Administración/Configuración de Linux - Gestionando usuarios

Cubre el objetivo del Linux Professional Institute (LPI) 1.107.1

Tabla de Contenidos

1 Introducción

Linux es un **sistema multiusuario**. Una de las primeras tareas de **administración** debe ser la de los **usuarios del sistema**. En Linux existen dos tipos o perfiles de usuarios: el **administrador, o 'root'**, y los **usuarios finales**.

En todo sistema Unix existe un **usuario administrador (root)**, que controla el funcionamiento completo del sistema.

La administración de usuarios en Linux es la referida a la creación y configuración de las cuentas de usuarios, grupos de usuarios.

Este manual es relativo al sistema operativo Linux y requiere nociones esenciales de programación en BASH.

2 Usuarios y grupos

Linux es un sistema multiusuario que usa **cuentas de usuario** para identificar a los distintos usuarios del ordenador.

Una **cuenta de usuario** es un conjunto de estructuras de datos y procedimientos que contienen y gestionan toda la información necesaria para distinguir a dos usuarios distintos dentro del sistema.

3 Conceptos importantes

3.1 Nombres de usuario

Cada usuario de un sistema Linux tiene un **nombre de usuario único**. Es el que se usa para acceder al sistema. Es un nombre simbólico, con sentido para el administrador del sistema, que le facilita el trabajo, aunque internamente, un usuario se representa mediante un número, su **UID**.

Linux es muy **flexible sobre los nombres de usuario**. La mayoría de las versiones de Linux dan soporte para nombres de usuario que consisten en cualquier combinación de **letras** (manyúsculas y minúsculas), **números** y muchos de los **signos de puntuación** (incluyendo puntos (**.**), barra baja (**_**) y espacio ( )). Sin embargo, **se recomienda que como nombre de usuario se usen sólo letras y números** para evitar problemas con algunas utilidades.

- ***Todos los nombres de usuario deben comenzar obligatoriamente con una letra.*** Así el nombre de usuario 45u no está permitido, pero el nombre u45 sí.
- ***Un nombre de usuario puede contener hasta 32 caracteres.*** Sin embargo, dado que algunas utilidades truncan los nombres de usuario mayores de 8 caracteres, se recomienda usar nombres de usuario que como **máximo tengan 8 caracteres**.
- ***Linux distingue mayúsculas de minúsculas,*** por lo tanto, los nombres de usuario juan y Juan son distintos. Mezclar nombres en mayúculas y minúsculas puede dar lugar a confusión. Por tanto, se recomienda usar sólo **nombres con letras en minúscula**. Nombres típicos en un sistema Linux son: facr, fconde, fconde01, f_conde o f.conde para mostrar sólo algunas variaciones posibles.

Los usuarios actualmente definidos en el sistema se gestionan mediante un archivo denominado **/etc/passwd**, y en la mayoría de sistemas Linux actuales, sus contraseñas, mediante el archivo **/etc/shadow**.

3.2 Grupos

Linux usa los **grupos** como un medio para organizar a los usuarios. Hay que tener muy presente que los grupos no son cuentas, sino un medio de **organizar conjuntos de cuentas**.

La principal finalidad de los grupos es aumentar la seguridad del sistema. **Cada fichero** en un sistema Linux **está asociado con un grupo específico** y se pueden asociar permisos al grupo, de forma que todos los usuarios de ese grupo tienen esos permisos. Linux proporciona acceso a la mayoría de los dispositivos hardware a través de ficheros, así que se puede controlar el acceso de los usuarios al hardware, a través de los permisos que se asocian a esos ficheros.

Un grupo puede contener a todos los usuarios del sistema, a ninguno, o a cualquier cantidad intermedia de usuarios. La pertenencia de los usuarios a los grupos se controla mediante el archivo **/etc/group**. Este archivo contiene una lista de todos los grupos definidos en el sistema y los usuarios que pertenecen a cada grupo. **Un usuario puede** (de hecho lo hace) **pertenecer a más de un grupo**.

Cada usuario tiene un grupo por defecto o **grupo primario.** Cuando un usuario accede al sistema, se le asocia ese grupo primario, de forma que cuando crea un archivo o ejecuta un programa, ese archivo y ese programa están asociados a un único grupo, el grupo primario del usuario.

Esto no significa que el usuario no pueda pertenecer también a otros grupos. Así, por ejemplo, si otro archivo está asociado a otro grupo, al que también pertenece el usuario, éste puede acceder a dicho archivo, siempre y cuando el archivo tenga permisos que permiten el acceso para los miembros de su grupo.

Un usuario puede crear archivos o ejecutar programas que tengan asociado otro grupo distinto de su grupo primario. Para ello, primero debe elegir otro grupo ***de aquellos a los que pertenece*** con la orden **newgrp**.

> Suponiendo que existe un usuario llamado *fconde*, cuyo grupo primario es ***fconde*** (es lo habitual en Ubuntu), pero que también es administrador del sistema y por tanto también pertenece al grupo *adm*, la siguiente orden es correcta:

```
fconde@iMac21:~$ newgrp adm
```

A partir de ese momento, cada archivo que cree el usuario fconde ya no tendrá como grupo el grupo *fconde*, sino el grupo *adm*.

