Usuarios y Grupos en Linux

Usuarios

El concepto de usuario en Linux permite separar entornos de ejecución para diferentes propósitos. Dos personas pueden trabajar simultáneamente en el mismo sistema, teniendo cada uno un usuario diferente, y un directorio personal diferente.

También es muy común que muchos servicios internos del sistema tengan su propio usuario para restringir el acceso de ese servicio como mecanismo de seguridad. De este modo, si un servicio ve su seguridad comprometida por un ataque, el acceso que tenga el

usuario de ese servicio servirá como contención del ataque, y no podrá acceder a ficheros pertenecientes a otro usuario (de persona o servicio).

> ***Demonio (daemon)*** *es el término usado en Linux para referirse al proceso de un servicio que se ejecuta en segundo plano de forma no interactiva. En general acaban por la letra* ***d****, como* ***httpd*** *o* ***ftpd****.*

La configuración de usuarios en Linux esencialmente se maneja en los dos siguientes ficheros.

Archivo `/etc/passwd`

Contiene información de las cuentas de usuarios y sus características.

> *name:password:UID:GID:GECOS:directory:shell*

Se pueden ver los campos con `man passwd` :

Login	Nombre del usuario
Password	El password del usuario en texto plano, o un asterisco `*` o una `X` si está encriptado.
UID	**User ID**. Número de identificación único de usuario. Los usuarios pueden cambiar muchos prámetros, incluso su `name` , pero el UID no lo deben cambiar nunca. El UID del root es `0` . Las cuentas de servicios y demonios tienen los números más bajos, mientras que las de usuarios finales comienzan en el valor definido en `UID_MIN` en el fichero `/etc/login.defs` .
GID	**Gruop ID**. Número de identificador único de grupo. Varios usuarios pueden tener el mismo grupo, aunque al crear un usuario se crea un grupo con ese

	mismo nombre por defecto salvo que se indique lo contrario. Los datos del grupo aparecen en `/etc/group` .
GECOS	Campo de comentarios que incluye información extra sobre el usuario (nombre real, dirección…) Informalmente se le llama información *finger*.
directory	**Home directory**. Directorio de inicio del usuario. Los usuarios finales se suelen situar bajo `/home` .
Shell	La shell que utiliza por defecto el usuario (en muchos casos es `/bin/bash`). Si el usuario tiene `/sbin/nologin` o `/usr/bin/false` , significa que no tiene permiso para loguearse en el sistema, lo cual es común en *daemons* como medida de seguridad.

Por ejemplo:

```
root:x:0:0:root:/root:/bin/bash
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
avahi:x:115:120:Avahi mDNS daemon,,,:/var/run/avahi-daemo
n:/bin/false
usuario:x:1000:1000:usuario,,,:/home/usuario:/bin/bash
```

Archivo `/etc/shadow`

Contiene información sobre contraseñas de los usuarios en `/etc/passwd` , que almacena de manera cifrada.

Login	Nombre del usuario
Password	Password encriptado. La función *hash* usada para encriptar el password se indica al comienzo.
Lastmod	Tiempo transcurrido desde el último cambio de clave.
Min	Número mínimo de días hasta que se puede volver

	a cambiar la contraseña.
Max	Número máximo de días hasta que el sistema obliga a cambiar la contraseña del usuario.
Aviso	Número de días previos al **Max** en los que el usuario es avisado de su obligado cambio de contraseña.
Inactividad	Número de días entre el vencimiento de la contraseña y el bloqueo de la cuenta.
Expiración	Fecha en la que la cuenta se deshabilita. Si se deja en blanco, la cuenta nunca expira.
Reservado	Campo reservado para futuros usos.

Sobre el campo del password:

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

Cuando se bloquea una cuenta (lock: `usermod -l user`), no se borra el password del usuario, sino que se añade una exclamación `!` al comienzo del hash del password para indicar que se ha bloqueado. Al desbloquear el usuario (unlock: `usermod -u user`) se elimina la exclamación, dejando el hash como estaba antes.

