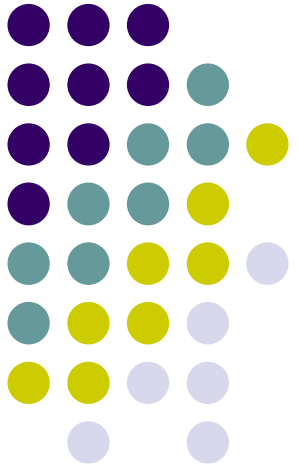
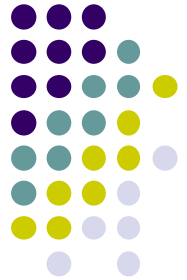
Gestión de la información y configuración de sistemas linux

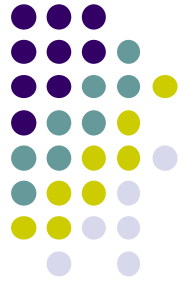
Unidad de trabajo 7



Índice



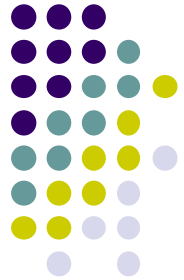
Índice



1. Introducción

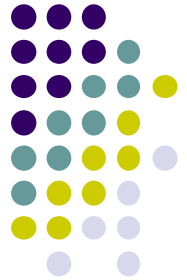
- Linus Torvalds empieza su desarrollo a principios de la década de los noventa
- Está inspirado en Minix (pequeño sistema UNIX) publicado por Andrew Tanenbaum
- La primera versión oficial (0.02) hace su aparición en octubre de 1991
- Aproximadamente en 1994 el núcleo es ya estable y el objetivo es añadir aplicaciones y utilidades para hacer el sistema más útil
- Entra en juego el proyecto GNU iniciado por la FSF conformando lo que hoy conocemos como GNU/Linux o simplemente LINUX

2. Características



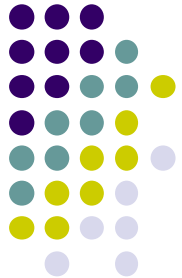
- Sistema Operativo Basado en UNIX
  - **Multitarea** : Se están realizando muchos trabajos a vez por debajo.
  - **Multiusuario**: Distintos usuarios con distintos permisos comparten la máquina.
  - **Sistema de Red** :Disponemos por defecto de distintos servicios de red para administrar.
  - **Sistema de Código Abierto**: Disponemos su código para el estudio y modificación.
- Es muy **eficiente y robusto**, la interacción con el usuario no ha sido algo prioritario.
- Es sensible a mayúsculas y minúsculas.
- En linux la estructura de directorios es única para varios dispositivos de almacenamiento. El separador de directorios es / .

2.1. Modo comando.



- Linux nos permite utilizar varias shell (intérprete de comandos): bash, sh , ksh, tcdh…
- Cada usuario tiene asignada (etc/passwd) la shell que utilizará, por defecto se suele utilizar la shell bash.
- $ suele ser el símbolo utilizado como prompt (indica que el SO está preparado para aceptar comandos).
- En el caso de que el usuario acceda como administrador (root) este signo se sustituye por #.

2.2. Sistema de archivos



- Existen 5 tipos principales de archivos
  - **Archivos ordinarios**. Contienen la información con la que trabaja cada usuario.
  - **Enlaces físicos (Vínculos físicos).**
  - **Enlaces simbólicos (Vínculos simbólicos).**
  - **Directorios**. Son archivos especiales que contienen referencias a otros archivos. Cuentan con información sobre archivos ordinarios, subdirectorios, vínculos, vínculos simbólicos, etc
  - **Archivos especiales**. Suelen representar dispositivos físicos como unidades de almacenamiento, impresoras, etc. Unix/Linux trata los archivos especiales como archivos ordinarios. De esta forma, un usuario puede abrir un archivo vinculado a una unidad de disquete, modificarlo, etc. Con ello consigue leer del disquete, escribir en el disquete, etc.