Si el usuario especifica en la orden **newgrp**, un grupo al cual no pertenece, se le pedirá una contraseña, la contraseña del grupo (más adelante se verá como establecer estas contraseñas).

Como hemos visto, los grupos son un mecanismo muy eficaz para permitir que un conjunto de usuarios trabaje sobre un conjunto de archivos, sin que el resto de usuarios de la máquina pueda acceder a ellos. Así por ejemplo, se puede establecer un grupo por cada clase, o por cada proyecto, o por cada departamento de una empresa.

Un usuario puede pertenecer a dos o más grupos, por ejemplo, un estudiante que pertenezca a varias clases o un empleado que trabaje en dos proyectos. De esta forma, tendrá acceso a los archivos de esos grupos según los permisos que se establezcan para ese grupo.

Un usuario **debe** pertenecer al menos a un grupo, su **grupo primario**, tan es así que ***si un usuario no pertenece a ningún grupo, no puede hacer login en el sistema***.

3.3 UIDs y GIDs

Como se ha mencionado anteriormente, los nombres de usuario y de grupo, son nombres simbólicos que ayudan en su tarea al administrador del sistema. Sin embargo, internamente, Linux define usuarios y grupos mediante números llamados **user ID (UID)** y **group ID (GID)** respectivamente.

Cuando un administrador usa un nombre de usuario o de grupo, el sistema Linux usa los archivos **`/etc/passwd`** y **`/etc/group`**, para localizar dicho nombre y averiguar cual es el UID o el GID asociado, y luego usa esos números al ejecutar órdenes.

En este apartado, vamos a estudiar cómo Linux asigna números a usuarios y grupos, lo que será muy util si hay problemas y sobre todo si se deben editar los archivos **`/etc/passwd`** y **`/etc/group`** manualmente.

Las distribuciones de Linux **reservan los primeros cien identificadores de usuario y grupo (0-99) para uso del sistema**. El más importante de todos es el ID 0 que se asigna al root (tanto el usuario como el grupo). El resto de IDs del sistema no están completamente estandarizados en todas las distribuciones de Linux. Así por ejemplo, el UID 2 y GID 2 que corresponden con el usuario y grupo *daemon* en Red Hat y SUSE, en Debian y Ubuntu corresponden con el usuario y grupo *bin*. Por eso es preferible **referirse a usuarios y grupos mediante su nombre simbólico** (que sí es igual en las distintas distribuciones Linux) que mediante su ID (que puede variar).

A partir del identificador 100, esos IDs están disponibles para usuarios y grupos ordinarios, aunque la mayor parte de distribuciones **reservan hasta el ID 500 o incluso hasta el 1000 (como es el caso de Ubuntu) para propósitos especiales**. Por lo tanto, al primer usuario "normal" del sistema se le asigna el ID 500 o 1000 (1000 en el caso de Ubuntu). Estos límites se definen en el archivo ***/etc/login.defs***, en concreto en las constantes UID_MIN, UID_MAX y GID_MIN, GID_MAX.

Cuando se crean nuevos usuarios, por lo general, el sistema localiza el siguiente ID más alto no utilizado, así por ejemplo, la segunda cuenta de usuario que se cree tendrá UID y GID 1001, la siguiente 1002, etc. Cuando se elimina una cuenta, el ID asociado podría reutilizarse, sin embargo, el sistema no suele hacerlo si todavía existen cuentas con IDs mayores que el eliminado, quedando un hueco en la secuencia de IDs usados.

Por ejemplo, si se crean las cuentas con IDs 1000, 1001 y 1002; a continuación se elimina la cuenta con ID 1001; y por último se crea una nueva cuenta de usuario, el sistema le asignará ID 1003 y no el 1001 aun a pesar de que está libre.

Se desperdician IDs, pero no suele ser un problema, ya que el número máximo de IDs que pueden asignarse en sistemas Linux con kernel versión 2.2.x es de 65536, y en kernel versión 2.4.x de 4.2 miles de millones. Lo que suele ser más que suficiente para el número habitual de usuarios en un sistema Linux.

Borrar una cuenta de usuario no siempre lleva asociado borrar todos los archivos asociados a

esa cuenta. Así si se reutiliza el ID del usuario al que pertenecía la cuenta borrada, automáticamente todos sus archivos no borrados pasan a ser propiedad del nuevo usuario, lo que puede tener consecuencias negativas.

3.4 El archivo /etc/passwd

Toda la información relativa a los usuarios del sistema (a excepción de su contraseña) se encuentra en el archivo **`/etc/passwd`**.