Comandos de gestión de usuarios

La Tool Suite Shadow es una colección de comandos que permiten gestionar usuarios sin necesidad de manipular los ficheros `/etc/passwd` y `/etc/shadow` directamente, lo cual no es recomendable ante la posibilidad de dejar inconsistencias en ficheros tan delicados. Algunos de los comandos:

- `useradd` – crea un usuario.
- `userdel` – borra un usuario.

- `usermod` – realiza modificaciones sobre los datos de `/etc/passwd`. Tiene una opción para cada uno de los campos, excepto el campo **GECOS**. Incluye opciones para (des)bloquear un usuario (`--lock` y `--unlock`).
- `chfn` – modifica la información *finger* (**GECOS**).
- `chsh` – modifica la shell.
- `id` – imprime información sobre el usuario y sus grupos. Ej:

```
$ id -u   # Imprime el UID
$ id -un  # Imprime el login
```

- `chage` – *change age* visualiza y modifica todas las fechas de contraseña de `/etc/shadow` .

También existen los comandos `adduser` y `deluser` que son más *user-friendly* y ahorran algo de trabajo, por lo que se utilizan con más frecuencia en la práctica.

> ➜ *Usa el manual* `man` *o las opciones* `-h` *o* `--help` *para ver todas las opciones que ofrecen estos comandos, ya que pueden cambiar según distribuciones.*

Para establecer y cambiar una contraseña:

- `usermod -p PASSWORD USER` guarda el password indicado sin encriptar (habría que pasarle un hash del password). No conviene usar esta opción.
- `passwd USER` es el comando que se utiliza, que permite introducir un password con seguridad.

Con el comando `finger` podemos obtener información del **GECOS** de cualquier usuario:

```
$ finger theuser
Login: theuser                          Name: Juan Pérez
Directory: /home/theuser                Shell: /bin/bash
Office: 101, +34 123 456                Home Phone: +34 983
```

```
12 34 56
On since Wed Feb 04 10:26 (EST) on pts/1   3 seconds idle
     (messages off)
No mail.
No Plan.
```

Grupos

Los grupos permiten conceder permisos a un conjunto de usuarios simultáneamente.

En Linux un usuario tiene los siguientes grupos:

- **Grupo primario:** es el que consta como su *GID* en `/etc/passwd`. Sólo puede haber un grupo primario.
- **Grupos secundarios o suplementarios:** son los gestionados en el fichero `/etc/groups`, donde se puede añadir un usuario a más grupos.

Además, durante la sesión de usuario se puede cambiar temporalmente el grupo al que pertenece el usuario:

- **Grupo real:** es su grupo primario que consta en `/etc/passwd`. Es el grupo al que pertenece un usuario cuando inicia sesión.

- **Grupo efectivo:** mediante el comando `newgrp` se puede cambiar el grupo primario al que pertenece el usuario, y la configuración es efectiva hasta que cierre la seción o vuelva a cambiar de grupo efectivo.

Archivo `/etc/group`

Este archivo contiene información sobre los grupos del sistema. Su estructura es similar a la de los archivos `passwd` y `shadow`, contando con los siguientes campos:

Grupo	Nombre del grupo.
passwd	Password que permite a un usuario cambiar de grupo. Si está vacío no requiere contraseña, y una `x` significa que se gestiona mediante el archivo `/etc/gshadow` .
GID	*Group ID*. Identificador único (numérico) para el grupo.
Miembros	Lista separada por comas con los nombres de usuario que pertenecen a ese grupo.

De forma similar, al fichero `/etc/shadow` , existe el fichero `/etc/gshadow` , que almacena los passwords de los grupos encriptados con un *hash* y también trabaja con los símbolos asterisco `*` y exclamación `!` .

