UNIDAD 4

Gestión básica de un SO. Usuarios, grupos y permisos

Sistemas Informáticos
1º de DAM Semipresencial
IES San Vicente 2020/2021

Index

Gestión de usuarios

Tanto Windows como Linux (entre otros) son sistemas operativos multiusuario, en los que más de un usuario puede tener cuenta y trabajar en el mismo sistema (incluso de forma simultánea).

Estos sistemas operativos deben proveer mecanismos para gestionar y controlar estos usuarios, la autenticación (login), permisos para programas, ficheros, etc. Los sistemas basados en Unix, como Linux, organizan los permisos por usuarios y grupos. Windows también lo hace así.

Al entrar en el sistema debemos identificarnos con un login y una contraseña. La contraseña original se guarda cifrada, de tal forma que cuando un usuario inicia sesión, la contraseña introducida se cifra (mismo algoritmo) y se comparan ambas cifradas. No puede conocerse de esta forma la contraseña original.

Usuarios en Linux

En las distribuciones Linux (y Unix) existe un usuario llamado **root** que tiene los privilegios máximos en el sistema, y que le permitirán realizar cualquier operación.

Muchas distribuciones, en la instalación desactivan el login como root, que podemos activar más tarde asignándole una contraseña con el comando **passwd**. Sin embargo, el usuario creado en el sistema puede usar el comando **sudo** delante de otro comando para ejecutarlo como si fuera el usuario root.

Sólo se recomienda usar la cuenta de root (o ejecutar comandos con sudo) para labores estrictamente administrativas, debiendo utilizar una cuenta de usuario normal para el resto de cosas.

El fichero /**etc**/**passwd** contiene toda la información relacionada con los usuarios (estos no tienen por qué ser usuarios con los que podamos hacer login sino que también podrían ser usuarios especiales que usen las aplicaciones para ciertas operaciones).

Cada línea es un usuario del sistema y se compone de 7 campos separados por dos puntos (“:”). Esto es lo que significa cada campo

- Nombre de la cuenta
- Contraseña cifrada (una x indica que se encuentra en el archivo /**etc**/**shadow**, un * indica que la cuenta está desactivada)
- ID del usuario (UID → número identificativo único)
- ID del grupo primario al que pertenece el usuario (GID) → ver /**etc**/**group**
- Información extra del usuario como el nombre completo
- Carpeta personal
- Shell (intérprete de comandos)

Más información: http://blogubuntu.com/chuleta-para-etc-passwd

Grupos en Linux

Los usuarios tienen siempre un grupo primario, y pueden tener cero o más grupos secundarios. La información de los grupos del sistema se encuentra en el archivo /**etc**/**group**.

El archivo /**etc**/**group** contiene la lista de los grupos del sistema y los usuarios que pertenecen a ellos (como grupo secundario, el primario está definido en /etc/passwd como hemos visto antes).

Cada línea se compone de 4 campos:

- Nombre del grupo
- Campo especial que suele estar vacío
- ID del grupo (GID)
- Lista de usuarios (separados por coma que pertenecen a ese grupo).

Comandos de gestión de usuarios y grupos (Linux)

- **useradd usuario** → Agrega un nuevo usuario al sistema. Existe una versión más sencilla en la mayoría de distribuciones: **adduser** usuario, que automatiza ciertos pasos como la creación de una carpeta de usuario, la asignación de contraseña o de un intérprete de comandos (/bin/bash por defecto).

- **passwd usuario** → Cambia el password a un usuario. Al crear un usuario necesita crear un password con este comando para poder usarlo.

- **groupadd grupo** → Agrega un nuevo grupo al sistema. Existe también el comando **addgroup**.

- **userdel usuario** → Borra el usuario del sistema. La opción -r borra también su directorio home con todo lo que contiene, y -f fuerza el borrado aunque aún esté conectado.