En ese archivo hay una entrada por cada usuario registrado, cada una con **7 campos separados** por el símbolo "**dos puntos**" **:**

fconde:x:1000:1000:Francisco de Asís Conde Rodríguez,,,:/home/fconde:/bin/bash

1. El primer campo es el **nombre de usuario**. En este ejemplo ***fconde***. Es el nombre simbólico que le asignó el administrador al crear su cuenta.
2. El segundo campo (**x**) indica que la **contraseña** se encuentra cifrada en el archivo ***/etc/shadow***.
3. El tercer campo es el **UID del usuario**. En este caso 1000.
4. El cuarto campo es el **GID del usuario**. En este ejemplo 1000. Es el identificador del grupo primario o grupo por defecto del usuario.
5. El quinto campo es lo que se conoce como **campo Gecos** [1] . Incluye información sobre el usuario que ayuda a identificarlo. Normalmente es el nombre completo del usuario. Parte de esta información se muestra cuando se pide información sobre un usuario con la orden **finger**. (Vease el apartado añadir usuarios).
6. El sexto campo es el **directorio raíz del usuario**. Su directorio de trabajo por defecto.
7. El séptimo campo es **la shell** que este usuario prefiere tener **por defecto**. En el ejemplo la ***shell bash***.

---

[1] este nombre proviene de **GCOS** (**S**istema **O**perativo **C**omprensivo **G**eneral) es una familia de sistemas operativos orientados hacia computadoras centrales (mainframes). Su primera versión fue desarrollada por General Electric en 1962; originalmente designado **GECOS** (the **G**eneral **E**lectric **C**omprehensive **O**perating **S**upervisor)

4 Configurando cuentas de usuario

Existen herramientas gráficas para realizar todas las operaciones de configuración de cuentas de usuario, sin embargo, éstas varían de una a otra distribución Linux. Por eso en este apartado se explican las herramientas de consola que son estándares.

4.1 Añadir usuarios

Para añadir usuarios se emplea la utilidad ***useradd*** (en algunos sistemas Linux se llama ***adduser***). Su sintaxis básica es:

```
useradd [-c comentario] [-d home-dir] [-e fecha-expiración] [-f días-inactivos] [-g grupo-por-defecto] [-G grupo[,...]] [-m [-k directorio-esqueleto] | -M] [-p password] [-s shell] [-u UID [-o]] [-r] [-N] nombreusuario
```

Algunos de esos parámetros sólo son válidos cuando el sistema usa shadow passwords. Aunque esa es la configuración estándar de la mayoría de sistemas Linux.

En su forma más simple, sólo hay que teclear **`sudo useradd -m nombreusuario`** para tener una cuenta lista para funcionar. El resto de opciones sirve para modificar el comportamiento por defecto de **useradd** que se define en el archivo **/etc/login.defs**.

Justo después de crear una cuenta de usuario el administrador del sistema debería ejecutar la orden **passwd** para asignarle una contraseña como se verá en la siguiente sección.

Los parámetros de la orden **useradd** tienen el siguiente significado:

- **-c comentario.** Introduce comentario en el campo Gecos del archivo ***/etc/passwd*** (el 5º campo de ese archivo). El nombre Gecos viene de General Electric Comprehensive Operating System, e incluye información para facilitar la identificación del usuario. Por ejemplo, en Ubuntu, por defecto ese campo se rellena con el nombre completo del usuario.

  La orden **finger** sirve para pedir información sobre un usuario. Entre otras cosas muestra el contenido del campo Gecos.

  ```
  fconde@iMac21:~$ finger fconde
  Login: fconde                           Name: Francisco de Asís Conde
  Rodríguez
  Directory: /home/fconde                 Shell: /bin/bash
  ```

- **-d home-dir.** Permite especificar el directorio que será el directorio raíz del usuario. Por defecto, si no se especifica esta opción la cuenta se creará en ***/home/nombreusuario***.

- **-e fecha-expiración.** Permite introducir la fecha en la que la cuenta dejará de estar activa (a partir de esa fecha, la cuenta no se borra, pero el usuairo no podrá acceder a ella). La fecha se expresa en el formato YYYY-MM-DD. Esta opción es muy apropiada para crear **cuentas temporales**, por ejemplo, las de empleados que se contratan sólo por un tiempo en la empresa.

- **-f días-inactivos.** Una cuenta se deshabilita si pasa un determinado número de días desde que su *password* caducó. La opción -f permite especificar para una cuenta concreta qué número de días deben pasar antes de que se deshabilite cuando el password ha caducado. La opción por defecto es -1, lo que indica que una cuenta nunca se deshabilita aunque su password haya caducado.

- **-g grupo-por-defecto.** Especifica el grupo por defecto para el usuario que se está creando. En Ubuntu, el grupo por defecto es un nuevo grupo con el mismo nombre que el usuario que se crea. Este comportamiento por defecto varía de una a otra distribución de Linux.

- **-G grupo[,...].** La opción -G permite especificar grupos adicionales a los cuales pertenece el usuario. Es una lista separada por comas.

- **-m [-k directorio-esqueleto].** La opción `-m` dice al sistema que **cree automáticamente el directorio raíz del usuario**. Normalmente la orden **useradd** lo que hace es copiar en el nuevo directorio el contenido de ***/etc/skel*** (con los permisos del nuevo usuario), que contiene los archivos que todo usuario debería tener por defecto en su directorio.

