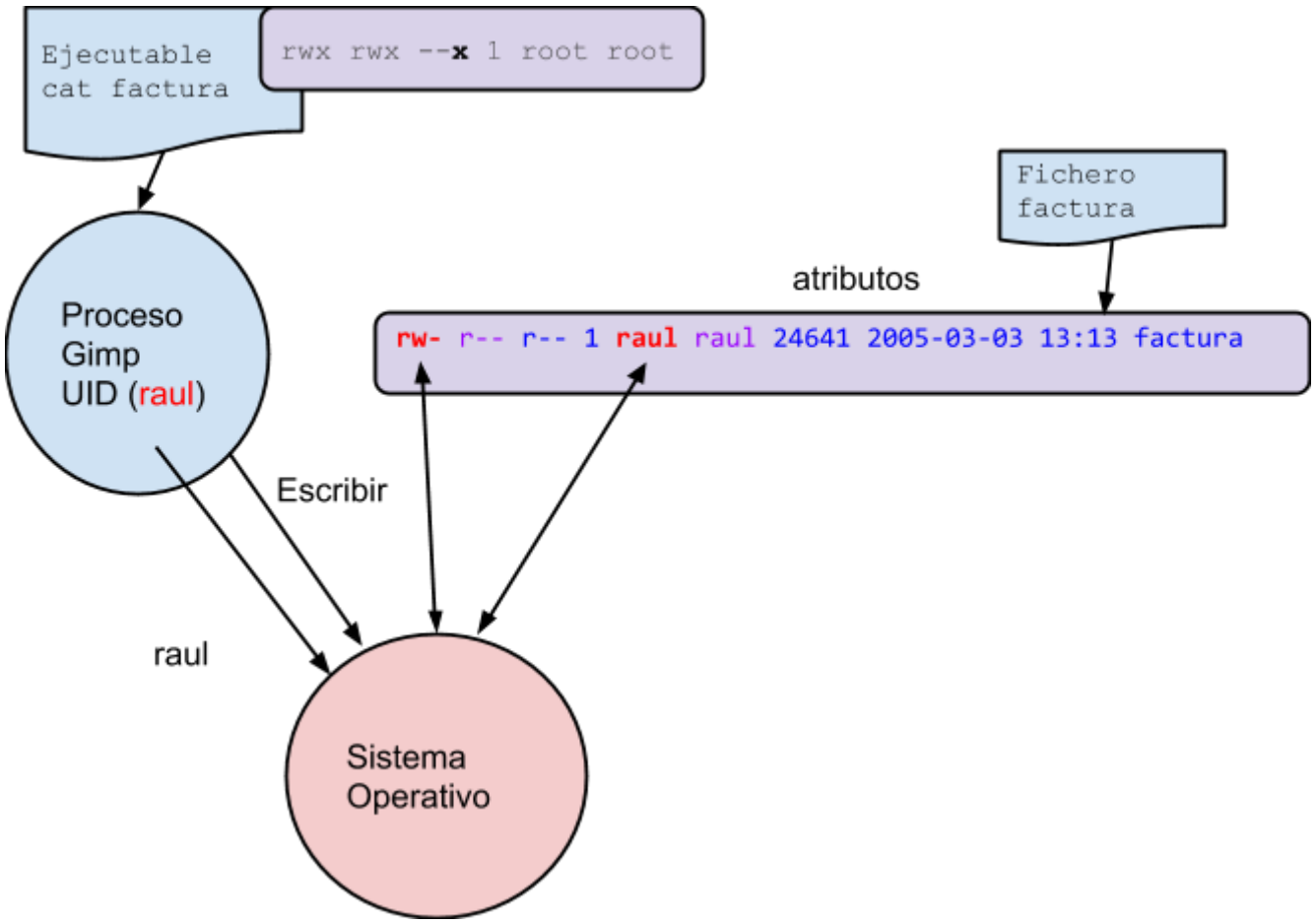
¿Quién puede realizar un cambio de grupo?

Cualquier usuario puede realizar un cambio de grupo sobre un fichero si se cumplen las siguientes dos condiciones:

- El usuario es dueño del fichero
- El usuario pertenece al grupo al que el fichero quiere ser asignado.

Permisos de los procesos en ejecución

Esta parte es difícil de entender. Los permisos sobre los ficheros se manifiestan en el momento en el que un proceso trata de escribir, leer o ejecutar un fichero. En ese momento se verifican las posibilidades. Observa la siguiente imagen que usaremos para entender esta idea.



Un usuario "raul" quiere ejecutar "cat factura", con él pretende leer el fichero factura. La secuencia de sucesos es la siguiente:

1. 'raul' escribe el comando por línea de comandos y pulsa enter.
2. El sistema operativo comprueba que raul no es el propietario del fichero ejecutable cat ("/usr/bin/cat"), después verifica que tampoco es miembro del grupo root y por ello se fija en los permisos que el fichero "*cat*" tiene para "*otros*". Observa que sí está marcada la "x" que indica que el fichero puede ser ejecutado por los otros.
3. Se crea un proceso a partir del fichero "*/usb/bin/cat*". Este proceso tiene un *uid* y un *euid* del usuario que ha creado el proceso, de raul. (El propietario del fichero "*cat*" (root) ya no tiene relevancia alguna)

Hasta aquí, se han verificado los permisos para ejecutar un proceso, pero ¡ No hemos terminado! De hecho ahora empieza la parte importante porque ya tenemos un proceso en marcha cuyo propietario es raúl. Todo lo que el proceso intente realizar será comprobando los privilegios de raul. Seguimos...

4. El proceso creado intenta leer del fichero "factura".
5. El sistema operativo verifica que raul es el propietario del fichero factura y en este caso, sí lo es. Por tanto se fija en los permisos del dueño (resaltados en rojo)
6. Al existir el permiso de escritura, el S.O. autoriza la operación de escritura que el proceso quería realizar.

Permisos entre diferentes usuarios

Vamos a describir una situación más compleja, para entender cómo funcionan realmente los permisos. Partiremos de la siguiente situación que debes reproducir en la medida de lo posible:

- Usaremos dos usuarios, en nuestro caso "pepe" y "maria".
- Trabajaremos en un directorio donde guardaremos ficheros de datos y programas.
- Los programas serán del usuario pepe
- Algunos ficheros de datos pertenecen a maria y otros a pepe.
- "albaranes" será el fichero de datos
- "manipular" será un programa ejecutable (tienes el código al final del documento, puedes compilarlo y generar un ejecutable). Este ejecutable puede leer o escribir en el fichero, lo decide el usuario cuando invoca al programa

La situación viene descrita en la siguiente imagen

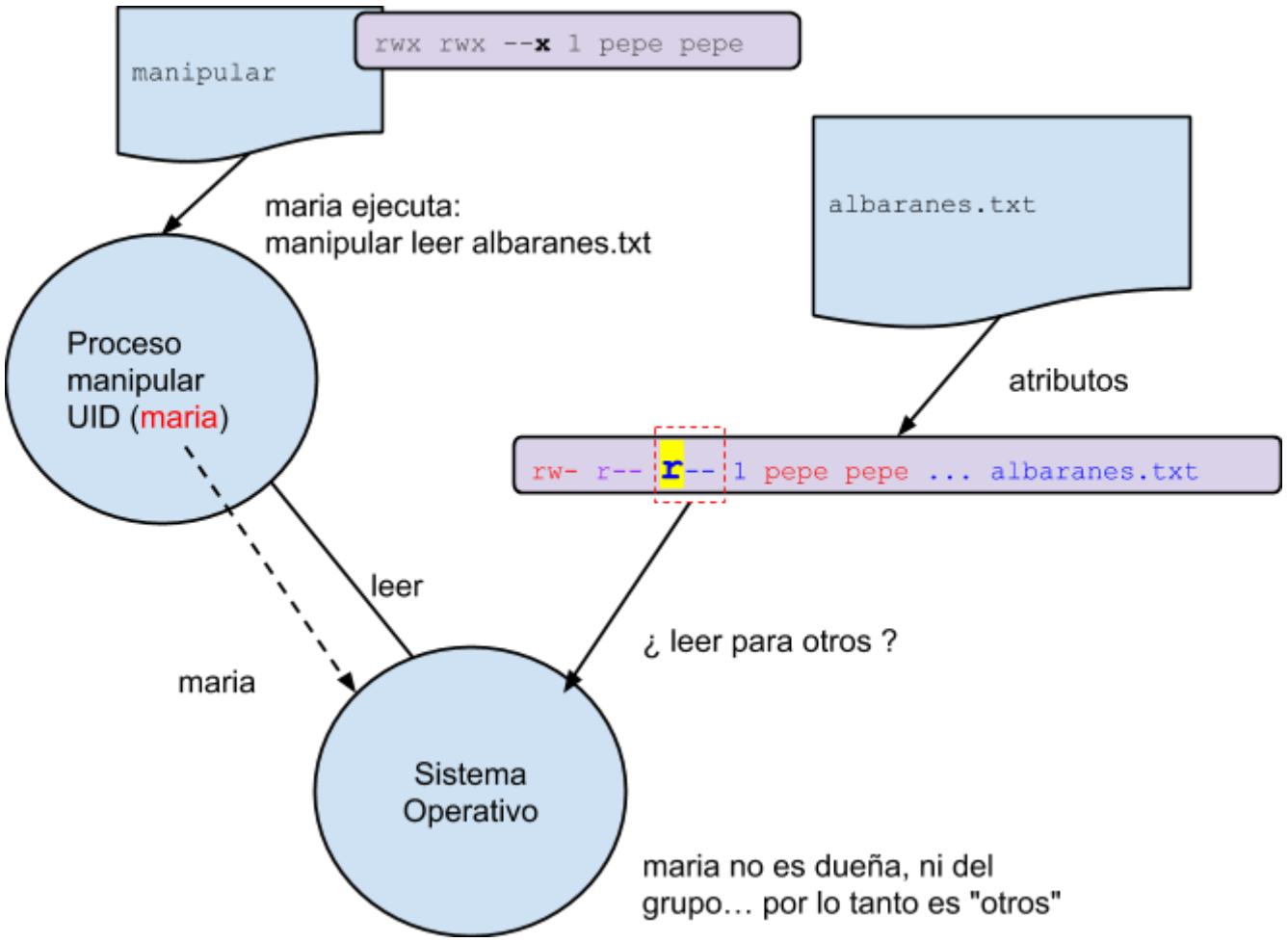
```
maria@2DAMASIX:/home/pepe > ll
total 40
-rw-rw-r-- 1 pepe  pepe     16 abr 29 13:11 albaranes
-rw-rw-r-- 1 pepe  pepe     16 abr 29 13:11 facturas
-rw-rw---- 1 maria maria    10 abr 29 12:24 ficheroMaria
-r--rw-r-- 1 pepe  pepe      1 abr 29 12:19 ficheroPermisos
-rwxrwxr-x 1 pepe  pepe  13251 abr 30 12:00 manipular*
-rw-r--r-- 1 pepe  pepe   1735 abr 30 12:00 manipular.cc
-rw-rw-r-- 1 maria maria    16 abr 29 13:22 maria
maria@2DAMASIX:/home/pepe > pwd
/home/pepe
maria@2DAMASIX:/home/pepe > whoami
maria
maria@2DAMASIX:/home/pepe >
```

- ¿Puede maria ejecutar el fichero manipular ?