2.2. Sistema de archivos

2.2. Sistema de archivos

- La estructura de archivos es jerárquica siendo root (/) o raíz el padre de todos ellos.
- Los nombres de archivos:
  - Distinguen mayúsculas de minúsculas.
  - Pueden contener cualquier carácter salvo / y \0.
  - No es recomendable usar caracteres interpretados por la shell ($,”,’,&,#,(,),*,[,],{,}, etc)
  - No puede empezar por –
- En cada directorio hay como mínimo 2 entradas:
  - . Directorio actual
  - .. Directorio padre

3.1. Comandos sobre directorios

- **pwd**
  - **pwd** muestra la ruta del directorio actual
- **cd [ruta][directorio]**
  - **cd** nos lleva al directorio personal del usuario
  - **cd [ruta][nombre directorio]** accede al directorio situado en la ruta especificada.

Rutas absolutas

cd /etc

cd /home

cd /home/asir1

Rutas relativas

cd etc

cd home

cd asir

cd home/asir1

3.1. Comandos sobre directorios

- **ls [parámetros][ruta][directorio]**
  - ls lista el directorio actual
  - **ls [ruta][directorio]** lista el contenido del directorio situado en la ruta especificada
  - Parámetros
    - **-l** nos da información detallada sobre los archivos permisos, nº enlaces, propietario, grupo, tamaño, fecha, nombre
    - **-a** nos muestra todos los archivos incluso los ocultos
    - **-h** muestra los tamaños de forma legible (p.e. 1K 234M 2G)
    - **-i** muestra el número de nodo-i de cada fichero
    - **-m** rellena el ancho con una lista de entradas separadas por comas
    - **-n** *--numeric-uid-gid* como -l, pero muestra los UIDs y GIDs numéricos
    - **-R** --recursive muestra los subdirectorios recursivamente
    - **-S** ordena los ficheros por tamaño
    - **-t** ordena por la fecha de modificación
    - **-r** --reverse invierte el orden, en su caso

3.2. Comandos sobre directorios

- **mkdir [opciones][ruta][directorio]**
  - **mkdir [nombre directorio]** crea un directorio en el directorio actual
  - **mkdir –p [ruta][nombre directorio]** si no existen los directorios intermedios, se creen también

mkdir fundamentos
mkdir –p iso/teoria

- **rmdir [opciones][ruta][directorio]**
  - **mkdir [opciones][nombre directorio]** sirve para borrar un directorio vacío.
  - opciones:
    - -i pide confirmación
    - -f elimina sin preguntar
    - -p borra todos los directorios vacíos de una ruta.

rmdir fundamentos
rmdir –p iso/teoria

3.2. Comandos de ficheros

- **touch archivo**
  - Si archivo no existe aún, lo crea con tamaño 0 y como propiedad de nuestro usuario.
  - Si archivo ya existe, actualiza la fecha de modificación

touch tema1

- **rm [opciones] nombre_fichero [nombre_fichero]**
  - -r borra de forma recursiva un árbol

rm tema1

rm fundamentos

- **cat nombre_fichero**
  - visualiza el contenido de un fichero
  - cat > nombre_fichero

cat >tema1

Contenido de tema 1

ctrl+C

cat tema1

3.2. Comandos de ficheros

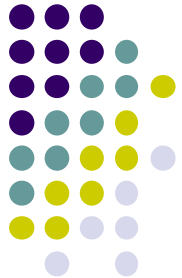
- **cp origen [origen2] destino**
  - Si destino es un fichero copia el origen1 con el nombre del destino.
  - Si destino es un directorio copia los ficheros origen en el directorio de destino

cp tema1 tema1R
ls
cat tema1R
cp tema1 iso
ls iso

- **mv origen destino**
  - Si destino es un fichero renombra el fichero
  - Si destino es un directorio mueve el fichero al destino

mv tema1R tema
ls
mv tema iso
ls
ls iso

5. Visualización de ficheros



- **more [opciones] nombre_fichero**
  - Para desplazarnos por el fichero:
    Avanzar página: **barra espaciadora**
    Avanzar una línea: **retorno de carro**
    Salir: **q**
  - **opciones**
    -d nos muestra esta información al final del texto
    -c borra primero la pantalla antes de escribir la siguiente
  - Este comando tiene un pequeño inconveniente solo se puede desplazar hacia delante