  > Si no se espeficica la opción **-m**, se crea la cuenta de usuario pero no su directorio raíz, con lo que en realidad la cuenta no es utilizable. El usuario puede hacer login al sistema, pero como no tiene ningún directorio propio, no podría crear ningún archivo salvo en el directorio temporal **/tmp**.

  Si se usa la opción **-m** se puede usar también la opción **-k** seguida de la ruta a un directorio. Esto permite usar otra plantilla para rellenar el contenido por defecto del nuevo directorio raíz del usuario en lugar de usar **/etc/skel**.

- **-M.** Esta opción no puede usarse con -m y -k. Lo que indica es que **no se cree un directorio raíz** para el usuario. Esto es especialmente útil para determinadas **cuentas del sistema** que en realidad no necesitan un directorio propio en el disco.

- **-p password.** Esta opción sirve para asignar una contraseña preencriptada al usuario. No es muy util salvo en scripts, porque primero es necesario encriptar la contraseña y después pasarla como opción. En su lugar es mucho mejor usar la orden passwd justo después de crear la cuenta.

- **-s shell.** Permite indicar la shell que el usuario usará por defecto. En la mayoría de las distribuciones Linux, la shell usada es ***/bin/bash***, pero se puede especificar cualquier otra que el usuario quiera o incluso cualquier otro programa aunque no sea una shell.

  Por ejemplo, las cuentas del sistema suelen incluir como shell el programa **/bin/false** ó **/sbin/nologin**, lo que prohibe que ningún usuario pueda hacer login al sistema usando esa cuenta.

- **-u UID [-o].** Permite especificar el UID que se quiere asignar a un usuario. Si se especifica un UID perteneciente a un usuario ya creado, se recibe un mensaje indicando que ese número de usuario ya

está en uso. Si se usa la opción -u también se puede usar la opción -o que permite asignar el mismo UID de un usuario ya creado a un nuevo usuario. Esto en la práctica, hace que se tenga un único usuario pero con dos nombres distintos. Recordemos que Linux identifica a los usuarios por su UID, no por su nombre.

Un intruso, una de las primeras cosas que hará será crearse una cuenta de usuario con UID 0, es decir, una cuenta de usuario que funcionará con acceso total al sistema como lo tiene el usuario root. Por lo tanto, un buen administrador deberá revisar de vez en cuando el archivo /etc/passwd y eliminar cualquier cuenta sospechosa que tenga UID igual a 0.

- **-r.** La opción -r indica que lo que se está creando es una cuenta de sistema, es decir, una cuenta con un UID menor que UID_MIN (constante definida en /etc/login.defs).
- **-N.** No crear grupo de usuario. Por defecto, muchas distribuciones Linux (entre ellas Ubuntu) crean un nuevo grupo para cada usuario con el mismo nombre del usuario y hacen que este grupo sea su grupo por defecto o grupo primario. Con esta opción, no se crea un grupo nuevo para cada usuario sino que se le asigna el grupo 100 (users) que es un grupo genérico para todos los usuarios que no sean cuentas del sistema.

4.2 Modificar cuentas de usuarios

Las cuentas de usuario pueden modificarse de varias formas. Una de ellas es accediendo al archivo **`/etc/passwd`** y editándolo, pero es mucho mejor usar las utilidades que se estudian en esta sección como **passwd**, **usermod** o **chage**.

4.2.1 Establecer la contraseña

Aunque la orden **useradd** incluye la opción -p para asignar una contraseña al usuario que se está creando, esto no es muy útil ya que requiere que la contraseña esté preencriptada. Por tanto, es mucho más rápido crear el nuevo usuario sin contraseña (sin la opción -p) y a continuación asignarle una contraseña con la orden ***passwd***, que tiene la siguiente sintaxis:

```
passwd [-l] [-u [-f]] [-d] [-S] nombreusuario
```

La orden **passwd** no sólo sirve para cambiar la contraseña, sino que también permite realizar otras tareas de administración relacionadas con las cuentas de usuario.

- **-l.** La opción -l (lock) permite **bloquear una cuenta**. Esto se traduce en que el primer carácter de la contraseña cifrada en el archivo */etc/shadow* es un signo de admiración (!).

  Por defecto, el usuario root en Ubuntu está bloqueado, por lo que nadie puede hacer login en el sistema como root, lo que le daría acceso total al sistema. Esto es así por seguridad.

  root:!:15655:0:99999:7:::

  Un usuario no puede hacer login usando una cuenta bloqueada, pero sus archivos permanecen en el disco y es fácil volver a desbloquear la cuenta. Es muy útil cuando el administrador sospecha que hay actividades fraudulentas en una cuenta. Puede bloquear la cuenta mientras investiga, para que el usuario no pueda seguir accediendo sin necesidad de borrarla permanentemente.

- **-u.** La opción -u (unlock) permite **desbloquear una cuenta** eliminando el signo de admiración inicial en el campo contraseña del archivo */etc/shadow*.

  Por defecto la orden **useradd** crea cuentas sin password y bloqueadas. Por tanto, usar la orden passwd -u en una cuenta recién creada, da como resultado un error, y que se tendría una cuenta sin password. Para forzar que se ejecute la orden se usa la opción -f que fuerza el desbloqueo de una cuenta aunque no tenga password.