  Sí, porque no siendo la dueña del fichero "manipular" ni perteneciendo al grupo, tiene permisos de ejecución

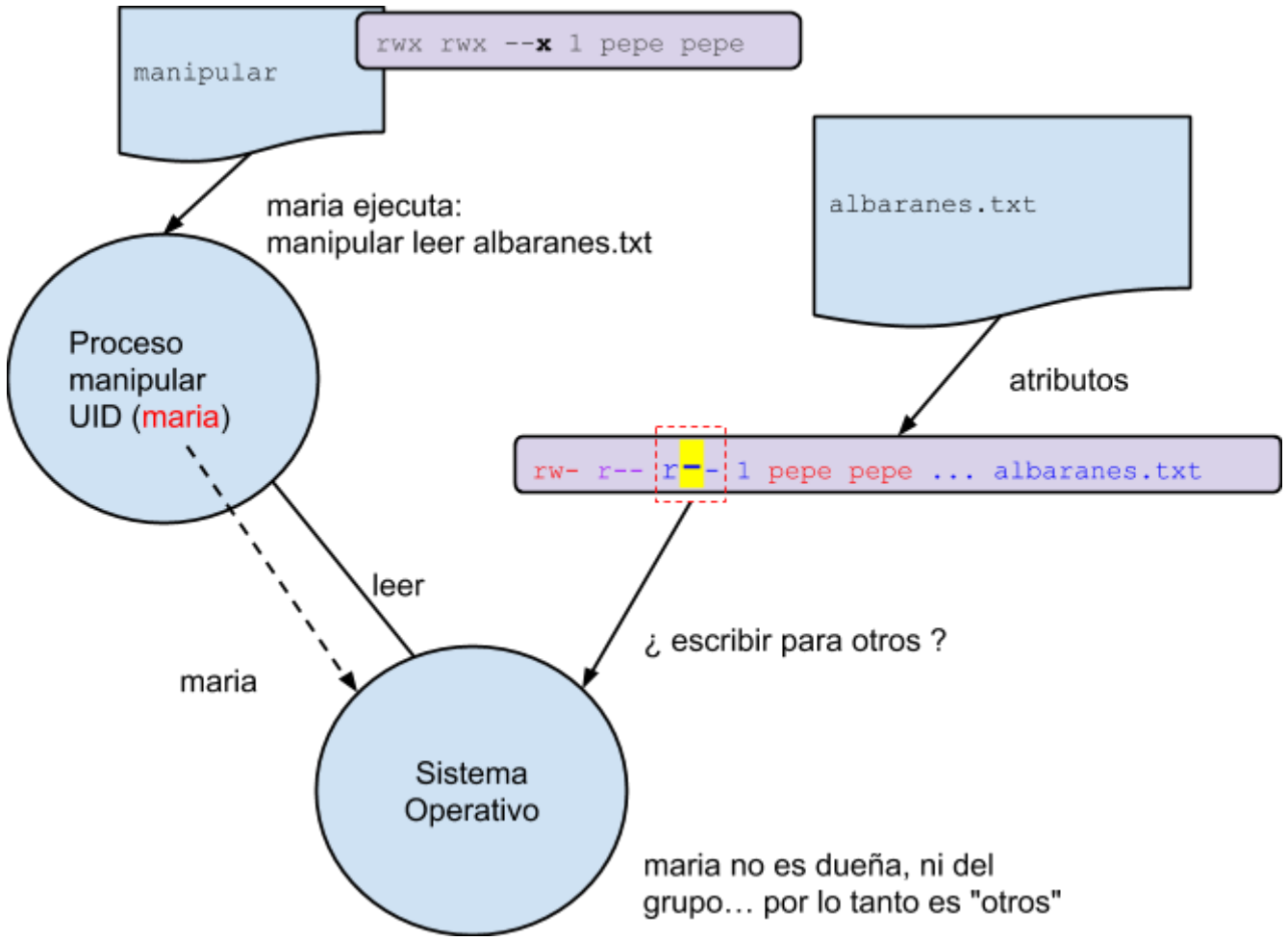
- ¿Puede maria ejecutar manipular para leer el fichero albaranes?

  Sí, porque cuando maría ejecute "manipular leer albaranes", el proceso creado tendrá como usuario que lo identifica a maria, y se le aplicarán privilegios de otros respecto del fichero "albaranes" (ya que no es la dueña de albaranes). Y entre esos privilegios está la lectura (r-- en el último grupo de permisos sobre albaranes)



- ¿Puede maría ejecutar "manipular" para escribir o modificar el fichero albaranes?

  NO, maria no puede cambiar el contenido del fichero albaranes. Cuando lance "manipular escribir albaranes", el proceso creado tendrá como usuario identificador a maria. Como ella no es dueña de albaranes (ni pertenece al grupo) se le aplicarán permisos de "otros", entre los cuales no está la ('w') escritura.



Sticky bit

Si has practicado con el comando `passwd`, habrás podido comprobar que cualquier usuario puede usar dicho comando y cambiar su contraseña. Realmente durante el cambio de contraseña, mientras se ejecuta el proceso "`passwd`", se accede y modifica el fichero /etc/shadow. ¿No te has preguntado cómo es posible que un proceso lanzado por un usuario modifique el fichero /etc/shadow? El `uid` del proceso "`passwd`" debería ser el del usuario que ha lanzado el proceso, y dicho usuario seguro que no tiene permisos para escribir en el fichero. Aquí vamos a descubrir el secreto.

Bits SUID, SGID y de persistencia (sticky bit)

Hay tres bits de permisos más: Bit de permisos SUID (Set User ID), el bit de permisos SGID (Set Group ID) y el bit de permisos de persistencia (sticky bit). Para entender los dos primeros; el SUID y el SGID veamos los permisos para el comando passwd anterior

```
#> ls -l /usr/bin/passwd
-r-s--x--x 1 root root 21944 feb 12  2006 /usr/bin/passwd
```

SUID

En vez de la 'x' en el trío de caracteres del dueño, encontramos ahora una 's' (suid). `passwd` es un comando propiedad de root, pero sin embargo debe de poder ser ejecutado por otros usuarios, no solo por root. Hasta aquí todo normal, la mayoría de comandos son así, propiedad de root, pero con permisos para que cualquiera los ejecute.

Es aquí donde interviene el bit SUID, donde al activarlo obliga al archivo ejecutable binario a ejecutarse como si lo hubiera lanzado el usuario propietario y no realmente quien lo lanzó o ejecutó. Es decir, es poder invocar un comando propiedad de otro usuario (generalmente de root) como si el propietario fuese quien lo lanza, y no otro usuario. En el caso del comando passwd anterior, es como si el usuario root fuese quien lanza el comando, y el proceso que se crea a partir de él estuviese asignado a root, con los consiguientes mayores privilegios.

Veámos el siguiente ejemplo:

```
maria@2DAMASIX:~ > ll
total 36
-rw------- 1 maria maria        42 abr 30 13:44 albaranes
-rw-rw-r-- 1 maria contables     5 abr 27 08:58 fichero
-rwsrwx--- 1 maria contables 13251 abr 30 13:32 manipular*
-rw-r--r-- 1 maria maria      1735 abr 30 13:32 manipular.cc
-rw-rw-r-- 1 maria maria         5 abr 29 11:53 nuevoFichero
-rw-rw-r-- 1 maria contables     5 abr 29 11:55 otroFichero
maria@2DAMASIX:~ >
```

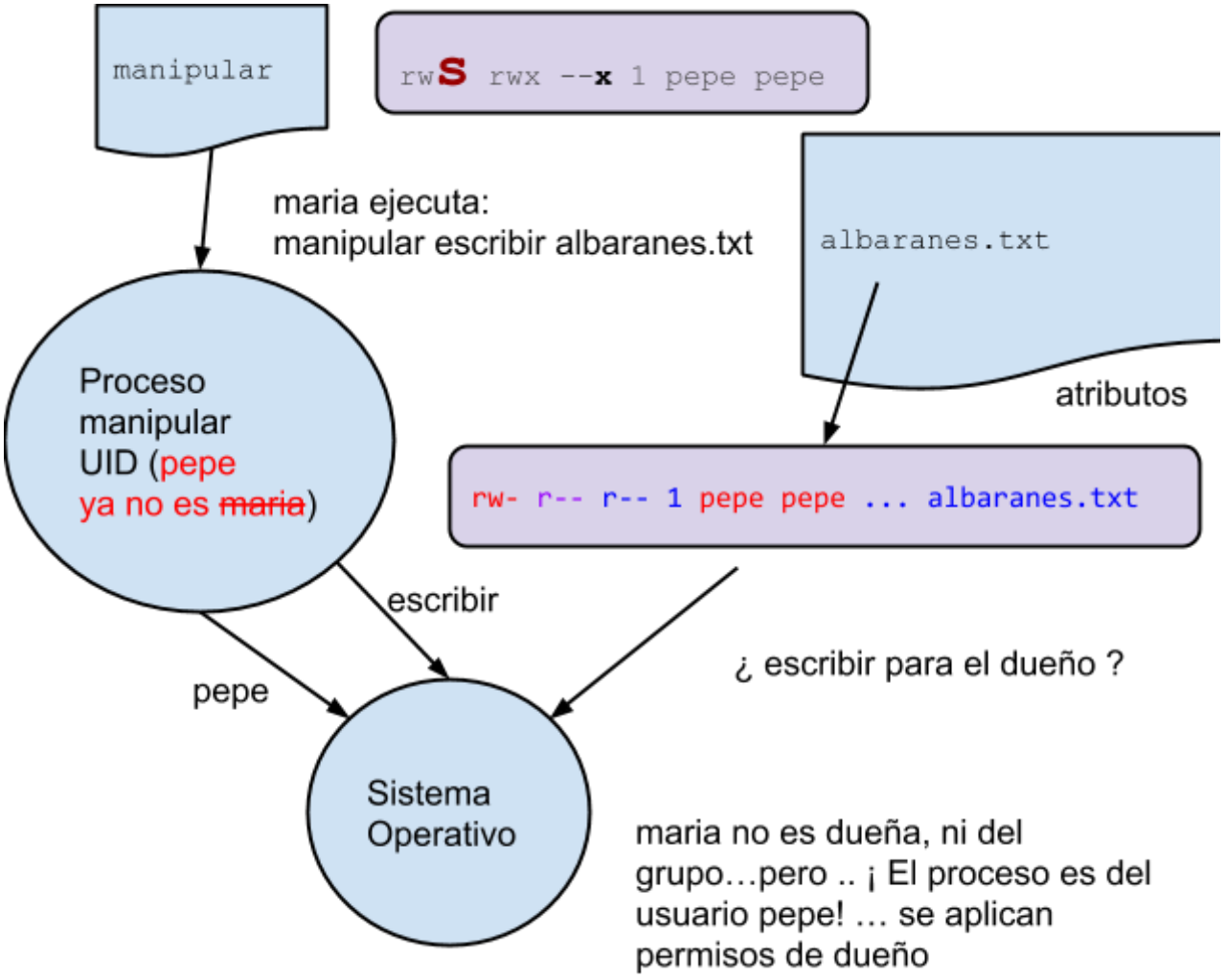
- El fichero "manipular" es un ejecutable que se usa para acceder a ficheros de datos, se usan indicando qué operación hay que hacer con el fichero. Ejemplo:
    - manipular leer fichero
    - manipular escribir fichero
- "albaranes" y "fichero" son ficheros de datos normales.
- como ves, los ficheros son del usuario "maria"
- El fichero "albaranes" no puede ser leído por nadie más que por maría. Ello implica que cualquier otro usuario no podría realizar las siguientes operaciones:
    - cat "albaranes"
    - tail "albaranes"
    - echo "albaran 126532" >> albaranes
    - read albaranes < albaranes