Comandos de gestión de grupos

La suite *Shadow* también incluye los comandos:

- `groupadd` – añade un nuevo grupo
- `groupdel` – borra un grupo
- `groupmod` – Modifica la información de `/etc/groups`
- `gpasswd` – Modifica el password del grupo, reflejado en `/etc/gshadow`

Modificar grupos de usuarios

Una vez tenemos un grupo creado, podemos añadírselo como primario o secundario a un usuario usando el comando `usermod` , con las siguientes opciones:

```
$ usermod --help
Usage: usermod [options] LOGIN

Options:
```

```
  ...
  -g, --gid GROUP               force use GROUP as new prim
ary group
  -G, --groups GROUPS           new list of supplementary G
ROUPS
  -a, --append                  append the user to the supp
lemental GROUPS
                                mentioned by the -G option
without removing
                                him/her from other groups
```

Por lo tanto:

```
$ # Modificar el grupo primario
$ usermod -g GROUP USER
$ # Reemplazar el grupo secundario
$ usermod -G GROUP USER
$ # Añadir un grupo secundario al usuario
$ usermod -a -G GROUP USER
```

Permisos sobre archivos y directorios

En Linux, cada fichero y directorio tiene permisos para:

- **Usuario**: el sistema de ficheros guarda un usuario propietario (*UID*), junto a los permisos que tiene asociados.
- **Grupo**: también se guarda un *GID* propietario, con permitos
- **Otros**: también tiene permisos para usuarios que no tienen ese *UID* ni ese *GID*.

El comando `ls -l` lista los archivos y directorios incluyendo información sobre sus permisos, de la siguiente manera:

```
$ ls -l
drwxr-xr-x 6 usuario grupo  4096 Jan  5 17:37 directory
-rw-r--r-- 1 usuario grupo  2048 Jul  6 12:56 file
```

El primer campo indica en la primera letra si es un archivo regular (`-`), directorio (`d`) o enlace (`l`). Tras ello, aparecen los permisos, organizados de **escritura**, **lectura** y **ejecución** para el *usuario*, *grupo* y *otros*.

> ➜ *El permiso de ejecución `x` en directorios significa que se puede acceder dentro del mismo y listar sus contenidos.*

Modificaciones de permisos

Los permisos de un archivo o directorio se pueden modificar con el comando:

```
$ chmod permisos archivo(s)
```

Modo relativo. Se puede tratar uno de los campos de forma aislada sin tocar el resto de los permisos:

- Se indica primero a quién se va a cambiar el permiso (se pueden poner varios):
  - `u` : usuario
  - `g` : grupo
  - `o` : otros
  - `a` : `all` , equivalente a `ugo` (también se puede dejar en blanco)
- Tipo de operación:
  - `+` : añadir permisos
  - `-` : quitar permisos
- Permisos:
  - `r` : lectura
  - `w` : excritura
  - `x` : ejecución

Ejemplos:

```
$ chmod u+x fichero
$ chmod go-x fichero
$ chmod +x fichero
```

Modo absoluto. Se puede reemplazar la información completa de los permisos utilizando un número en base 8 (octal) de tres cifras. Los permisos coinciden con los del número en binario. Lo bueno de trabajar en octal es que cada caracter se puede trabajar de manera independiente.

Ejemplo de permiso `754` :

- `7` en binario es `111` ⇒ permisos de *usuario*: `rwx`
- `5` en binario es `101` ⇒ permisos de *grupo*: `r-x`
- `4` en binario es `100` ⇒ permisos de *otros*: `r--`

```
$ chmod 754 file
$ ls -l file
-rwxr-xr-- 1 usuario grupo  2048 Jan  6 13:03 file
```

Máscara `umask`

La máscara del sistema operativo define los permisos que se asignan por defecto a archivos y directorios en el momento de su creación.

El comando `umask` sin parámetros imprime el valor que tiene actualmente, y se puede manejar en modo simbólico y modo octal:

```
$ umask
0022
$ umask -S
u=rwx,g=rx,o=rx
```

Si se pasa un parámetro al comando, se establece la nueva máscara.

Modo simbólico: permite establecer los permisos usando las letras

`u` (*user*), `g` (*group*), `o` (*other*) y `a` (*all*). Se puede hacer en modo relativo usando los símbolos `+` y `-`, o en modo absoluto con `=`, y combinaciones de ambos.

```
$ # Modo relativo
$ umask g-w
$ umask a+x
$ # Modo relatuvo (u,o) y modo absoluto (g)
$ umask u-w,g=r,o+r
```

Modo octal: permite establecer los permisos numéricamente, siempre en modo absoluto.