- **-d. Elimina la contraseña** de una cuenta, dejándola sin password.
- **-S.** Muestra la **información relativa a la contraseña** de una cuenta concreta. Por ejemplo, si tiene password (P) o no (L), etc.

Los usuarios normales sólo pueden acceder a la información de su propia cuenta con **passwd**. Tampoco pueden ejecutar opciones como -l, -u, -f, -d. Es necesario ser ***root*** o ejecutar la orden ***passwd*** precedida de ***sudo*** para poder especificar estas opciones.

Cuando se ejecuta passwd sin opciones, el sistema pregunta al usuario que ejecuta la orden su contraseña actual y si es correcta, le pregunta la nueva contraseña dos veces (para evitar contraseñas incorrectas por errores de tecleo):

```
fconde@iMac21:~$ passwd
Cambiando la contraseña de fconde.
(actual) contraseña de UNIX:
Introduzca la nueva contraseña de UNIX:
Vuelva a escribir la nueva contraseña de UNIX:
```

Como puede verse, ninguna de las contraseñas se muestra en la pantalla mientras se escribe, ya que de esta forma se impide que alguna otra persona que pueda estar mirando pueda ver la contraseña.

Si el usuario que ejecuta la orden passwd es root, no se le pide la contraseña actual antes de introducir la nueva contraseña. De esta forma el usuario root puede cambiar la contraseña de un usuario "normal" que la haya olvidado. De otra forma sería imposible, ya que ningún usuario (ni siquiera root) puede ver la contraseña de ningún usuario del sistema.

Linux acepta como contraseñas cualquier combinación de letras, números y símbolos de puntuación. En las contraseñas Linux también distingue entre mayúsculas y minúsculas.

4.2.2 Desbloqueando al usuario root

Como ya se ha indicado, el usuario *root* suele estar bloqueado por defecto por seguridad, ya que el usuario *root* tiene acceso total al sistema. Eso significa que nadie puede hacer *login* como *root*, aunque un usuario administrador sí que puede ejecutar algunas órdenes como *root* con **sudo**. Sin embargo, si es necesario, se puede desbloquear al usuario *root* siempre que se sea un usuario administrador del sistema.

En primer lugar, hay que abrir la consola en modo *root* con la orden:

```
sudo -s
```

El prompt cambia, y en lugar de terminar en el signo $, termina en signo de número #. Eso indica que estamos en modo *root*.

```
fconde@iMac21:~$ sudo -s
root@iMac21:~#
```

A continuación se ejecuta **passwd**. Como somos el usuario *root*, el cambio permite asignar una contraseña y desbloquear la cuenta.

A partir de ese momento se puede hacer login en el sistema como root, tanto en modo consola, como en modo gráfico interactivo.

4.2.3 Modificar un usuario ya creado

Para crear una cuenta de usuario ya creada se usa la orden **usermod**, cuyas opciones son bastante parecidas a **useradd**. Las opciones más interesantes de usermod son:

```
usermod [-d directorio [-m]] [-G grupo[,...] -a] [-l nuevonombre] [-L]
[-U] [-u UID [-o]] nombreusuario
```

- **-d directorio.** Cambia el directorio raíz del usuario (su directorio de inicio) por el que se especifica como argumento, pero no mueve los archivos que hubiera en el directorio raíz anterior al nuevo directorio.

  Si se especifica la opción -m, entonces sí que se mueven los archivos al nuevo directorio.

- **-G grupo[,...]** Establece los grupos que se pasan como argumento como grupos suplementarios de los que el usuario también es miembro. Es una lista de grupos separados por comas y sin espacios.

La anterior lista de grupos a los que pertenecía el usuario se elimina y se sustituye por esta, por lo que hay que tener cuidado de no olvidar ninguno.

Para eso se usa la opción -a. Cuando se combina con la opción -G, la lista de grupos que se pasa como argumento se **añade** a la lista actual de grupos a los que pertenece el usuario en lugar de reemplazarla.

- **-l nuevonombre.** Cambia el nombre de usuario de la cuenta al nuevo que se pasa como argumento.

  ```
  usermod conde -l fconde
  ```

  cambia el nombre de usuario de la cuenta conde a fconde.

- **-L.** Permite bloquear la cuenta.
- **-U.** Permite desbloquear la cuenta.
- **-u UID [-o].** Permite cambiar el UID de una cuenta de usuario al que se pasa como argumento. Si el UID que se pasa ya está asignado a una cuenta de usuario el sistema da un error. Para forzar que se reutilice el mismo UID en dos cuentas distintas se usa la opción -o.

  Hay que usar esta opción con cuidado, ya que cambiar el UID de un usuario no afecta para nada a los archivos que ya hubiese creado ese usuario que conservan el UID anterior. Si es realmente necesario hacerlo, debería seguirse por la ejecución de la orden:

  ```
  chown -R nombreUsuario /home/nombreUsuario
  ```

  Para asegurarnos de que todos los archivos de la cuenta de usuario que están en su directorio raíz cambian su propietario al nuevo UID que se le haya asignado.