- Sin embargo, algunos otros usuarios sí que podrían ejecutar el fichero "manipular". Estos usuarios son los pretenecientes al grupo "contables", ya que el bit "x" está presente en el bloque de permisos para el grupo.

Suponga que juan es una usuaria miembro del grupo ""contables"". Cuando ella ejecute la siguiente línea de comandos:

manipular leer fichero

Se creará un proceso. Pero dicho proceso se creará con el UID del dueño del fichero "manipular". Esto es, con el UID de maria. A efectos prácticos este proceso lee y escribe en nombre de maria, por lo que la lectura y escritura sobre el fichero "albaranes" es como si la realizase maria, que SI tiene permisos. Por ello, la operaciòn tiene éxito.



SGID

El bit SGID funciona exactamente igual que el anterior solo que aplica al grupo del archivo. Es decir si el usuario pertenece al grupo 'ventas' y existe un binario llamado 'reporte' que su grupo es 'ventas' y tiene el bit SGID activado, entonces el usuario que pertenezca al grupo 'ventas' podrá ejecutarlo. También se muestra como una 's' en vez del bit 'x' en los permisos del grupo.

STICKY BIT (Bit de persistencia)

Este bit se aplica para directorios como en el caso de /tmp y se indica con una 't':

```
#> ls -ld /tmp
drwxrwxrwt 24 root root 4096 sep 25 18:14 /tmp
```

Puede apreciarse la 't' en vez de la 'x' en los permisos de otros. Lo que hace el bit de persistencia en directorios compartidos por varios usuarios, es que sólo el propietario del archivo pueda eliminarlo del directorio. Es decir cualquier otro usuario va a poder leer el contenido de un archivo o ejecutarlo si fuera un binario, pero sólo el propietario original podrá eliminarlo o modificarlo. Si no se tuviera el sticky bit activado, entonces en estas carpetas públicas, cualquiera podría eliminar o modificar los archivos de cualquier otro usuario.

Estableciendo los permisos especiales

Para cambiar este tipo de bit se utiliza el mismo comando `chmod` pero agregando un número octal (1 al 7) extra al principio de los permisos, ejemplo:

```
#> ls -l /usr/prog
-r-x--x--x 24 root root 4096 sep 25 18:14 prog
#>chmod 4511 /usr/prog
#> ls -l /usr/prog
-r-s--x--x 24 root root 4096 sep 25 18:14 prog
```

Nótese que el valor extra es el '4' y los demás permisos se dejan como se quieran los permisos para el archivo. Es decir, los permisos originales en este ejemplo eran 511 (r-x--x--x), y al cambiarlos a 4511, se cambió el bit SUID reemplazando el bit 'x' del usuario por 's'.

Los posibles valores serían los siguientes:

- - - - - - - - -	= 0	Predeterminado, sin permisos especiales. No se requiere indicar.
- - - - - - - - t	= 1	Bit de persistencia, sticky bit
- - - - - s - - -	= 2	Bit sgid de grupo
- - - - - s - - t	= 3	Bit sgid y sticky
- - s - - - - - -	= 4	Bit suid
- - s - - - - - t	= 5	Bit suid y sticky
- - s - - s - - -	= 6	Bit suid y sgid
- - s - - s - - t	= 7	Bit suid, sgid y sticky

Cambio de usuario durante la sesión en un terminal.

El comando "su" permite a un usuario entrar a un terminal como otro usuario. Es decir, supón que un usuario tiene una terminal abierta. Al escribir el siguiente comando

su juanito

Pasaría a ser preguntado por la contraseña de juanito. Y si es correctamente introducida, entonces se iniciaría una sesión de comandos en esa terminal bajo el usuario juanito. Al ejecutar el comando exit, se continuaría con la sesión anterior.

Existen dos variantes, su y "su -". con la primera tan sólo se cambia el usuario, pero no se realizan todas las tareas de entrada a la sesión normal de un usuario (por ejemplo, no aparecemos en el directorio de inicio). Mientras que añadiendo un guión como argumento, "su - juanito" se realizan todas las tareas y ejecutan scripts de inicio, y por tanto el entorno de juanito es el que prevalece durante esa sesión

Si no se pone usuario, se asume el usuario root.

Ejecución de comandos con otras credenciales (sudo)

<contraseña de usuario sudoer es pedidda>

Desde que los sistemas linux se han popularizado y extendido, se ha intentado acercar el sistema linux a los usuarios normales. Una de las tendencias es ocultar la existencia del usuario root por cuestiones de falta de familiaridad a este nombre ("root" suena mal para un usuario normal). Por contra, para configurar y administrar el sistema, se le otorgan poderes especiales a un usuario que suele ser creado durante la instalación. Dado que el nombre de este usuario lo elige la persona que instala el sistema y además suele ser el suyo propio (por ej: "juan", "jperez", "juanito", etc), resulta muy familiar trabajar con él y entender que este es el usuario principal.

La forma en la que este usuario puede administrar el sistema es la de "escalar" privilegios en la ejecución de ciertos comandos. Es decir, mientras ejecuta comandos de forma normal, tiene los mismos privilegios que un usuario normal y no puede administrar nada prácticamente. Pero llegado un momento, consigue obtener permisos para ejecutar algún comando reservado para root.

El comando sudo consigue ejecutar cualquier otro comando con los privilegios que tendría si lo ejecutase root. Observa la siguiente captura de pantalla:

```
lliurex@2DAMASIX:~ > whoami
lliurex
lliurex@2DAMASIX:~ > passwd pepe
passwd: You may not view or modify password information for pepe.
lliurex@2DAMASIX:~ > sudo passwd pepe
```

```
[sudo] password for lliurex:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
lliurex@2DAMASIX:~ >
```

1. En el primer comando "whoami" se manifiesta el usuario que está ejecutando el terminal que hemos capturado. Vemos que somos el usuario lliurex
2. Seguidamente intentamos cambiar la contraseña de otro usuario mediante el comando "passwd pepe". pepe es el usuario al que lliurex está intentándole cambiar la contraseña. Evidentemente, esto no está permitido y el sistema se niega y te advierte "*passwd: You may not….*"
3. Ahora volvemos a ejecutar el comando, pero anteponemos "sudo" (realmente ejecutamos el comando "sudo" y "password pepe" son simplemente argumentos para sudo). Lo que ocurre es lo siguiente:
    a. sudo verifica que el usuario "lliurex" tiene privilegios y le está permitido usar el comando sudo.
    b. Seguidamente se le pregunta al usuario lliurex por su contraseña (no por la de root).
    c. Si la contraseña es correcta, se ejecuta el comando "passwd pepe" pero con los privilegios que tendría root, y ¡ Así sí !

Si se han creado más usuarios despueś de la instalación (con useradd o adduser, por ejemplo), verás que éstos usuarios nuevos no pueden ejecutar sudo al igual que el usuario que instaló el sistema. Si se desea otorgar esa posibilidad, hay que actuar sobre la configuración del comando sudo o sobre la pertenencia a grupos:

- modificar el archivo /etc/sudoers. Allí se puede especificar qué usuarios pueden ejecutar qué comandos. Observa, ya que estás, la línea:

    ```
    # Members of the admin group may gain root privileges
    %admin ALL=(ALL) ALL
    ```

    Hay más líneas, pero ésta en concreto indica que cualquier miembro del grupo de administradores podrá ejecutar cualquier comando

- Basándonos en lo anterior, podemos dar permisos a otro usuario, haciéndolo miembro del grupo de administradores. Observa el fichero /etc
-

Ejercicio: ¿Qué hace la siguiente línea?

```
echo  ' su - root -c '"echo $USER"' | xargs  sudo
```

¿Y ésto?

```
sysop@pc01:~$ cat passwordPass.sh
#!/bin/bash
echo a
sysop@pc01:~$ chmod oug+x passwordPass.sh
sysop@pc01:~$ SUDO_ASKPASS=$HOME/passwordPass.sh
sysop@pc01:~$ export SUDO_ASKPASS
sysop@pc01:~$ echo  'su - -c "echo $USER"' |xargs sudo
-A2>/dev/null
root
sysop@pc01:~$
```

Comando " su "

por rellenar.

ACL

https://help.ubuntu.com/community/FilePermissionsACLs

Ejercicios

Ejercicios

Ejercicios aprendizaje de clase. Estos ejercicios expresan una intención. Léelos antes de hacerlos y asimila dicha intención

1. Crear tres usuarios 'pepe', 'juan' y 'maria'.
    1. Los usuarios pepe y maria deben ser creados con el comando de alto nivel que no plantea problemas.