- **groupdel grupo** → Borra el grupo del sistema. No se puede borrar un grupo si es el grupo primario de algún usuario, así que habría que cambiarle el grupo primario a ese usuario o borrar dicho usuario primero.

- **whoami** → Informa de quién es el usuario actual en la sesión.

- **su usuario** → Inicia sesión como otro usuario. Con el comando exit se vuelve a la sesión anterior. Si no ponemos nombre de usuario, inicia sesión como root por defecto.

- **su – usuario** → Igual que el anterior pero el login es completo ejecutando una serie de scripts de inicio de sesión. Por ejemplo, nos mueve a la carpeta home del usuario, carga las variables de entorno, y la configuración personal del usuario. (-l y --login son opciones equivalentes a usar simplemente“-”).

- **usermod -g grupo usuario** → Cambia el grupo primario de un usuario por el definido.

- **usermod -G grupo1,grupo2,grupo3,... usuario** → Le asigna al usuario los grupos secundarios especificados sobreescribiendo los que tuviera antes. Si sólo queremos añadir nuevos grupos habrá que especificar la opción **-a** al principio.

- groups → Lista de grupos a los que pertenece el usuario actual (el primer grupo es el primario).

Permisos de archivos (Linux)

En Linux, todo archivo pertenece a un usuario y a un grupo (por defecto el grupo primario del usuario que lo creó) y se pueden cambiar mediante unos sencillos comandos

Por cada archivo se puedan otorgar permisos a 3 entidades diferentes:

- **Permisos de usuario**: Afectan al propietario del archivo

- **Permisos de grupo**: Afectan a los usuarios que estén dentro del grupo al que pertenece el archivo.

- **Permisos otros usuarios**: Afectan a los usuarios que no entran en la clasificación anterior.

Además, por cada tipo de usuario, hay 3 tipos de permisos que se pueden tener o no habilitados:

- **r (Permiso de lectura)**: Permite al usuario visualizar el contenido de un fichero. En un directorio se podrá visualizar la lista de ficheros que contiene.

- **w (Permiso de escritura)**: Permite al usuario modificar/eliminar un fichero. En un directorio permite crear y borrar ficheros.

- **x (Permiso de ejecución)**: Permite ejecutar un fichero ejecutable (binario o un script). En un directorio es necesario ya que sin este permiso no podremos acceder al directorio ni a su contenido, por lo que no podremos operar con nada de lo que contiene (no podemos ejecutar comandos sobre el directorio).

El usuario root tendrá prioridad sobre cualquier propietario de archivo, y no le afectarán los permisos establecidos pudiendo realizar cualquier acción.

Los permisos de cada fichero están representados mediante una cadena de 10 caracteres, y tiene la siguiente estructura:

Tipo	Usuario/Propietario			Grupo propietario			Otros		
-	r	w	x	r	w	x	r	w	x

El tipo de fichero puede ser:

- **-** : archivo normal y corriente.
- **d**: directorio
- **l**: enlace simbólico (o blando) a otro archivo
- **b**: archivo de bloques (discos duros, disqueteras, etc...)
- **c**: archivo de caracteres (terminales, puertos usb, serie, teclado, …)

Cuando consultamos los permisos de un archivo y aparece un guión "-" en alguno de ellos, significa que no tiene habilitado ese permiso.

Ejecutando el comando "**ls -l**" en un directorio podremos ver los permisos de cada fichero y varios atributos más. Ejemplo:

```
$ ls -l
total 4
drwxr-xr-x   2 arturo users 48 ene 26 12:16 dir1
lrwxrwxrwx   1 arturo users  5 ene 26 12:16 enlace1 -> fich1
-rw-r--r--   1 arturo users 28 ene 26 12:16 fich1
```

1ª columna. Muestra los atributos de cada fichero (tipo y permisos)

2ª columna. Muestra el número de enlaces que tiene un fichero o directorio.

3ª columna. Muestra el propietario del fichero.