  Esa orden no cambia aquellos archivos que puedan estar en otros lugares distintos del directorio raíz del usuario. Si existieran esos archivos deberían localizarse con la orden **find** y cambiarse con **chown** para evitar efectos no deseados.

Cambiar las características de una cuenta de usuario (como por ejemplo el directorio raíz) mientras el usuario está conectado al sistema puede tener consecuencias no deseadas. Por eso, un administrador, debería ejecutar la orden **who** (que indica los usuarios que están conectados en un instante dado) antes de realizar cambios en una cuenta para comprobar que el usuario al que afectan los cambios no está conectado.

4.2.4 La orden chage

Esta orden permite modificar los ajustes de una cuenta de usuario relativos a la caducidad de la cuenta. Es posible configurar las cuentas de usuario en Linux de forma que caduquen si:

- La contraseña no se ha cambiado en un periodo de tiempo específico.
- La fecha actual del sistema es posterior de una fecha de caducidad prefijada.

Estos ajustes son controlados por medio de la orden **chage** que tiene la siguiente sintaxis:

```
chage [-l] [-m minnumdias] [-M maxnumdias] [-d ultimodia] [-I diasinactividad]
[-E fechacaducidad] [-W diasaviso] nombreusuario
```

Que tienen el siguiente significado:

- **-l.** Hace que se muestre la información relativa a la cuenta cuyo nombre de usuario se pasa como argumento.

  ```
  fconde@iMac21:~$ chage -l fconde
  Último cambio de contraseña : oct 27, 2012
  La contraseña caduca : nunca
  Contraseña inactiva : nunca
  La cuenta caduca : nunca
  Número de días mínimo entre cambio de contraseña : 0
  Número de días máximo entre cambio de contraseñas : 99999
  Número de días de aviso antes de que expire la contraseña : 7
  ```

- **-m minnumdias.** Establece el mínimo número de días entre dos cambios de contraseña. Un valor de 0 (el valor por defecto) indica que el usuario puede cambiar la contraseña tantas veces al día como quiera. Un valor de 1 indica que el usuario puede cambiar la contraseña una vez al día. Un valor de 2 indica que el usuario puede cambiar la contraseña una vez cada dos días, etc.
- **-M maxnumdias.** Establece el máximo número de días entre dos cambios de contraseña. Un valor de 30 indica que el usuario debe cambiar la contraseña al menos una vez cada 30 días. El valor por defecto es de 99999 que equivale aproximadamente a 270 años, lo que en la práctica equivale a que el sistema no le va a obligar a cambiarla nunca a menos que el usuario quiera cambiarla por propia iniciativa.

  Si el usuario cambia la contraseña antes de que se agote el periodo de tiempo especificado, el sistema reinicializa la cuenta atrás al máximo número de días.
- **-d ultimodia.** Establece el último día en que la contraseña fue cambiada. Normalmente se mantiene automáticamente por las herramientas del sistema, pero puede cambiarse a mano para alterar artificialmente la cuenta de días desde el último cambio de contraseña.
- **-I diasinactividad.** Si la contraseña ha caducado y el usuario no la cambia, al cabo de unos días la cuenta se bloquea, de forma que es obligatorio que el usuario cambie la contraseña antes de poder utilizarla de nuevo. Este parámetro permite decidir el número de días que pasan desde que la contraseña caduca hasta que la cuenta se bloquea.
- **-E fechacaducidad.** Establece la fecha en la que la cuenta va a dejar de estar activa. Tiene formato YYYY/MM/DD.
- **-W diasaviso.** Establece con cuantos días de antelación se va a avisar al usuario de que su contraseña va a caducar. Es una buena práctica que el sistema avise al usuario de que su contraseña va a caducar para que el usuario pueda cambiar la contraseña y que su cuenta no se bloquee. El valor por defecto es una semana (7 días).

Obviamente la orden chage sólo puede ser ejecutada por root, excepto la opción -l aplicada a la propia cuenta del usuario que ejecuta la orden.

4.3 Eliminar usuarios

Cuando ya no se va a necesitar mas una cuenta de usuario, lo mejor es eliminarla del sistema. Tener cuentas operativas pero en desuso puede plantear problemas de seguridad, por ello el administrador del sistema, de vez en cuando debe eliminar aquellas cuentas que ya no sean necesarias.

La orden para eliminar cuentas de usuairo es **userdel** cuya sintaxis es la siguiente:

```
userdel [-r [-f]] nombreusuario
```

Las opciones tienen el siguiente significado:

- **-r.** Elimina también los archivos del usuario que están en su directorio raíz /home/nombreusuario y el propio directorio raíz, así como el correo del usuario. Si no se especifica la opción -r, la cuenta de usuario se elimina pero no sus archivos, lo que puede dar lugar a confusión.

  Si dentro del directorio /home/nombreusuario hay archivos que no son propiedad de la cuenta nombreusuario, entonces esos archivos permanecen en el disco. La opción -f que sólo se puede usar junto con -r, fuerza a eliminar todos los archivos del directorio /home/nombreusuario tanto si son propiedad de nombreusuairo como si no.