    2. Para el usuario juan, usa el comando "useradd" (el comando incómodo). Intenta usando usermod, recrear toda la configuración de un usuario normal (su contraseña, su directorio de inicio, su grupo primario, etc)
    3. Si no se ha creado un grupo "juan", intenta crear un grupo juan y después configura el usuario para que éste sea su nuevo grupo primario.
    4. Cambiar o crear las contraseñas igual que los nombres de los usuarios creados.
2. Comprobad que existen los directorios de inicio, que se puede conectar como dicho usuario y que en su directorio existen ficheros con permisos, dueños y grupo bien asignados
3. Cread un grupo llamado **'contables'**
    1. Nombra como administrador del grupo "contables" a"pepe".
    2. Observa y anota los cambios en los ficheros de definición de grupos y usuarios. Demuestra que pepe es el administrador del grupo "contables".
    3. ¿Es además pepe miembro de dicho grupo? ¿Debe serlo?
4. Entra al sistema como pepe
    1. Intenta meter a "juan" como miembro del grupo de 'contables'.
    2. Intenta cambiar la contraseña del grupo 'contables'.
5. Entra al sistema como maria
    1. Toma privilegios para pertenecer Temporalmente al grupo "contables"
    2. Verifica con el comando adecuado los grupos a los que pertenece.
    3. Crea un fichero nuevo y observa sus atributos.
    4. Cámbia los atributos de grupo una vez creado el grupo y restáuralos
    5. Intenta volver a ser capaz de crear ficheros cuyo grupo sea maria, (sin salir de la sesión)
6. Entra al sistema como "maria" e:
    1. Intenta hacerte miembro del grupo 'contables' usando la contraseña del grupo
    2. Intenta usando la contraseña, pertenecer temporalmente otra vez al grupo "contables"
    3. Intenta hacerte miembro del grupo "contables" ahora que perteneces temporalmente a él.

USUARIOS Y PERMISOS SOBRE FICHEROS.

7. Entrar a la sesión como pepe.
    1. Cread dos ficheros facturas y albaranes en /home/pepe. Ponle algo de contenido
    2. Permitid que maria y juan puedan leer el contenido del fichero "factura" pero no puedan averiguar que existe un fichero llamado "albaranes"
    3. Permite a maria poder crear ficheros nuevos en el directorio de pepe. Haz una prueba creando como maria un fichero

1. Intenta que el fichero creado por maría no pueda ser leído por pepe (incluso estando en su directorio)

8. Entrar en la sesión como pepe y restaurar los permisos normales de su directorio de inicio.
    1. Crea un directorio llamado docscontables.
    2. modifica el directorio docsContables para que pertenezca al grupo contables. Hazlo siendo pepe.
    3. Crea un fichero en su interior llamado "docImportanteContables". Verifica los atributos del fichero
    4. Impide que usuarios que no son del grupo de contables puedan leer el contenido del fichero "docImportanteContables" de dos maneras
        1. sin modificar los permisos del fichero.
        2. modificando los permisos y/o atributos del fichero

        Después de probar la opción 1, restaura los cambios a la situación anterior para afrontar el punto 2.

    5. Modifica los permisos necesarios para que nadie pueda manipular los directorios de pepe en /home/pepe, excepto los usuarios de contables en el directorio "docImportanteContables". Es decir, que ya que el directorio está dentro de /home/pepe, los demás usuarios no puedan interferir con las cosas de pepe, pero puedan seguir trabajando los contales con los ficheros del directorio /home/pepe/docsContables.

9. Crea un directorio /home/contables pensado para la gestión contable que será utilizable por el grupo contables
    1. Los usuarios del grupo contables pueden crear y borrar ficheros. Los demás usuarios no pueden entrar para nada al directorio, ni listar ni borrar ficheros.
    2. Los ficheros creados por la usuaria juan pueden ser leídos y escritos por ella, leídos por los miembros del grupo contables y los demás no podrán leer ni escribir en esos ficheros, independientemente del resto de permisos del directorio.

        Los permisos de estos ficheros se deben aplicar desde el primer momento de su creación y no cambiándolos despueś de creados

    3. La usuaria maria, que no pertenece al grupo contables, sabe la contraseña del grupo. logra con ello que entre en el directorio /home/contables y cree un fichero allí. Éste fichero no debería poder ser leído por los del grupo contables pero sí por otros usuarios que no son del grupo.

Bit setuid básico

10. El usuario pepe, creará con el fichero fuente "manipular.cc" un programa para leer algunos de sus ficheros (por ejemplo "fichSecreto") . Estos ficheros sólo podrán ser accedidos por el resto de usuarios con el programa manipular creado. Es decir, juan y maria no podrán leer el fichero con 'cat', 'gedit' o similar, pero sí podrán lanzar el programa para acceder a esos ficheros 'manipular leer fichSecreto', o 'manipular escribir fichSecretos')

11. Ahora, la usuaria maria es la que desea hacer algo similar al ejercicio anterior, para restringir el acceso a uno de sus ficheros, pero quiere controlar más quien puede acceder a él
    1. Usará igualmente manipular.cc y creará un ejecutable.
    2. Al igual que pepe anteriormente, no desea que nadie más pueda mirar directamente sus ficheros, ni escribir en ellos (mediante gedit, cat, etc). Todos los demás usuarios únicamente podrán acceder via el ejecutable "manipular" que ella crea
    3. Esta vez desea que los usuarios del grupo contables, usando un programa creado a partir de "manipular.cc" puedan acceder en lectura y escritura, mientras que el resto de usuarios que no pertenecen al grupo de contables, tan sólo puedan usar el programa para acceder en lectura de forma que no puedan hacer 'manipular escribir ficheroSecretoDeMaria'

Ejercicio compacto (solucionado)

Resumen de usuarios y grupos

Usuarios	pepe, juan y maria
Grupo (solo hay uno)	contables
Miembros de "contables"	juan y pepe
Administrador del grupo contables	juan

1. Fase 1
    1. Crea un directorio dentro de pepe pensado para el Grupo Contables, llamado "Facturas"
    2. Permite que los usuarios del grupo contables puedan leer y cambiar el contenido del directorio.
    3. Los usuarios que no son del grupo contables podrán entrar, pero no cambiar el contenido (añadir o quitar ficheros)

4. Los usuarios del grupo contables crearán ficheros y deberán cambiar los permisos para que
    1. los de su grupo puedan leer y escribir en el fichero,
    2. Los otros sólo puedan leer el fichero
    3. El mismo usuario que crea el fichero sólo pueda escribir, no leerlo.

Pruebas una vez hecho el directorio

1. Entra como María (Que no es del grupo contables) y
    a. Entra en el directorio facturas.
    b. Lista el directorio
    c. Crea un fichero allí → no deberías poder
2. Entra como juan y repite los pasos anteriores para maria. Deberías poder hacerlo todo. Pero además
    a. lee el fichero que creas allí

Prueba una vez pepe ha creado un fichero y le ha cambiado los permisos para cumplir con lo que dice el enunciado. Supongamos que ha creado y cambiado los permisos de un fichero llamado "facturaNueva"

1. Entra como pepe al directorio
    a. lista el directorio
    b. añade contenido al fichero "facturaNueva"
    c. lee el contenido de "facturaNueva" → no deberías poder
2. Entra como juan y repite los pasos de pepe. Deberías poder hacerlo todo
3. Entra como maria y repite los pasos de pepe, No deberías poder añadir, pero sí leer.

2. Fase 2. En esta fase vamos a utilizar el ejecutable obtenido a partir de la compilación del fichero "manipular.c" que hay en el anexo de este documento. Este ejecutable debe ser compilado y su función es "leer", "modificar" o "modificar añadiendo" cualquier fichero que se le pasa por argumentos. Usaremos este ejecutable para que los diferentes usuarios accedan y manipulen los ficheros de texto que hasta ahora se accedían con comandos como "cat fichero", "echo texto >fichero" o "gedit Fichero". Llamemos "manipular" a este ejecutable
    a. Añade ese ejecutable al directorio "facturas" y configúralo para que sólo los del grupo contables puedan ejecutarlo. Repite todas las pruebas y saca conclusiones

3. En esta fase se van a cambiar las condiciones. Ahora se confía algo más en la usuaria maria. Se desea :
    a. Que maría pueda seguir leyendo los ficheros del directorio pero no pueda escribir sobre ellos de cualquier manera (antes ya era así)
    b. Se desea, que maría pueda "escribir" en los ficheros del directorio, pero sólo usando el ejecutable manipular. Este programa es seguro y se confía en que

usándolo se puede garantizar la seguridad de los ficheros del grupo contabilidad.

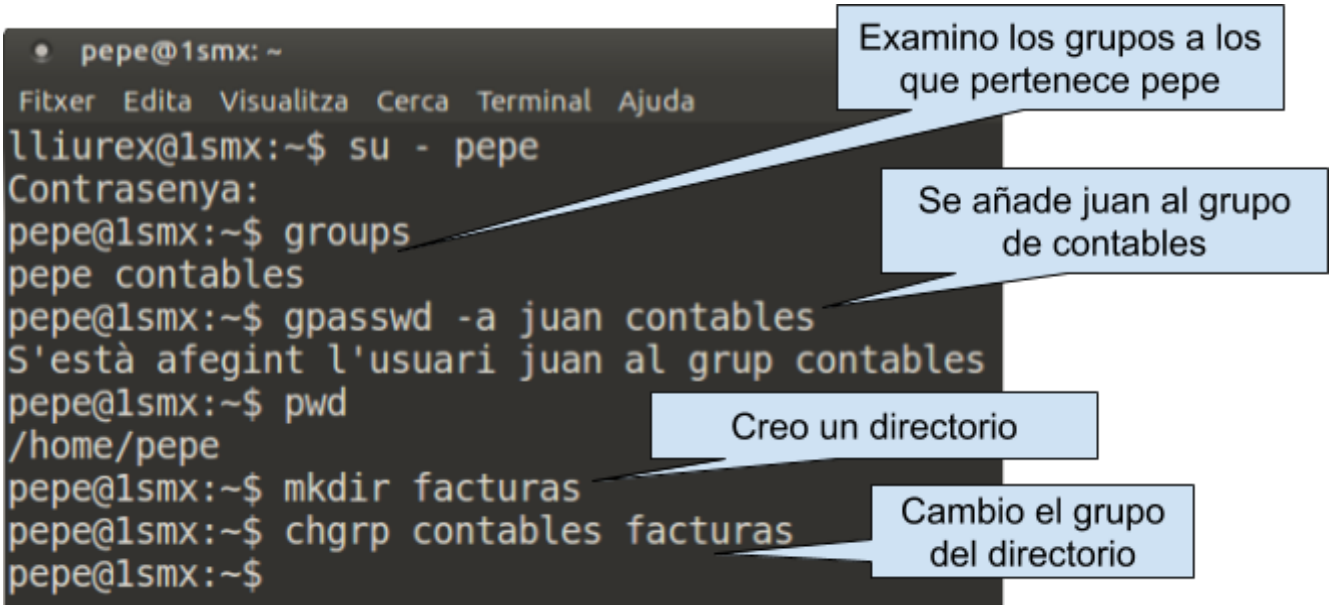
Solución

comandos y pantallas soluciones:

Para crear los usuarios y grupos hay que ser root y ejecutar la siguientes secuencia de comandos:

```
adduser juan
adduser pepe
adduser maria
addgroup contables
gpasswd -A pepe
adduser pepe contables
# ahora convertiremos a pepe en administrador del grupo contables
gpasswd -A pepe contables
```

Dado que pepe es administrador del grupo, ya no es preciso que root sea el que añada al usuario juan. Esto puede hacerlo pepe



El grupo y dueño del directorio es el que aparece a continuación:

```
pepe@1smx:~$ ls -l
total 4
drwxrwxr-x 2 pepe contables 4096 des 14 12:26 facturas
pepe@1smx:~$
```

Para continuar con el ejercicio hemos de crear documentos y empezar a probar quién puede hacer qué cosas.