El modo octal se usa en base a los permisos máximos, que para directorios es `777` y para archivos `666` (sin ejecución). Los bits de la máscara que estén a 1 desactivarán ese permiso a los permisos máximos. Ej: `umask = 023`

```
Umask      023: 000010011
Directorio 777: 111111111 -> 111101100 = rwxr-xr--
Archivo    666: 110110110 -> 110100100 = rw-r--r--
```

Muchos sistemas tienen la máscara `022`, que se establece con:

```
$ umask 022
```

Cambio de propietario y de grupo

Estos dos atributos se pueden modificar con los comandos:

- `chown archivo nuevo_propietario` : modifica el propietario (*UID* asociado).
- `chgrp archivo nuevo_grupo` : modifica el grupo (*GID* asociado).

Con ambos se puede usar la opción `-R` en directorios, para que realice la operación de manera **recursiva**, es decir, que lo aplique a

todo lo que contiene directamente o en subdirectorios, sub-subdirectorios etc.

Sudoers

En Linux se pueden ejecutar determinados comandos como si fuesen el *root* con `sudo comando`. Si el usuario intenta ejecutar algo con `sudo`, se comprueba si tiene permisos para ejecutar el comando, y si no los tiene, no lo ejecuta y el intento queda registrado en el sistema.

Se puede configurar los privilegios de los usuarios para ejecutar ciertos comandos con el fichero `/etc/sudoers`. Las modificaciones sobre este archivo no se deben realizar como si fuese un fichero normal, sino que hay que modificarlo con el comando `visudo`.

Este archivo está dividido en varias secciones:

Definición de alias: Existen varios tipos de alias para realizar agrupaciones. Algunos de ellos son:

- `Cmnd_Alias`: define alias de comandos. Por ejemplo:

```
Cmnd_Alias CMND_RED = /sbin/ifup, /sbin/ifdown
```

- `User_Alias`: agrupaciones de usuarios.

```
User_Alias ADMINS = user1, user2
```

- `Host_Alias`: alias de hosts, que pueden indicarse por su nombre, IP o IP son máscara de subred.

```
Host_Alias MIEMPRESA = 172.26.0.0/16
```

Reglas de acceso: En esta sección se asignan a los comandos que pueden realizar los usuarios y desde qué hosts lo puede hacer.

Tiene el formato:

```
usuario host = (usuario_privilegiado) comandos
```

- **Usuario:** puede ser un usuario o alias de usuario. Indica a quién afecta la regla. Si comienza por `%`, entonces el nombre se refiere a un grupo del sistema.
- **Host:** indica desde qué host se permiten realizar esos comandos.
- **Usuario_privilegiado:** indica qué permisos de usuarios privilegiados se permiten al usuario. Este campo es opcional.
- **Comandos:** indica los comandos o alias de comandos que se permiten para el usuario.

Es necesario escribir los comandos usando su ruta completa. Para consultar la ruta de un comando puedes usar `which <nombre_comando>`. Los comandos más comunes se suelen localizar en:

`/bin`	comandos **esenciales** del sistema (`cat`, `ls` ...) accesibles desde una etapa muy temprana en el arranque del sistema. Son accesibles por todos los usuarios.
`/sbin`	lo mismo, pero que requieren permisos de superusuario.
`/usr/bin`	comandos **no esenciales**, pero de uso general instalados a nivel de sistema para todos los usuarios.
`/usr/sbin`	equivalente al anterior, pero que requiere permisos de superusuario.
`/usr/local/bin`	comandos que no forman parte de la distribución del SO.

Por ejemplo:

```
ADMINS MIEMPRESA = CMND_RED
```

Si se quiere permitir acceso completo a un usuario, se puede indicar con el alias especial `ALL`. Por ejemplo, el usuario `root` se suele configurar como:

```
root ALL=(ALL) ALL
```

No es necesario reiniciar el sistema una vez modificado este fichero con `visudo`. Una vez configurado, un usuario podrá ejecutar esos comandos permitidos con `sudo` usando su propia contraseña.