En ocasiones puede ser interesante hacer una copia de respaldo de los archivos de un usuario antes de eliminarlos del directorio /home. De esta manera se conserva el trabajo hecho por un empleado que ya no pertenezca a la empresa, etc. Para ello se usa la orden find ejecutada como usuario root. Por ejemplo, si se quiere hacer una copia de todos los archivos del usuario user1 en la carpeta del usuario root (donde sólo él puede leerlos) antes de eliminar la cuenta de user1 la orden es:

```
find / -user user1 -exec cp {} /root/user1 \;
```

Si se olvidó usar la opción -r al eliminar una cuenta de usuario, ya no es posible usar el nombre de usuario para localizar sus archivos con la orden find (opción -user nombreusuario), en su lugar, hay que usar la opción -uid id, donde id es el identificador del usuario cuyos archivos se quieren eliminar.

Si no nos acordamos del id del usuario que hemos borrado, siempre se puede ir a la carpeta /home/nombreusuario y allí con la opción ls -l ver a quién está asignado alguno de los archivos que contenga.

5 Configurando grupos

Las órdenes para configurar grupos tienen una estructura muy parecida a las órdenes para configurar usuarios. También existen herramientas gráficas para hacer el trabajo, pero, al igual que pasaba con las herramientas para usuarios, no están estandarizadas entre las distintas distribuciones de Linux, por lo que en esta sección se van a estudiar las herramientas de línea de órdenes.

5.1 Añadir grupos

Linux proporciona la orden **groupadd** para añadir un nuevo grupo al sistema. Es necesario ser *root* para ejecutar esta orden o usar **sudo**. Es parecida a la orden **useradd**, pero tiene menos opciones. Su sintaxis es:

```
groupadd [-g GID [-o]] [-r] nombregrupo
```

El significado de cada opción es el siguiente:

- **-g GID [-o].** Permite elegir un identificador concreto para el grupo: GID. Si se omite este parámetro, el sistema automáticamente asigna el siguiente identificador que esté disponible mayor que GID_MIN (esta constante se define en el archivo /etc/login.defs)

  Si se intenta crear un grupo con el mismo identificador de otro grupo ya creado, el sistema dará un error. Para forzar que se use el mismo identificador para el nuevo grupo se usa la opción -o (sólo se puede usar junto con la opción -g). En la práctica, se tiene un único grupo, pero con varios nombres.

- **-r.** Indica a **groupadd** que cree un grupo de sistema en lugar de un grupo "normal" de usuario.

  Una cuenta de sistema, tiene identificador menor que GID_MIN (definido en /etc/login.defs)

En la mayoría de los casos, no será necesario especificar ninguna de las opciones.

> Por ejemplo, para crear un grupo de usuario llamado proyecto1, la orden es:
>
> `groupadd proyecto1`

5.2 Modificar grupos

La información de un grupo puede ser modificada una vez que éste se ha creado. Para ello se usan dos órdenes: **groupmod** y **usermod**.

La orden **groupmod** tiene la siguiente sintaxis:

```
groupmod [-g GID [-o]] [-n nuevonombregrupo] nombregrupo
```

El significado de las opciones es el siguiente:

- **-g GID [-o].** Cambia el identificador del grupo por el nuevo que se pasa como argumento. Si ya existe un grupo con el nuevo identificador que se quiere asignar al grupo, el sistema devuelve un mensaje de error. La opción -o (sólo se puede usar junto con -g) fuerza que se reutilice un identificador ya en uso en otro grupo.
- **-n nuevonombregrupo.** Permite cambiar el nombre del grupo por el nuevo que se pasa como argumento.

Uno de los cambios que más frecuentemente se realizan a los grupos, que consiste en añadir y eliminar usuarios que pertenecen al mismo, se realiza con la orden **usermod** que ya se ha estudiado. En concreto, la

opción -G permite añadir a un usuario a un grupo. Se pueden especificar varios grupos separados por comas sin espacios.

> La siguiente orden (ejecutada como root):
>
> ```
> usermod -G grp1,grp2,grp3 usuario1
> ```
>
> Añade al usuario: usuario1, a los grupos ya creados grp1, grp2 y grp3, ***y lo elimina de los otros grupos a los que pudiera pertenecer.***

Es muy importante asegurarse de que en la opción -G se escriben todos los grupos a los que se quiera que pertenezca el usuario, ya que el usuario dejará de pertenecer a aquellos que no se escriban. Para averiguar los grupos a los que pertenece un usuario se puede usar la orden:

```
groups nombreusuario
```

También se pueden ver en el archivo */etc/group*. Por ejemplo, la siguiente orden muestra todos los grupos a los que pertenece el usuario **fconde** en un sistema Linux:

```
fconde@iMac21:/home$ sudo less /etc/group fconde | grep fconde
adm:x:4:fconde
cdrom:x:24:fconde
sudo:x:27:fconde
dip:x:30:fconde
plugdev:x:46:fconde
lpadmin:x:109:fconde
fconde:x:1000:
sambashare:x:124:fconde
pepe:x:1002:fconde
```

que en este ejemplo concreto son: adm, cdrom, sudo, dip, plugdev, lpadmin, fconde, sambashare y pepe.