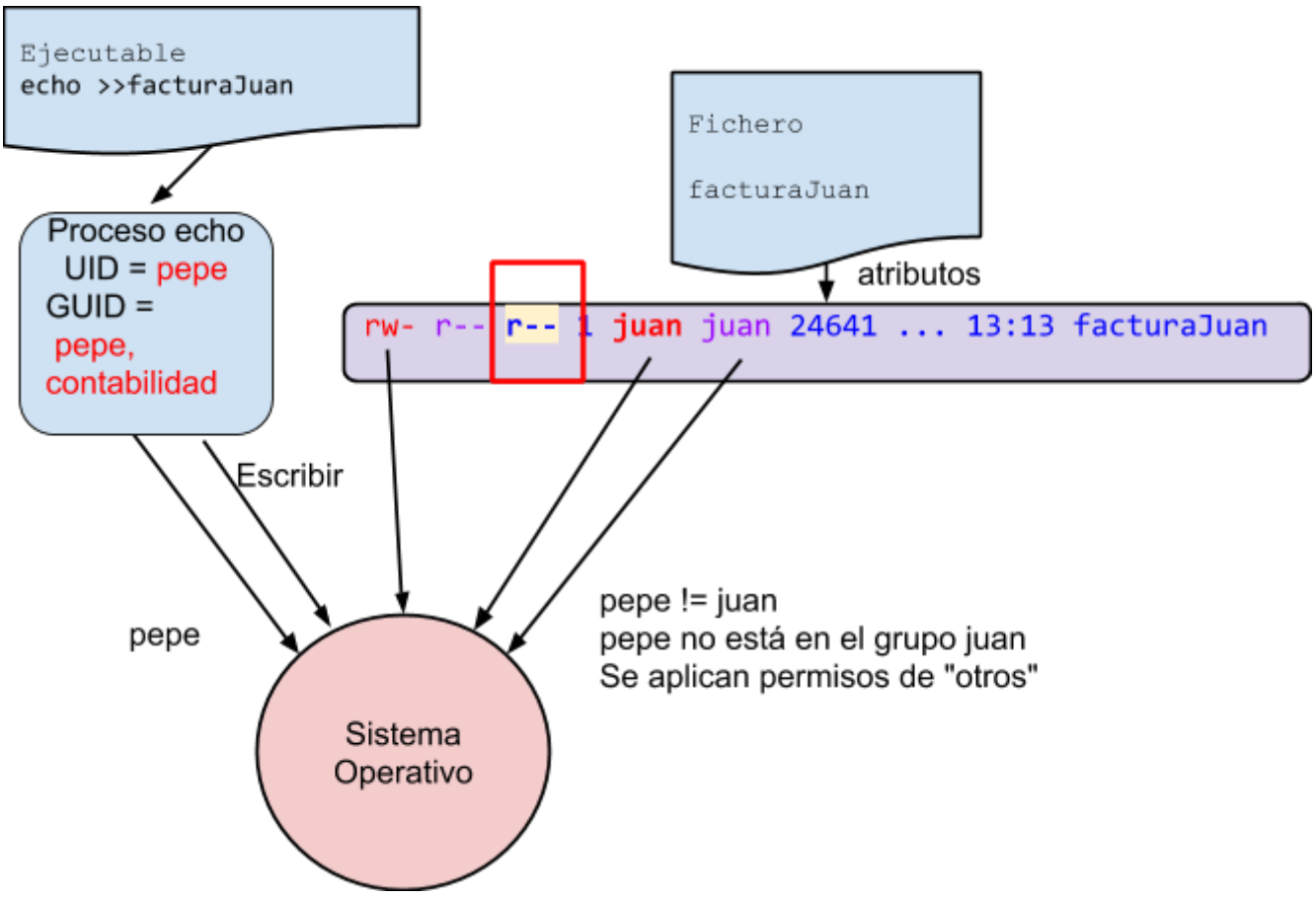
Vamos a pasar a ser el usuario juan, así también probamos que el usuario pertenece al grupo. Juan Crea documentos dentro del directorio facturas

```
pepe@1smx:~$ su - juan
Contrasenya:
juan@1smx:~$ cd /home/pepe/facturas/
juan@1smx:/home/pepe/facturas$ echo "contenido" > facturaJuan
juan@1smx:/home/pepe/facturas$ ls -l
total 4
-rw-rw-r-- 1 juan juan 10 des 14 12:32 facturaJuan
juan@1smx:/home/pepe/facturas$
```

Ahora intentamos -siendo pepe- escribir, Pero pepe no puede "ampliarlos"

```
pepe@1smx:~$ cd facturas/
pepe@1smx:~/facturas$ cat facturaJuan
contenido
pepe@1smx:~/facturas$ echo "ampliar contenido" >> facturaJuan
-su: facturaJuan: S'ha denegat el permís
pepe@1smx:~/facturas$ ls -l
total 4
-rw-rw-r-- 1 juan juan 10 des 14 12:32 facturaJuan
pepe@1smx:~/facturas$
```

Analicemos qué está pasando



cambiar el grupo para cumplir deseos del enunciado. ¿Ojo quien cambia el grupo del fichero ! Sólo el dueño puede hacerlo. Observa:

```
pepe@1smx:~/facturas$ chgrp contables facturaJuan
chgrp: s'està canviant el grup de 'facturaJuan': L'operació no és permesa
pepe@1smx:~/facturas$ su - juan
Contrasenya:
juan@1smx:~$ cd /home/pepe/facturas/
juan@1smx:/home/pepe/facturas$ chgrp contables facturaJuan
juan@1smx:/home/pepe/facturas$
```

falta eso de que "el dueño no puede leer sus ficheros"

```
juan@1smx:/home/pepe/facturas$ cat facturaJuan
contenido
juan@1smx:/home/pepe/facturas$ chmod u-r facturaJuan
juan@1smx:/home/pepe/facturas$ cat facturaJuan
cat: facturaJuan: S'ha denegat el permís
juan@1smx:/home/pepe/facturas$
```

Fase 2

```
pepe@1smx:~$ cp /home/lliurex/manipular.c .
pepe@1smx:~$ gcc -o  manipular manipular.c
pepe@1smx:~$ mv manipular facturas/
pepe@1smx:~$
```

Ahora hay que cumplir el enunciado porque hay cosas que no funcionan, como demuestra el ejemplo siguiente:

```
pepe@1smx:~/facturas$ ls -l
total 20
--w-rw-r-- 1 juan contables    10 des 14 12:32 facturaJuan
-rwxrwxr-x 1 pepe pepe      13304 des 14 12:48 manipular
pepe@1smx:~/facturas$ su maria
Contrasenya:
maria@1smx:/home/pepe/facturas$ ./manipular escribir facturaJuan
Escribir
No tienes permiso
 maria@1smx:/home/pepe/facturas$
```

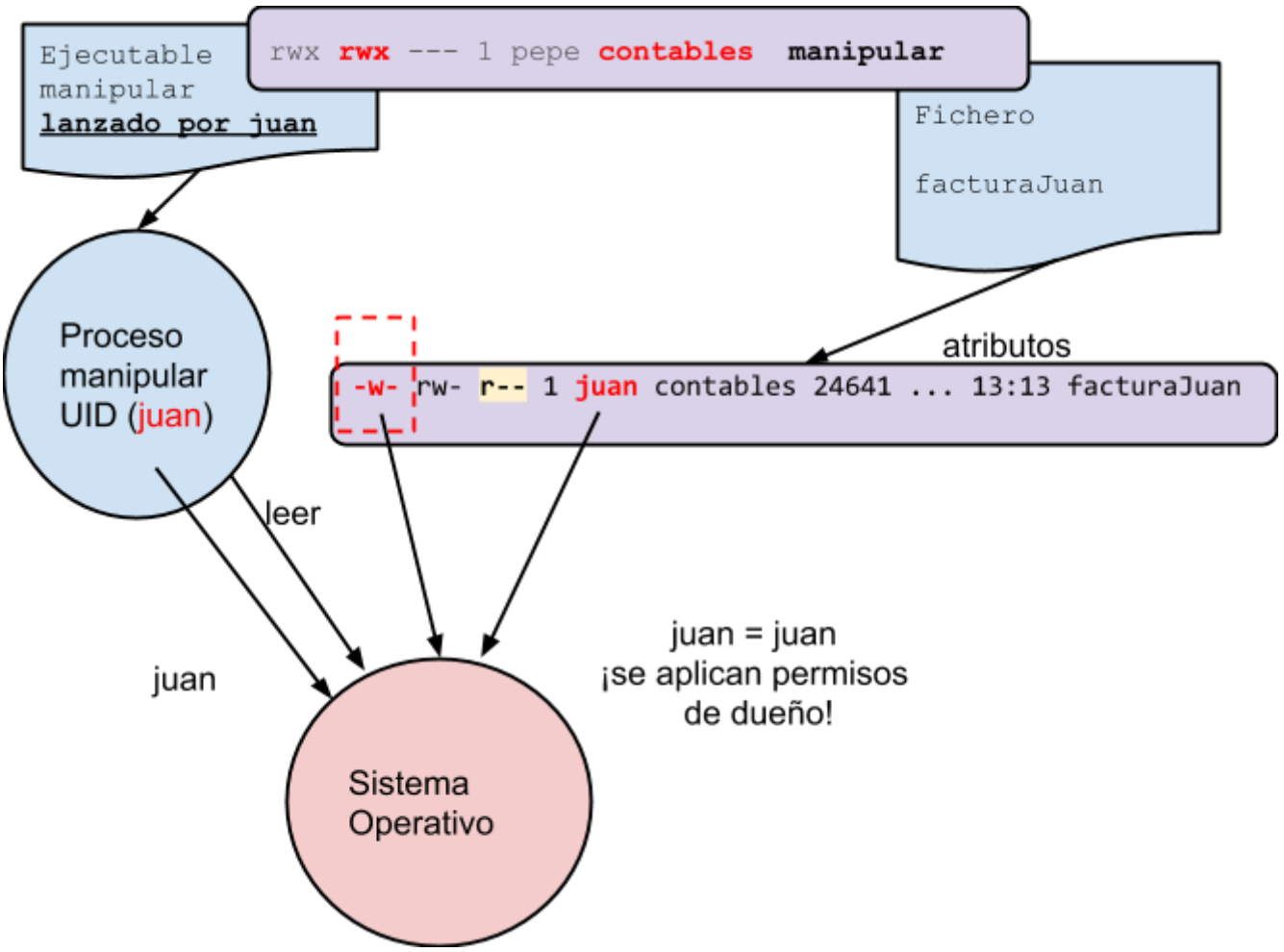
maria ¡No tenía permiso ni siquiera para ejecutar "manipular" !