4ª columna. Indica a qué grupo pertenece el fichero. Cuando se crea, este grupo coincide con el grupo primario del usuario que lo creó.

5ª columna. Es el tamaño en bytes del fichero o directorio.

6ª columna. Es la fecha y hora de la última modificación.

7ª columna. Es el nombre del fichero o directorio. En el caso de un enlace también indica el fichero que enlaza.

Existe una gestión de permisos más avanzada en Linux mediante listas de control de acceso o ACLs. Sin embargo no las vamos a cubrir en este tema limitándonos a la gestión de permisos básicos, que suele ser suficiente para la gran mayoría de usos.

Comandos de gestión de permisos (Linux)

- **chown**: Cambia el propietario de uno o varios archivo (sólo lo puede hacer el usuario root).
  - chown [opciones] propietario archivo1 archivo2 ...
  - Opción "-R": si alguno de los ficheros afectados es un directorio, cambia también los propietarios de los archivos que contenga dentro.
- **chgrp**: Parecido al anterior pero sólo cambia el grupo al que pertenece el archivo.
  - chgrp [opciones] grupo archivo1 archivo2 ...

- Con el comando chown podemos cambiar también el grupo de los archivos simplemente añadiendo ":" y el nombre del nuevo grupo al usuario. chown usuario:grupo archivo

- Opción "-R": si alguno de los ficheros afectados es un directorio, cambia también los propietarios de los archivos que contenga dentro.

- **chmod**: Cambia los permisos asociados a un archivo. Una manera de hacerlo es asociando una máscara de 3 bits (del 0 al 7 en decimal) a cada entidad (propietario, grupo, otros), de la siguiente manera:

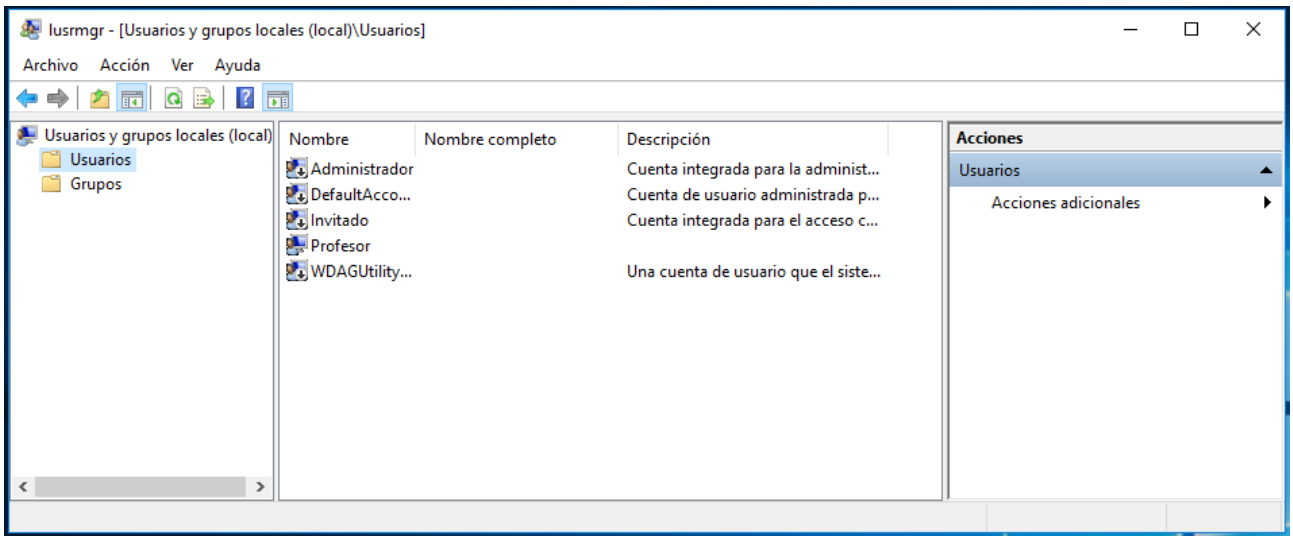
Propietario	Grupo	Otros
111	111	111
7	7	7