Muchos administradores prefieren usar la orden **gpasswd** en lugar de **usermod** para añadir usuarios a grupos, ya que permite añadir nuevos usuarios a grupos sin necesidad de escribir todos los grupos a los que ya pertenezcan esos usuarios. Lo que es menos propenso a errores.

Una de las órdenes más potentes para modificar grupos es la orden **gpasswd**. Su sintaxis es:

```
gpasswd [-a user] [-d user] [-R] [-r] [-A usuario1[,...]] [-M
usuario1[,...]] grupo
```

El significado de las opciones es el siguiente:

- **-a nombreusuario.** Añade el usuario que se indica como nuevo miembro del grupo, dejando el resto de miembros de ese grupo inalterados.

  Esta opción suele ser preferible a la opción -G de la orden usermod para añadir usuarios a grupos.

- **-d nombreusuario.** elimina el usuario que se indica como miembro del grupo. A partir de ese momento el usuario ya no pertenece al grupo.

- **-R.** Impide el uso de la orden newgrp en este grupo.

- **-r.** Elimina la contraseña para el grupo.

- **-A usuario1[,...].** Sustituye la lista de usuarios administradores de este grupo por la que se pasa como argumento. Es una lista de nombres de usuairos separados por comas y sin espacios.

  Un usuario administrador de un grupo puede añadir y eliminar miembros a ese grupo, así como cambiar su contraseña.

  El uso de esta opción sobreescribe completamente el contenido de la lista de administradores del grupo, por lo que hay que asegurarse de que se escribe el nombre de usuario de todos los administradores.

- **-M usuario1[,...].** Funciona igual que la opción -A, pero además añade a los usuarios como

miembros del grupo.

Si se usa la orden **gpasswd** sin ninguna opción, entonces la orden permite cambiar la contraseña para un grupo.

La contraseña de un grupo afecta a la orden **newgrp**, que como sabemos, permite que un usuario miembro de un grupo, pueda hacerlo temporalmente su grupo primario, de forma que a partir de ese momento, todos los archivos que cree o los programas que ejecute estén asignados a ese grupo. Si un grupo tiene contraseña, aunque un usuario pertenezca al grupo, cuando usa la orden **newgrp** para hacer de ese grupo, su grupo primario, el sistema le va a pedir la contraseña.

5.3 Eliminar grupos

Para eliminar un grupo se usa la orden **groupdel**, cuya sintaxis es muy sencilla:

```
groupdel nombregrupo
```

Si el grupo que se quiere eliminar es el grupo primario de algún usuario, el sistema informa del error y no permite que se borre al grupo. Antes habría que **cambiar el grupo primario del usuario a otro grupo para poder eliminar el grupo**.

De la misma forma que ocurría cuando se eliminaba un usuario, si se elimina un grupo, se pueden localizar todos los archivos asignados a ese grupo mediante su GID. Para ello se usa la orden **find** con la opción -gid.

> Por ejemplo, la orden:
>
> ```
> find / -gid 1000
> ```
>
> busca en el sistema completo todos los archivos cuyo identificador de grupo GID es igual a 1000.

6 Ejercicios

1. Revisa el archivo /etc/passwd y escribe la lista de los usuarios definidos en el mismo que tengan como shell /bin/false. ¿Qué tipo de usuarios son? ¿Por qué tienen esa shell asignada?
2. Supongamos que se tiene un disco duro auxiliar montado en el directorio /home2. Supongamos que existen los grupos de usuarios depto1 y depto2.

   Se quiere crear una nueva cuenta de usuario completamente operativa para un usuario llamado Juan López, cuya directorio raíz deberá estar ubicado en /home2, que pertenezca a los grupos depto1 y depto2 siendo su grupo primario depto1 y con shell por defecto tcsh.

   ¿Cual es la/s orden/es que debes escribir para conseguirlo?
3. Siendo el usuario root, se quiere eliminar la cuenta del usuario user1 cuyo directorio raíz es /home/user1 y que además tiene algunos archivos en el directorio /tmp.

   Escriba las órdenes necesarias para eliminar completamente la cuenta del usuario user1 incluyendo todos sus archivos, pero haciendo previamente una copia de seguridad de los mismos en el directorio /root.
4. Averigua a qué grupos pertenece el usuario con el que te hayas conectado para hacer las prácticas. Hazlo al menos de dos formas distintas. Explica cómo lo has hecho.
5. ¿Qué dos formas existen en Linux para añadir, con una orden, un usuario a un grupo ya creado? ¿Cual de ellas es mejor? ¿Por qué?
6. Un usuario del sistema con nombre de usuario empleado1 ha olvidado su contraseña. ¿Qué harías para restablecerla?
7. ¿Para qué se usa la contraseña de un grupo?
8. Si se pudiese eliminar un grupo que sea el grupo primario de algún usuario, el sistema quedaría en