5. Visualización de ficheros

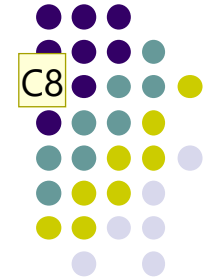
- **sort [opciones]... [nombre_fichero]...**
  - Muestra la concatenación ordenada de todos los FICHERO(s) en la salida estándar.
  - Opciones :
    - **-r** invierte el resultado de las comparaciones
    - **-c** comprueba si la entrada está ordenada o no
    - **-o** FICHERO escribe el resultado en FICHERO, en lugar de la salida estándar
    - **-u** ordena el fichero y no muestra líneas duplicadas
    - **- km** salta m campos hasta el comienzo de la clave
    - **-n** compara de acuerdo con el valor numérico de la cadena
    - **-tx** define el carácter separador de campos como x en lugar de un espacio en blanco

sort –t: -k3 –n /etc/passwd

5. Visualización de ficheros

- **wc [opciones]... [nombre_fichero]...**
  - Cuenta las líneas, palabras y caracteres dentro de un fichero de texto
  - Opciones :
    - **-l** muestra el número de líneas
    - -m muestra el número de caracteres
    - -w muestra el número de palabras

5. Visualización de ficheros



- **tail [opciones]... [nombre_fichero]...**
  - Muestra las últimas 10 líneas de cada FICHERO en la salida estándar.
  - Opciones :

    -nc muestra los últimos n bytes

    -n muestra las últimas n líneas en lugar de 10

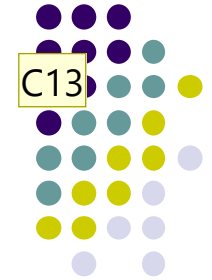
    -f
- **head [opciones]... [nombre_fichero]...**
  - Muestra las primeras 10 líneas de cada FICHERO en la salida estándar.
  - Opciones :

    -nc muestra los primeros n bytes

    -n muestra las primeras n líneas en lugar de 10

C8 Practica 4

Camino; 19/04/2011



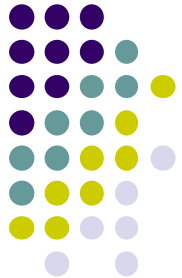
5. Visualización de ficheros

- **cut [opciones]... [fichero]...**
  - Extrae las partes seleccionadas de cada fichero en la salida estándar
  - Opciones
    - **-cLISTA** muestra solamente estos caracteres
    - **-dDELIM** usa DELIM en vez de caracteres de tabulación para delimitar los campos
    - **-fLISTA** muestra solamente estos campos; también muestra cualquier línea que no tenga un carácter delimitador, a menos que se especifique la opción -s
    - **-s** no muestra las líneas que no contienen delimitadores

C13 Practica 4

Camino; 19/04/2011

6. Otras órdenes



- **echo [opciones] mensaje**
  - Permite visualizar por pantalla cadena de caracteres, valores de variables y resultados de ejecución de ordenes
- **Clear**
  - Limpia la pantalla
- **cal [mes [año]]**
  - Muestra el calendario del mes y/o año seleccionado
- **time comando**
  - Muestra el tiempo de CPU y tiempo real transcurrido en la ejecución de una orden
    - real es el tiempo real de espera de un proceso >= user+ssy
    - user es el tiempo en modo usuario
    - sys es el tiempo en modo proceso

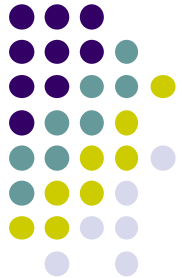
6. Otras órdenes

- **date**
  - Permite visualizar por pantalla cadena de caracteres, valores de variables y resultados de ejecución de ordenes
- **ln [-s] origen enlace**
  - Los enlaces le permiten dar a un único fichero múltiples nombres.
  - Los ficheros son identificados por el sistema por su número de inodo, el cual es el único identificador del fichero para el sistema operativo.
  - Un directorio es una lista de números de inodo con sus correspondientes nombres de fichero.
  - Cada nombre de fichero en un directorio es un enlace a un inodo particular.
  - Un **enlace duro** crea directamente el enlace al inodo
  - Un **enlace simbólico** permite dar a un fichero el nombre de otro, pero no enlaza el fichero con el inodo.