Solución. La solución es quitar permisos de ejecución sobre "manipular" a los "otros", pero para evitar que juan se vea impedido para ejecutar ese fichero, hay que cambiar el fichero "manipular" de grupo. Así los miembros del grupo contables sí podrán ejecutar el fichero, pero los que no son miembros no.

En el ejemplo siguiente, se cambian los permisos pero no el grupo y se prueba con juan. A continuación, ya se establece el grupo adecuado.

```
pepe@1smx:~/facturas$ ls -l
total 20
--w-rw-r-- 1 juan contables    10 des 14 12:32 facturaJuan
-rwxrwxr-x 1 pepe pepe      13304 des 14 12:48 manipular
pepe@1smx:~/facturas$ chmod o-x manipular
pepe@1smx:~/facturas$ su juan
Contrasenya:
juan@1smx:/home/pepe/facturas$ ./manipular leer facturaJuan
bash: ./manipular: S'ha denegat el permís
juan@1smx:/home/pepe/facturas$ exit
exit
pepe@1smx:~/facturas$ chgrp contables manipular
pepe@1smx:~/facturas$ ls -l
total 20
--w-rw-r-- 1 juan contables    10 des 14 12:32 facturaJuan
-rwxrwxr-- 1 pepe contables 13304 des 14 12:48 manipular
pepe@1smx:~/facturas$ su juan
Contrasenya:
juan@1smx:/home/pepe/facturas$ ./manipular leer facturaJuan
leer
No tienes permiso
 juan@1smx:/home/pepe/facturas$
```

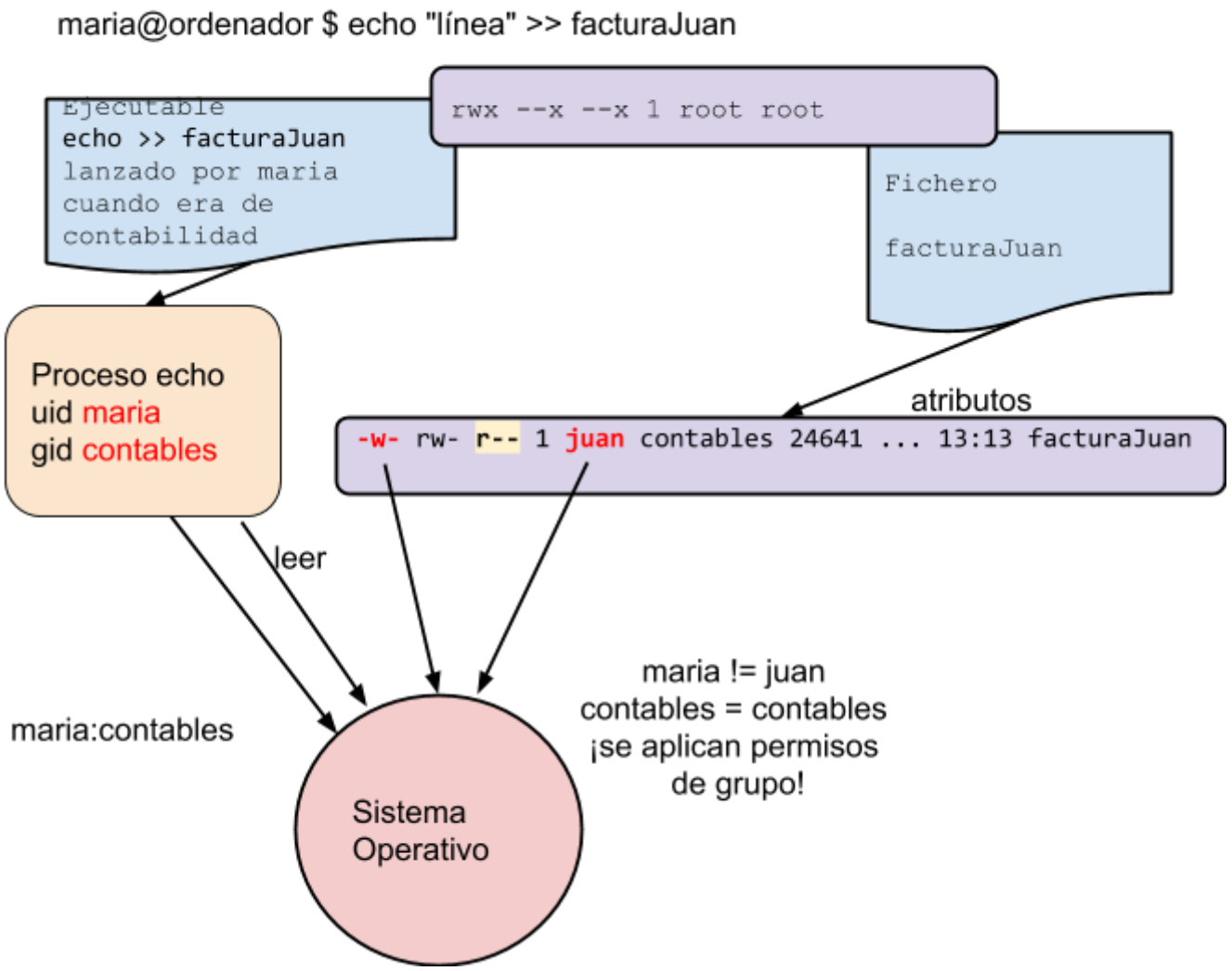
El hecho de no poder acceder al fichero finalmente ¡ No es incorrecto ! es lo que el enunciado dice: "un usuario no podrá leer sus archivos". la explicación de lo que ocurre la ofrece el siguiente diagrama



Solución FASE 3

Hay una solución no correcta pero aproximada, usando el comando `newgrp` y contraseñas de grupo. Sin embargo, Aquí se solventa mediante el uso del bit setuid

```
Tornau a escriure la nova contrasenya:
pepe@1smx:~/facturas$ su maria
Contrasenya:
maria@1smx:/home/pepe/facturas$ id
uid=1006(maria) gid=1006(maria) grups=1006(maria)
maria@1smx:/home/pepe/facturas$ groups
maria
maria@1smx:/home/pepe/facturas$ newgrp contables
Contrasenya:
maria@1smx:/home/pepe/facturas$ groups
contables maria
maria@1smx:/home/pepe/facturas$ id
uid=1006(maria) gid=1007(contables) grups=1007(contables),1006(maria)
maria@1smx:/home/pepe/facturas$ echo "hola" >facturaMaria
maria@1smx:/home/pepe/facturas$ ls -l
total 24
--w-rw-r-- 1 juan  contables    10 des 14 12:32 facturaJuan
-rw-rw-r-- 1 maria contables     5 des 14 13:15 facturaMaria
-rwxrwxr-- 1 pepe  contables 13304 des 14 12:48 manipular
maria@1smx:/home/pepe/facturas$ ./manipular anyadir facturaJuan
Anyadir
fichero escrito
maria@1smx:/home/pepe/facturas$ echo "linea peligrosa" >> facturaJuan
maria@1smx:/home/pepe/facturas$
```