  - Los 3 bits se corresponden con lectura, escritura y ejecución. Al comando chmod hay que indicarle del 0 al 7 los permisos que se le concederán a cada entidad. Cuando un bit es 0, no se tiene el permiso, y cuando es 1 si se tiene.
    - Ejemplo: chmod 754 archivo.
    - Propietario: 7 (111 -> rwx, lectura, escritura y ejecución)
    - Grupo: 5 (101 -> r-x, lectura y ejecución)
    - Otros: 4 (100 -> r--, sólo lectura).
- Otra forma de de usar **chmod** es la siguiente:
  - chmod (ugo)(+-=)(rwx) archivo
    - u (modifica los permisos del propietario), g (grupo), o (otros).
    - + (añade los permisos que se especifiquen), - (los elimina), = (añade los permisos, pero eliminando antes los que ya tuviera asignados).
    - r (lectura), w (escritura), x (ejecución).
  - Ejemplo: **chmod go+wx**. Añade los permisos de escritura y ejecución al grupo y a otros usuarios. Se puede separar por comas las asignación de permisos en lugar de ejecutar varias veces el comando: chmod g+wx,o+x
  - Opción "-R": si alguno de los ficheros afectados es un directorio, cambia también los propietarios de los archivos que contenga dentro.

Gestión de usuarios y grupos (Windows)

Podemos acceder a la gestión de usuarios y grupos ejecutando el comando "**lusrmgr.msc**", bien desde la consola o CMD, presionando tecla de Windows + R y escribiendo el comando a continuación, o directamente escribiendo en la barra de búsqueda.

La creación de usuarios y grupos es bastante sencilla, simplemente pulsando con el botón derecho del ratón y seleccionando la opción usuario nuevo o grupo nuevo. Se puede agregar o quitar un usuario de un grupo accediendo a las propiedades de dicho usuario, en la pestaña "**Miembro de**".



Comandos de gestión de usuarios (Windows)

Para realizar estas acciones deberemos tener iniciado el terminal con privilegios de administrador (cmd → botón derecho, ejecutar como administrador)

Añadir un usuario desde la línea de commandos:

net user "Usuario" /add

net user "UserName" "Password" /add

Borrar un usuario

Net user "Usuario" /delete

Consultar los usuario de un grupo

net localgroup grupo // (invitados, usuarios, administradores, ...)

Como añadir a un usuario de grupo

net localgroup grupo "Usuario" /add

Como borrar a un usuario de grupo

net localgroup users "Usuario" /delete

Permisos de archivos (Windows)

En NTFS (sistema de archivos de Windows) un fichero posee los siguientes atributos y control de permisos:

- SID del propietario → Quien crea el archivo. Se puede cambiar.

- Lista de control de acceso de protección (ACL) → Permisos que los usuarios tienen sobre el fichero.

- Lista de control de acceso de seguridad (SACL) → Qué acciones sobre el archivo debe auditar el equipo (guardar un registro de acciones y usuario que las hizo)

Cuando se crea un archivo/carpeta esta hereda los permisos de la carpeta padre.

Sólo un usuario con control total sobre el archivo (propietario, administrador,...) puede cambiar los permisos de un archivo. Cuando se establece un permiso en una carpeta se puede elegir si todo lo que hay dentro hereda ese permiso.

Al copiar un archivo, el archivo copiado se considera nuevo, así que se le establecen los permisos desde 0 (los de la carpeta padre).

Al mover un archivo:

- Si va a la misma unidad se le mantienen los permisos que tuviera.
- Si es en otra unidad, igual que en la copia.

A los permisos de un archivo se accede desde botón derecho → propiedades → seguridad. Los permisos asignados o denegados (explícitos) sustituyen a los heredados (permisos por defecto del archivo).