C11 Practica 6

Camino; 19/04/2011

6. Otras órdenes

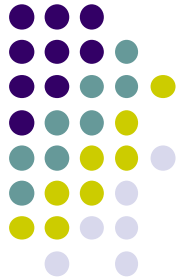


- **grep [opciones] patron archivos**
  - Localiza cadenas de texto dentro de ficheros.
  - Lleva dos argumentos: el primero es el patrón que se ha de buscar, y el segundo es una lista de nombres de archivos en los que se va a buscar el patrón.
  - Opciones
    - **–i** que ignora la diferencia entre mayúsculas y minúsculas,
    - **–n** donde cada línea viene precedida del número de línea
    - **–v** que sirve para mostrar las líneas que **no** tienen el patrón indicado
  - El patrón de búsqueda dentro del fichero puede utilizar los siguientes metacaracteres (o comodines):
    - **punto .** representa cualquier carácter en esa posición
    - **asterisco *** repetición del carácter anterior
    - **Ctrl ^** comienzo de línea
    - **$** final de línea

6.1. Redireccionar la E/S

- UNIX dispone de la capacidad de cambiar los archivos estándar de entrada (teclado), así como los de salida (terminal) por otros archivos diferentes.
  - **>** redireccionamiento de salida (para crear, destructivo).
  - **>>** redireccionamiento de salida no destructivo
  - **<** redireccionamiento de entrada.
  - **2>** redireccionamiento de mensajes de error (crea el fichero).

7. Gestión de seguridad



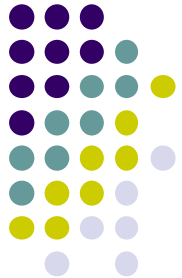
- Existen principalmente dos tipos de cuentas de usuarios:
  - Las cuentas de usuarios ordinarios
  - La cuenta de usuario root o superusuario
- Root es el encargado de realizar todas las tareas de administración, teniendo acceso a todo el sistema.
- Cada persona que utilice el sistema debe tener su propia cuenta.
- El sistema tiene un conjunto de información acerca de cada usuario.
  - **Nombre de usuario**
    - Es el identificador único dado a cada usuario del sistema.
  - **Identificador de usuario**
    - El user ID o UID es un número único dado a cada usuario del sistema.
    - El sistema normalmente identifica a los usuarios por el UID

7. Gestión de seguridad

- **Identificador de grupo:**
  - El group ID o GID es el identificador del grupo al que pertenece el usuario por defecto.
  - Cada usuario pertenece a uno o más grupos definidos por el administrador del sistema
- **Contraseña:**
  - la contraseña o clave del usuario se almacena de forma encriptada.
- **Nombre completo:**
  - El nombre real o completo del usuario se almacena junto con el nombre de usuario.
- **Directorio de trabajo:**
  - Cuando un usuario accede al sistema automáticamente entra en su directorio HOME..
- **Intérprete de comandos**:
  - cada usuario tiene asociado un intérprete de comandos con el que se conectará al sistema .
- La información anterior se almacena en un fichero llamado **/etc/passwd**:

nombre_usuario: clave: UID: GID: nombre_completo: directorio_personal: shell

7. 1. Administración de usuarios



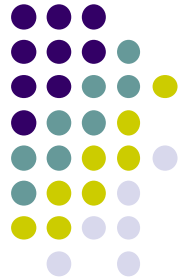
- **useradd** nombre_usuario **[opciones]**
  - Permite añadir usuarios
  - Opciones
    - -u identificador del usuario
    - -g grupo inicial del usuario
    - -d directorio personal
    - -s intérprete de comandos (por defecto, /bin/sh)
    - -G grupos a los que pertenece el usuario
    - -m crea el directorio personal
- **usermod [opciones] nombre_usuario**
  - Permite modificar la información de un usuario
- **userdel** [-r] nombre_usuario
  - Opciones
    - -r borraría el directorio personal del usuario

7. 2. Administración de grupos

- Un usuario puede pertenecer a uno o más grupos.
- La única importancia de los grupos es que permiten establecer **permisos de ficheros** para un grupo de usuarios.
- Cada fichero tiene un grupo propietario y un conjunto de permisos para las operaciones que pueden hacer los miembros del grupo con el fichero.
- Hay varios grupos definidos en el sistema, como pueden ser **bin**, **mail** o **sys**.
  - Los usuarios no deben pertenecer a ninguno de esos grupos; son grupos especiales que se utilizan para permisos de ficheros del sistema.
- Los usuarios pueden pertenecer al grupo que ya existe por defecto, o bien a nuevos grupos que creemos en el sistema.
- El fichero **/etc/group** almacena información sobre los grupos:

nombre_grupo: clave: GID: miembros

7. 2. Administración de grupos



- groups [user]
  - Si no se especifica un usuario, la orden nos muestra el grupo al que pertenecemos.
  - Si se especifica un usuario, la orden nos muestra el grupo al que pertenece el usuario
- groupadd nombre_grupo [opciones]
  - **Opciones**