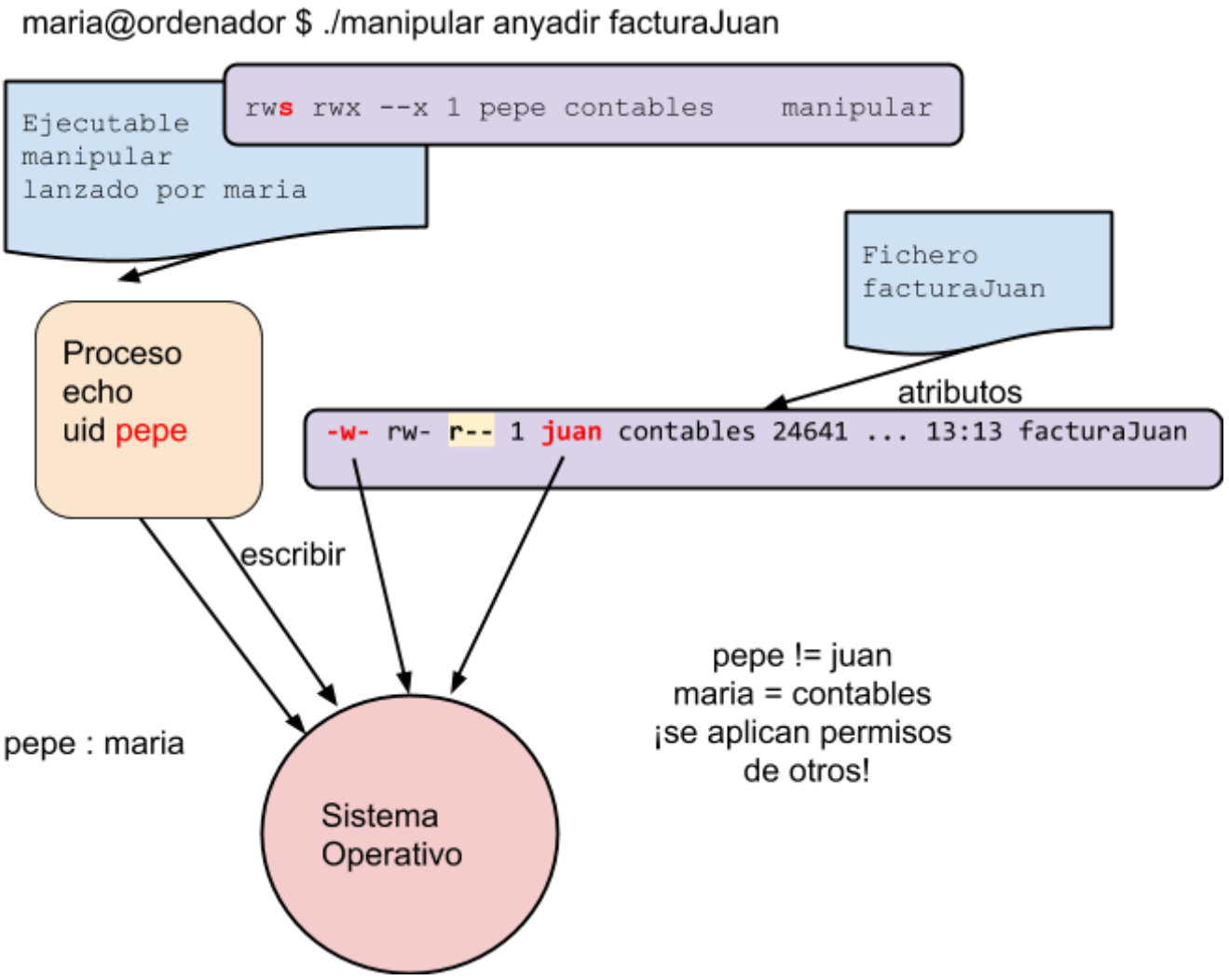


bit setuid ya

Primer intento

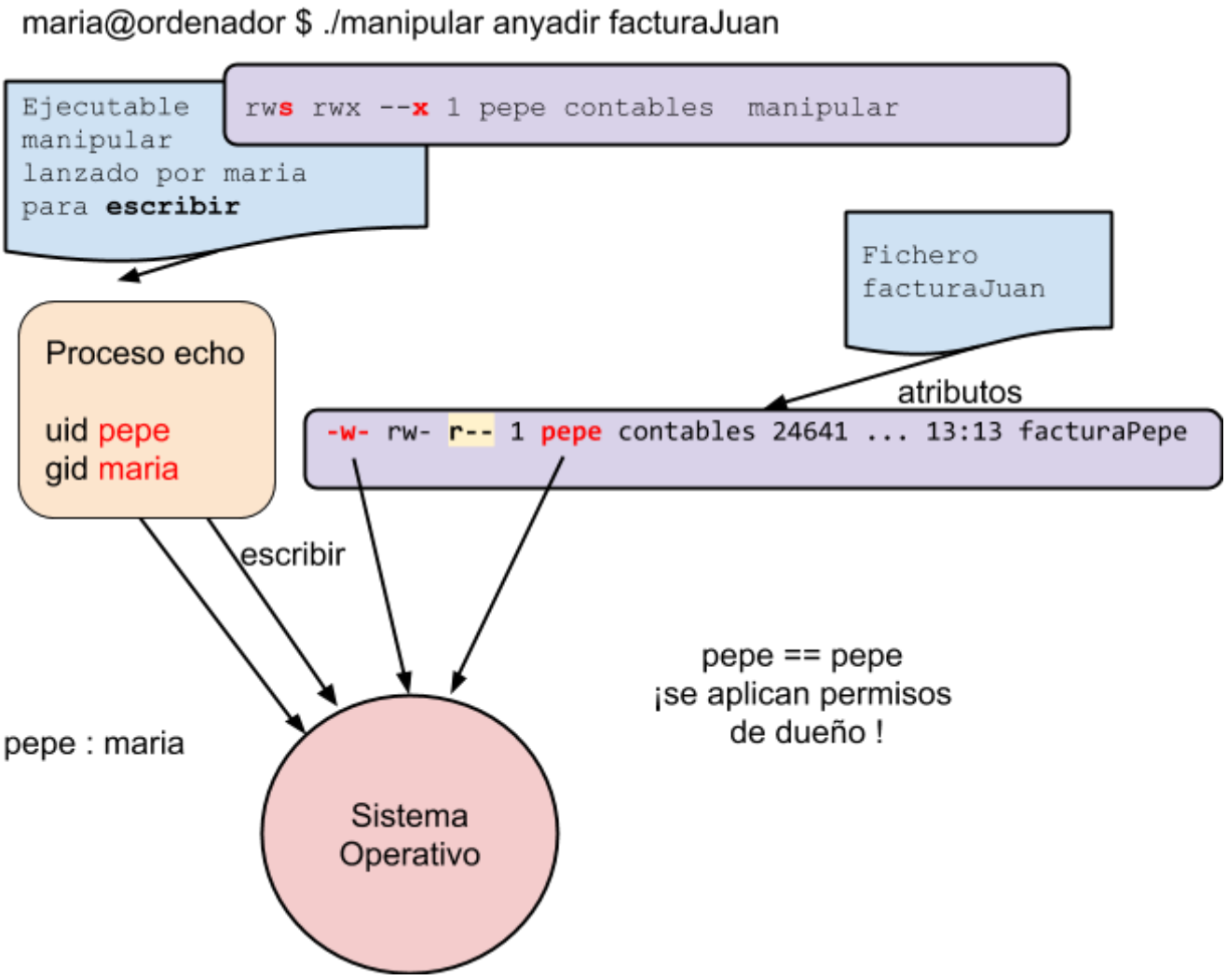
```
pepe@1smx:~/facturas$ chmod u+s,o+x manipular
pepe@1smx:~/facturas$ ls -l
total 24
--w-rw-r-- 1 juan  contables    64 des 14 13:17 facturaJuan
-rw-rw-r-- 1 maria maria         5 des 14 13:15 facturaMaria
-rwsrwxr-x 1 pepe  contables 13304 des 14 12:48 manipular
pepe@1smx:~/facturas$ su maria
Contrasenya:
maria@1smx:/home/pepe/facturas$ ./manipular anyadir facturaJuan
Anyadir
No tienes permiso
 maria@1smx:/home/pepe/facturas$
```

¿Qué pasa ?



Pero con un fichero de pepe sí que funciona:

```
pepe@1smx:~/facturas$ echo "datos" >> facturaPepe
pepe@1smx:~/facturas$ ls -l
total 28
--w-rw-r-- 1 juan  contables    64 des 14 13:17 facturaJuan
-rw-rw-r-- 1 maria maria         5 des 14 13:15 facturaMaria
-rw-rw-r-- 1 pepe  pepe          6 des 14 13:42 facturaPepe
-rwsrwxr-x 1 pepe  contables 13304 des 14 12:48 manipular
pepe@1smx:~/facturas$ su maria
Contrasenya:
maria@1smx:/home/pepe/facturas$ ./manipular anyadir facturaPepe
Anyadir
fichero escrito
maria@1smx:/home/pepe/facturas$
```



Anexos .

Codigo para manipular ficheros "manipular.cc". copia y pega el contenido a un fichero llamado "manipular.cc" y compílalo con la siguiente línea:

```
g++ -o manipular manipular.cc
```

Esto generará un fichero ejecutable llamado "manipular" que podrás usar en las actividades de después.

```
#include <stdio.h>
//#include <cstring>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>
// este programa lee, escribe (de nuevo) o añade líneas en un fichero para probar
permisos
// opcion -d muestra el usuario qeu está lanzando el programa con getlogin_r()

void leer(const char * Filename){
```

```
  printf("leer\n");
  FILE *f = fopen(Filename,"r");
   if (f==NULL) {
       if (errno == EACCES )
       fprintf(stderr,"No tienes permiso \n ");
       else
       fprintf(stderr,"Error no controlado\n");
       return;
       }
  const int linesize = 1000;
  char line[linesize];
  int contador=0;
  while (fgets(line,linesize,f)) {
       contador++;
       printf(" contenido de la línea %d -> %s",contador, line);
  }
  printf("He Leído El fichero y tiene %d líneas \n",contador);
}

void escribir(const char *Filename){
  printf("Escribir\n");
  FILE *f = fopen(Filename,"w");
       if (f==NULL) {
       if (errno == EACCES )
       fprintf(stderr,"No tienes permiso \n ");
       else
       fprintf(stderr,"Error no controlado\n");
       return;
       }


  fprintf(f,"Fichero sobreescrito. unica linea.\n");
  printf("fichero escrito\n");

}
void anyadir(const char *Filename){
  printf("Anyadir\n");
  FILE *f = fopen(Filename,"a");
       if (f==NULL) {
       if (errno == EACCES )
       fprintf(stderr,"No tienes permiso \n ");
       else
       fprintf(stderr,"Error no controlado\n");
       return;
       }

  fprintf(f,"Línea añadida por operacion anyadir\n");
  printf("fichero escrito\n");

}

int main(int argc, char **argv) {
  int argOperacion  = 1;
  int argFichero    = 2 ;
  if (argc == 4 ) {
    const char *opcion_d  = argv[1];
       if (strcmp(opcion_d,"-d") != 0) {
       fprintf(stderr,"Uso: %s [-d] operacion(leer,escribir,anyadir) fichero
\n",argv[0]);
```

```
      exit(1);
    }
    else {
        char nombre[20];
        getlogin_r(nombre,20);
        printf("ejecutandose como usuario %s\n",nombre);
        argOperacion++; argFichero++; // desplazamos las posiciones esperadas de
argumentos
    }
  } else
      if (argc != 3 ) {
      fprintf(stderr,"Uso: %s [-d] operacion(leer,escribir,anyadir) fichero
\n",argv[0]);
      exit(1);
   }

  const char *operacion  = argv[argOperacion];
  const char *fichero = argv[argFichero];

  if (strcmp(operacion,"leer")==0)      leer(fichero);
  if (strcmp(operacion,"escribir")==0)  escribir(fichero);
  if (strcmp(operacion,"anyadir")==0)   anyadir(fichero);
  return 0;
}
```

Recopilación de la actividad realizada en clase

<profesor: la clase se puede dar haciendo esta actividad junto con la explicación teórica de este documento>

Estos son los pasos dados en clase. Repítelos en tu casa verificando a cada paso los cambios que puedas comprobar. ¡ Atención ! La actividad requiere que cada paso sea realizado posiblemente por un usuario diferente. Cambie al usuario indicado con el comando "su - *usuario*" cuando sea necesario. Si no se indica nada, root es quien hace cada acción

1. Crear al usuario "mortadelo" con el adduser.
2. Crear al usuario "filemon" con el comando
   `useradd -m -b "/home/filemon" -d "/home/filemon" -u 1002 -s /bin/bash -U filemon`
3. Crear grupo "agentes"
4. Añadir mortadelo y filemón a agentes
5. Hacer a filemon administrador del grupo agentes.