- **Lectura** → Sólo leer los datos sin poder alterar, ni ejecutar nada
- **Mostrar el contenido de la carpeta** → Es igual que el de antes (sólo se puede asignar a una carpeta)
- **Lectura y ejecución** → Se pueden ver y leer los archivos, y se puede ejecutar el archivo si es un ejecutable.
- **Escritura** → Permite crear y modificar archivos. No se pueden eliminar archivos. Tampoco implica lectura ni ejecución.
- **Modificar** → Implica lectura, escritura y ejecución. No se pueden cambiar permisos ni cambiar el dueño de los archivos.
- **Control total** → Permite realizar cualquier operación.

El propietario de un archivo (usuario creador), tiene control total sobre el mismo y sus permisos, pero no puede trasferir la propiedad del archivo.

Toma de posesión por parte de otro usuario:

- Propiedades → Seguridad → Opciones avanzadas
- Se debe tener activada la opción de incluir los permisos heredables (los que obtiene de la carpeta padre)
- Ahora hay que seleccionar al usuario que tomará posesión, y activarle el último permiso → ”Tomar posesión”
- Ahora desde la pestaña ”Propietario”, se podrá cambiar el dueño del archivo.

Cuando un usuario o un proceso (que suele actuar con los permisos del usuario que lo lnazó), realiza una acción sobre un conjunto de archivos, el sistema comprueba que tiene permisos para hacer esa operación con todos ellos. Si no, aparece un mensaje de error.

Si un permiso no se ha definido para un fichero, siempre se presupone como prohibición. Hay que conceder permiso explícitamente.

Los permisos explícitos (establecidos a posteriori), tienen prioridad sobre los heredados (de la carpeta padre). Dentro de estos, un permiso negativo (prohibición) tiene prioridad sobre uno positivo, si afectan a lo mismo.

Otros comandos de gestión básica

Tanto en Windows como en Linux hay cientos de commandos que nos permiten realizer tareas ordinarias y de administración del Sistema desde la línea de comandos. Tenéis más información sobre commandos en documento sobre comandos disponible en el aula virtual.

Además de los commandos ya vistos vamos a revisar alguno de los commandos más utilizados dentro del Sistema operativo Linux.

- logout. Nos permite salir de la session actual
- reboot. Reinicia el Sistema.
- shutdown . Apaga el sistema

cat

Permite mostrar el contenido de un archivo → **cat archivo**. Si el archivo es muy grande se recomienda usar **cat archivo | less**, para verlo de forma paginada (ya aprenderemos para qué sirven los pipes o filtros "|" en los comandos).

find

Busca ficheros en la estructura de directorios, que cumplan una serie de condiciones. Si no se especifica la ruta de búsqueda, empezará desde el directorio actual.

find [RUTA] [OPCIONES]

-name XXX → Busca en fichero con el nombre especificado

-perm /XXX → Busca los archivos con los permisos especificados. Ejemplo: **-perm /220** (archivos que pueden ser escritos por propietario y grupo). Sería equivalente a usar **-perm /u+w,g+w**.

-user XXX → Busca los archivos de un determinado usuario

-exec comando{} +; → Ejecuta un comando con cada fichero que encuentra.

Ej.- find -exec rm{} +;

-mtime días → Buscar archivos que fueron modificados hace más de x días o menos, dependiendo el signo utilizado delante del número. Ejemplo: -mtime -2 → Archivos modificados en los últimos 2 días (2 días o menos)

grep

Busca un texto patrón en cada línea de un fichero de entrada.

grep [OPCIONES] patrón [FICHERO_ENTRADA]

-w → Busca cadenas completas y no subcadenas, es decir la cadena patrón debe estar precedida y finalizada por blancos o fin de línea.

Cuenta las líneas del fichero en las que se encontró la cadena patrón.

-i → No diferencia mayúsculas de minúsculas.

-r → Busca de forma recursiva

-n → Muestra el número y la línea completa que contiene el patrón.