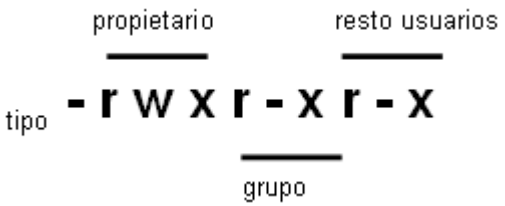
    -g asigna un identificador de grupo al grupo que estamos creando
- groupmod nombre_grupo
  - Modifica el grupo de usuarios
- groupdel nombre_grupo
  - Borra el grupo de usuarios indicado
- id [usuario]
  - muestra información del usuario y grupos a los que pertenece

7. 3. Permisos

- El UNIX es un sistema multiusuario, para proteger ficheros de usuarios particulares de la manipulación por parte de otros, UNIX proporciona un mecanismo conocido como permisos de ficheros.
- Los **permisos de ficheros** están divididos en tres tipos:
  - lectura (r): permite a un usuario leer el contenido del fichero
  - escritura (w): Permite modificar o borrar el fichero
  - ejecución (x): el intérprete de comandos entiende que ese fichero es un programa y puede ejecutarlo
- Los **permisos de directorios** están divididos en tres tipos:
  - lectura (r): permite listar el contenido del directorio, pero no permite acceder al mismo.
  - escritura (w): en este caso es posible crear ficheros y subdirectorios en el interior del directorio, y también se puede borrar. El propietario de un directorio siempre puede borrar los archivos contenidos en él, aunque sean de otro usuario.
  - ejecución (x): aplicado a un directorio, permite acceder a él.

7. 3. Permisos

- Cada uno de esos permisos puede asignarse a tres **clases de usuarios**:
  - el propietario del fichero
  - el grupo al que pertenece el fichero
  - el resto de usuarios del sistema
- El comando **ls–l** nos permite ver un listado largo de ficheros y directorios, en el que aparecen los permisos asociados.
- Los permisos se muestran en un campo de 10 caracteres.
  - El primer carácter no es exactamente un permiso, sino que indica el tipo de fichero (el guión indica un fichero normal).
  - El resto de permisos se dividen en tres grupos de tres caracteres cada uno.
    - 1º indica los permisos que el propietario del fichero tiene sobre él
    - 2º y 3º indican permisos para el grupo y el resto de usuarios.
  - Cuando se deniega alguno de los permisos, su carácter correspondiente se sustituye por un guión.



7. 3. Permisos

- chmod tipo_usuario símbolo tipo_permiso fichero
  - Donde tipo_usuario puede valer:
    - u (user) que es el propietario del fichero
    - g (group)que indica el grupo
    - o (other) resto de usuarios
    - a (all) todos los usuarios
  - Símbolo puede ser:
    - + añade permisos
    - - quita permisos
    - = asigna justo los permisos especificados
  - Y tipo_permiso puede valer:
    - r permiso de lectura
    - w permiso de escritura
    - x permiso de ejecución
  - Con la opción –R se cambian recursivamente los permisos de directorios y archivos.

Si tenemos un fichero con los siguientes permisos:

– r w – r – – r – –

Queremos quitar los permisos de lectura (-r) a todos los usuarios menos al propietario:

chmod og-r nombre_fichero

– r w – – – – – – – –

7. 3. Permisos

- Ejemplos
  - Dado un fichero llamado **programa** con los permisos –rw– r– – r– – y queremos que el shell lo identifique como ejecutable, dando permisos de ejecución a todos los usuarios, podemos hacer: **chmod a+x** programa
  - Dado un fichero llamado **fichero** sin ningún permiso y queremos darle al propietario y grupo permiso de lectura y escritura y al resto de usuarios sólo de lectura, podemos hacer: **chmod a+r,ug+w** fichero
  - Tenemos un directorio llamado **directorio** que es accesible por su propietario y por el grupo, pero no por el resto de usuarios, es decir, drwxr–x– – – . Queremos hacerlo accesible a todos los usuarios del sistema. Podemos copiar los permisos del grupo al resto de usuarios: chmod o=g **directorio**

7. 3. Permisos

- Existe otra forma de cambiar los permisos que se asignan a los ficheros. Se basa en el uso de un código octal para identificar cada uno de los tres tipos de permisos (lectura, escritura y ejecución).