6. Entrar (con el comando su - desde un terminal ya abierto) como el usuario mortadelo

7. Crear un fichero y comprobar sus atributos. Observar el grupo del fichero creado, especialmente el dueño y grupo del fichero
8. Cambiar el grupo del fichero con el comando chgrp de mortadelo a agentes.
9. Cambiar el grupo primario de mortadelo a agentes.
10. Vuelve a crear otro fichero, y comprueba su grupo.
11. Cambia el grupo de este último archivo a mortadelo

(volver a ser root, comando "exit" en el terminal)

12. Añadir al usuario pistolez

13. Entrar como filemón al terminal y añadir a pistolez al grupo agentes. Verificar que filemon puede añadir usuarios al grupo que administra

14. Resumen: " Crear una carpeta llamada /expedientes, el dueño filemon y el grupo será agentes".

    a. La carpeta sólo la puede crear root, por tanto el dueño y grupo será root. Verifíca el dueño y grupo después de crear la carpeta.
    b. root debe ceder la propiedad de la carpeta creada a filemon con el comando chgrp
    c. filemon después ya podrá cambiar de grupo de la carpeta a agentes

15. Filemón va a empezar a crear archivos con los expedientes para que los lea cualquier miembro del grupo "agentes", pero El súper, decide expulsar a "pistólez" del grupo agentes y lo manda a dirigir el tráfico

    a. Como filemón, expulsa del grupo agentes al usuario pistolez

16. Filemón crea un fichero su primer expediente en un fichero llamado "`ficherosecreto`" en la carpeta `/expedientes` con el contenido "`sospechamos que pistolez es un agente infiltrado, por ello el super le ha apartado`".

    a. lista el fichero y observa quién es el dueño y quién es el grupo dueño del fichero.
    b. Verifica los atributos del fichero mediante el comando ls -l. intenta ver qué permisos se tendrán sobre él, tanto filemón (dueño del archivo), como mortadelo (que pertenece al grupo agentes), como pistólez (que no debería poder acceder.

17. Como pistolez, intenta realizar las siguientes acciones:

    a. Situándote en la carpeta raíz, lista los contenidos de la carpeta expedientes con ls -l . Deberías poder ver en el listado, el fichero "ficherosecreto" creado por filemón

b. Entra a la carpeta /expedientes y sitúa allí tu directorio actual
c. Lee el contenido del fichero secreto anterior

Como se puede ver, pistolez ha podido realizar las tres acciones, ninguna de ellas deseable. Explícate por qué

18. Filemon debe impedir el acceso de pistolez al contenido del fichero. Pero no lo tiene muy claro, así que va a intentar eliminar permisos y verificar si pistolez puede acceder. Para cada una de las siguientes acciones, intente repetir los pasos del punto anterior siendo tanto pistolez (no debería poder) como mortadelo (sí debería poder, pero quizá no ahora mismo)

    a. Empieza quitando permisos de lectura para "otros" sobre el fichero creado. Después de probar a acceder como pistolez y como mortadelo restaura el permiso quitado
    b. Quita permisos de lectura sobre la carpeta /expedientes.
        i. Verifica que pese a todo pistolez puede ejecutar con éxito el comando
        `cat /expedientes/secreto`
        ii. Verifica igualmente que no puede listar el directorio /expedientes. Es decir no puede ejecutar el comando:
        ls /expedientes

        Finalmente, restaura el permiso eliminado
    c. Ahora quita permisos de ejecución "x" sobre el directorio sobre otros. Verifica los intentos de acceder. del punto anterior, tanto con mortadelo como con pistolez
19. Como se ha visto, a veces mortadelo no puede acceder al fichero porque éste tiene como dueño a "filemon" y también como grupo a "filemon". mortadelo está accediendo como usuario "otro" y si se limita a pistolez, también se limita a mortadelo.

    a. Para que mortadelo pueda acceder al archivo, cambia el grupo del fichero a "agentes" así mortadelo se verá afectado por los permisos del grupo y no de "otros"
    b. Como consecuencia, sólo los miembros del grupo "agentes" podrán acceder al fichero

20. Filemón hace otro expediente sobre un bandido en un fichero llamado "Sanchez".

    a. Para que pistolez no pueda acceder, le cambia el grupo al archivo "ficherosecreto" a "agentes" y quita los permisos a los otros igual que en el punto anterior para evitar que ofelia se inmiscuya

b. Mortadelo es un chapuzas de cuidado y a veces fastidia los expedientes escribiendo tonterías. Haz la comprobación de que mortadelo puede escribir datos en este nuevo expediente
c. Filemón quiere impedir que mortadelo altere ese expediente, pero no quere impedir que lo lea. Cambie los permisos del fichero y compruebe que se cumplen las restricciones ahora establecidas
d. dd

21. Filemón decide que no es correcto que pistolez pueda entrar a ver el nombre de los ficheros existentes en el directorio (pese a que no tiene acceso al contenido de los mismos).

    Por ello:

    a. Restaura los permisos del ficherosecreto para que otros puedan acceder y leerlo (ahora ¿Podría leer ofelia este fichero?).
    b. Cambia los permisos del directorio /expedientes para que otros no puedan acceder. hazlo de la siguiente manera:
        i. Quita los permisos para atravesar el fichero /expedientes. Comprueba que pistolez no puede acceder al fichero, pero sí que puede realizar "ls /expedientes" y ver qué ficheros hay allí.
        ii. Ahora restaura el permiso de atravesar el directorio ("x") y quita el permiso de lectura sobre el directorio. Verifica que pistolez no puede listar los contenidos del directorio. Verifica que puede visualizar el contenido de /expedientes/ficherosecreto.

22. Crearemos una nueva usuaria llamada "ofelia", que sin ser una agente y por tanto no pertenecer al grupo, va a trabajar en la organización y ocasionalmente colaborará con los expedientes

23. Vamos a permitir que ofelia, sin ser añadida al grupo agentes, pueda acceder a "ficherosecreto" gracias a que temporalmente, sí que será del grupo.

    a. Como ofelia, entra en el sistema y ejecuta la orden "newgrp agentes". Esto modificará el estado de la sesión iniciada y ofelia será durante esa sesión miembro del grupo "agentes". (recuerda que el grupo tiene una contraseña… si no la recuerdas, vuélvela a crear)
    b. Para que la orden tenga éxito, ofelia debe escribir correctamente la contraseña de acceso al grupo (que no ha sido establecida).
    c. Como usuario filemon, intenta ejecutar el comando "gpasswd agentes" para establecer la contraseña de acceso al grupo.
    d. Repetir el intento de acceso al ficherosecreto

24. Como ofelia tiene demasiados privilegios (Sabe la contraseña y es una peligrosa cotilla). Filemón va a hacer un truco para permitir que Ofelia acceda sólo a ciertos expedientes y no pueda fisgonear.

    a. Filemón creará un archivo llamado "expedientePaOfelia".

    b. Filemón Cambiará la contraseña del grupo, así ofelia ya no puede acceder porque la contraseña vieja no funciona.

    c. Para que ofelia acceda debe habilitar los permisos para otros en la carpeta expedientes, pero ¡Y aquí viene lo bueno! No se va a permitir a ofelia **listar** los archivos que existen en dicha carpeta. TAn sólo va a poder entrar y leer el fichero anterior "expedientePaOfelia"

    d. Entrar como ofelia y verificar que puede accederse al fichero :

    ```
    cat /expedientes/expedientePaOfelia
    ```

    e. Verificar que ofelia no puede hacer "ls /expedientes"

A partir de aquí es el bit **setuid** ¡ Sólo si se puede compilar !

25. Filemón va a controlar de otra manera el acceso al directorio agentes.

    a. En principio quitará los permisos a otros y al grupo (de esta manera ni sabiéndose la contraseña del grupo se puede entrar)
    b. Seguidamente filemón proporcionará un programa especial para que quien quiera acceder a la carpeta lo pueda hacer con su programa.
    c. En nuestro caso el programa es manipular.cc que una vez compilado puede llamarse manipular. Copia el programa en la carpeta /home/filemon y deja los permisos como están para que ofelia pueda ejecutar dicho programa (rwxr-xr-x)

26. Es la hora de que Ofelia utilice el programa de filemón para acceder a los ficheros de la carpeta /expedientes. Por tanto, como Ofelia...

    a. ejecuta el programa pasándole la orden de leer y el fichero a leer
    `/home/filemon/manipular leer /expedientes/ficherosecreto`
    b. Como verás, Ofelia no ha podido acceder al fichero usando ese ejecutable porque pese a que el ejecutable es de filemon, el ejecutable se ejecuta con el usuario de ofelia, que no tiene permisos en esa carpeta

27. El acceso a la carpeta /expedientes cuyos permisos son `rwx------` parece imposible para alguien que no sea el dueño (filemon). Pero queda un recurso. probémoslo

    a. Como filemon, ve al fichero ejecutable "manipular" y cámbiale los permisos de la siguiente manera

    ```
    chmod u+s manipular
    ```

    b. Este cambio de permisos activa un elemento llamado "bit setuid",

28. Como ofelia, vuelve a lanzar el mismo comando que antes
    a. `/home/filemon/manipular leer /expedientes/ficherosecreto`
    b. Observa que ahora sí que puedes leer el fichero. Repasa las notas sobre el bit setuid y cómo el ejecutable que ha lanzado ofelia se ha ejecutado con los privilegios de filemon.
29. Repetiremos anterior, pero eliminando el bit setuid y estableciendo el bit setgid. Es decir vamos a elevar privilegios al ejecutar ese ejecutable especial, pero esta vez el proceso se lanzará con el gid del grupo del ejecutable.
    a. Entra como filemon y cambia el grupo del ejecutable manipular a "agentes". Ahora el fichero sigue siendo de filemon, pero el grupo es agentes
    b. Ve a la carpeta de /expedientes y verifica que pertenece al grupo "agentes"
    c. Filemon quita permisos a sí mismo a la carpeta pero , permite permisos al grupo. Por lo que los permisos de la carpeta quedan ---rwx---.
    d. Como ofelia, vuelve a lanzar el ejecutable anterior, observa que sí puedes leer. El proceso lanzado se ejecuta con el GUI de agentes.
30. A partir del ejercicio anterior, haz la siguiente prueba.
    a. filemon añade el bit setuid al fichero manipular
    b. Ofelia intenta otra vez ejecutar el mismo comando. ¿Por qué no puede?
31. d
32. d
33. d
34. d

Posix Acl

https://www.usenix.org/legacy/publications/library/proceedings/usenix03/tech/freenix03/full_papers/gruenbacher/gruenbacher_html/main.html