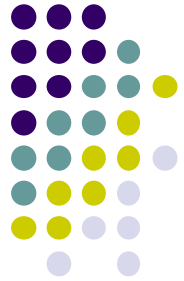
Usuario			Grupo			Otros		
r	w	x	r	w	x	r	w	x
4	2	1	4	2	1	4	2	1

- Se trata de formar un número de tres cifras en el que la primera cifra indica los permisos del propietario, la segunda los del grupo, y la tercera los del resto de usuarios del sistema.
  - Cada cifra se forma a su vez sumando los números correspondientes a los permisos que se quieran dar.

Usuario			Grupo			Otros		
r	w	x	r	-	x	r	-	x
4	2	1	4	-	1	4	-	1
4+2+1=7			4+1=5			4+1=5		

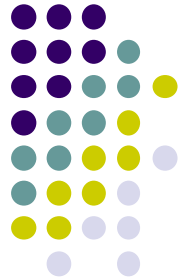
Si hacemos: **chmod 755 fichero**
Otorgamos permisos de lectura y ejecución a todos los usuarios del sistema, y de escritura al propietario del archivo.

7. 3. 1. Máscara



- Cuando se crea un archivo o un directorio nuevo por defecto se activan los siguientes permisos:
  - Archivo rw−rw−rw− en código octal corresponde al 444
  - Directorio rwxrwxrwx en código octal corresponde al 777
- Esto es consecuencia de la mascara del usuario, que es un número octal de tres dígitos.
- **umask** [num octal]
  - sin parámetros muestra la máscara del usuario
  - con parámetro modifica la máscara a la indicada en octal.
- umask funciona de la siguiente manera y en este orden ya que esta función no es conmutativa.
  - Al crear un archivo se aplica la función INH al permiso 666 y a la mascara
  - Al crear un directorio se aplica la función INH al permiso 777 y a la mascara

7. 3.1. Máscara



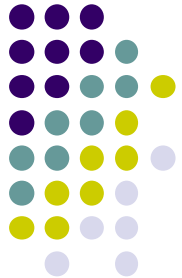
- La tabla de verdad de esta función es:

A	B	INH
0	0	0
0	1	0
1	0	1
1	1	0

Ejemplo con máscara 022

fichero	**directorio**
110 110 110	111 111 111
000 010 010	000 010 010
110 100 100	111 101 101
rw- r - - r--	rwx r-x r-x

7. 3. Permisos



- **chown** nombre_usuario nombre_fichero
  - Modifica el propietario de un fichero
- **chgrp** nombre_grupo nombre_fichero
  - Para asignar o cambiar un grupo a un fichero
- Consideraciones
  - El root tiene todos los permisos y derechos de acceso a cualquier archivo o directorio del sistema.
  - Los permisos de acceso se suelen configurar de forma que se tenga un control total sobre los ficheros propios, menor sobre los del resto de miembros del grupo, muy limitado sobre otros archivos de usuarios, y nulo o mínimo sobre los archivos del sistema
  - Los permisos son asignados o modificados exclusivamente por el propietario o el administrador.
  - Aunque eliminemos todos nuestros derechos sobre un archivo del que somos propietarios, podemos volver a recuperarlos con la orden **chmod**. El único caso en que perdemos definitivamente el control es al transferir su propiedad a otro usuario con la orden **chown**

7. 3. Permisos

- Consideraciones
  - Para ejecutar un archivo es preciso poseer permisos de ejecución, no sólo sobre dicho archivo, sino sobre todos los directorios de la ruta especificada para acceder a él.
  - Al transferir la propiedad de un archivo a otro usuario, conviene previamente mover o copiar dicho archivo a su directorio de trabajo u otro donde el nuevo propietario tenga los permisos pertinentes.
  - Un usuario puede pertenecer a varios grupos, pero en cada momento sólo tiene activada su pertenencia a uno de ellos.
  - El concepto de grupo asociado a un fichero se refiere a los miembros del grupo, excluido el propietario, que puede incluso no pertenecer al grupo. Análogamente, cuando hablamos del resto de usuarios, excluimos al propietario y a los miembros del grupo asociado al archivo.
  - Al crear un archivo, el propietario asignado es el usuario que lo crea, y el grupo, su grupo activo.
  - Si un usuario pretende cambiar su grupo activo por otro al que no pertenece, UNIX no le deja.