Administración de Usuarios

Miguel Ortuño
Escuela de Ingeniería de Fuenlabrada
Universidad Rey Juan Carlos

Octubre de 2024

Usuarios y grupos

- Cada usuario tiene un identificador (*UID*), un grupo principal (o primario) al que pertenece (*GID*), una serie de grupos adicionales, un nombre de usuario (*login*), un directorio de trabajo (*home*)
- La orden `id` nos da el UID, el GID y los grupos adicionales de un usuario
- Cada usuario puede tener dos tipos de recursos en un sistema UNIX: Procesos y Ficheros

- Cada UID y cada GID puede tener asociado un nombre, especificado en los ficheros `/etc/passwd` y `/etc/group`, respectivamente
- La información de `/etc/passwd` y `/etc/group` la utilizan diversas órdenes de administración. Ambos ficheros deben existir y ser coherentes para que el sistema funcione correctamente
- No es recomendable editar estos ficheros directamente, sino mediante mandatos como `usermod`
- Si se edita `/etc/passwd` y `/etc/group/` directamente, debe usarse `vipw` y `vigr`

Importancia de las contraseñas

- El campo *passwd* en el `/etc/passwd` y el `/etc/shadow` se encuentra cifrado con una función *hash* para evitar que los usuarios (y administradores) puedan conocer las contraseñas de otros usuarios.
- Se usa un cifrado de un solo sentido: no existe algoritmo para averiguar la contraseña a partir de estos ficheros.
- Pero se pueden probar varias contraseñas, hasta millones por segundo (John the Ripper).
- Es imprescindible elegir palabras clave seguras, que no aparezcan en diccionarios, evitando nombres o fechas significativas, combinando símbolos, y de la mayor longitud posible.

- Ejemplos de malas contraseñas:

  `123456`
  `4312`
  `toby`
  `r2d2`
  `tornillo`
  `fromage`
  `mostoles`

- Contraseñas que parecen buenas, pero son malas:

  `XCV330`
  `NCC-1701-A`
  `ARP2600V`

- Buenas contraseñas
  Para usos ordinarios, una contraseña razonablemente buena será parecida (¡pero no igual!) a una de estas

```
Contraseña  Nemotécnico
-----------------------------------------------
00QuReMa:   Queridos Reyes Magos:
3x4igDoze   3x4=doce
1pt,yTp1    uno para todos,todos para uno
19Dy500n    19 dias y 500 noches
R,cmqht?0   Rascayú, cuando mueras que harás tú?
wali1YS!    we all live in a Yellow Submarine
Lh10knpr.   le hare una oferta que no podrá rechazar
```

  Para aplicaciones especialmente sensibles, es necesario ampliar el *keyspace* a 14 caracteres o más

- Es conveniente que busquemos e inutilicemos las contraseñas débiles de nuestros usuarios, ya que suponen un primer punto de entrada en nuestro sistema
- En Unix, el root no puede leer las contraseñas. Pero en otros entornos sí. Y muchos usuarios emplean siempre la misma contraseña
  Ejemplo:
  1. Juan Torpe usa como contraseña dgj441iU en juan.torpe@hotmail.com
  2. Juan Torpe se apunta en `www.ladygaga-videos-prohibidos.com`, con su cuenta de correo y su contraseña de siempre
  3. El administrador malicioso de este web ya conoce el nombre de Juan, su cuenta de correo y su contraseña.
     - Puede usar la función *¿contraseña olvidada?* y colarse en cualquier otra cuentas de Juan

  Debemos instruir a nuestros usuarios sobre esto

- Los usuarios sin duda olvidarán en ocasiones su contraseña y tendremos que generles una nueva, de forma segura
- Pero es muy poco profesional que nosotros como administradores olvidemos una contraseña. Debemos usar varias y guardarlas de forma medianamente segura (gpg, keepassx, lastpass, etc)

Secret sharing

Aunque pongamos cuidado extremo en guardar nuestra contraseña, esto puede no ser suficiente.

- ¿Y si a pesar de todo, la perdemos o nos la roban?
- ¿Y si no estamos disponibles? (Viaje, enfermedad, muerte...)
- ¿Y si nos secuestran?
- ¿Y si hacemos algún disparate?

Podríamos dividir la contraseña en n trozos y repartirlos entre n personas de nuestra confianza, de forma que con el acuerdo de todos, el secreto es recuperable

- Problema: Bastaría con que se pierde un trozo para perder el secreto
- Solución: Algoritmos de *secret sharing*

Un esquema *secret sharing* es aquel que permite dividir un secreto en n fragmentos, de forma que basten t ($t < n$) para reconstruirlo
Armory incluye esto de manera nativa, con electrum podemos usar una herramienta independiente, ssss
Con la orden `ssss-split`, dividimos el secreto en n fragmentos

```
koji@mazinger:~$ ssss-split -t 3 -n 5
WARNING: couldn't get memory lock (ENOMEM, try to adjust RLIMIT_MEMLOCK
Generating shares using a (3,5) scheme with dynamic security level.
Enter the secret, at most 128 ASCII characters
ABRETE SESAMO
1-378a29cbbe9b38f0d473bdab0654
2-d7cc58bbce72070afc675f0991dd
3-d42a4e6f0dca1d32a6d1f96e4e92
4-629dd862cb052f8774bdb26d91df
5-617bceb608bd35bf2e0b140a4e82
```

Con la orden `ssss-combine`, recuperamos el secreto

```
koji@mazinger:~$ ssss-combine -t 3
WARNING: couldn't get memory lock (ENOMEM, try to adjust RLIMIT_MEMLOCK
Enter 3 shares separated by newlines:
Share [1/3]: 2-d7cc58bbce72070afc675f0991dd
Share [2/3]: 3-d42a4e6f0dca1d32a6d1f96e4e92
Share [3/3]: 5-617bceb608bd35bf2e0b140a4e82
Resulting secret: ABRETE SESAMO
```

Si el secreto es mayor de 128 bytes, se cifra con una clave tradicional y se aplica ssss a esta nueva clave

/etc/passwd

- Contiene la información de todos los usuarios del sistema.
- Contenido: líneas con campos separados por dos puntos: *login*:*passwd*:*UID*:*GID*:*info*:*home-dir*:*shell*
- El campo "*login*" puede tener hasta 32 caracteres en Linux, pero se recomienda limitarlo a 8, como en los UNIX clásicos
- El campo "*passwd*" contiene la contraseña cifrada (con DES o con MD5) y puede estar en otro fichero, en el `/etc/shadow`.
- El campo "*info*" contiene el nombre real del usuario e información adicional como el teléfono, etc. Por (desafortunados) motivos históricos, también se le denomina GECOS
- En algunos sistemas, puede haber información externa (NIS, LDAP...)
- Programas que lo utilizan directamente: `login`, `su`, `passwd`.

/etc/group

- Nombres de grupos del sistema, y miembros de cada grupo.
- Contenido: líneas con campos separados por dos puntos: *nombre*:*passwd*:*GID*:*lista-logins*
- "*lista-logins*" son usuarios separador por comas que pertenecen a ese grupo.
- El campo "*passwd*" no se suele utilizar. Permite ingresar en un grupo en el que no se es miembro.
- En algunos sistemas, puede haber información externa (NIS, LDAP...)

/etc/shadow

- Si existe, contiene las contraseñas cifradas de los usuarios del sistema.
- Contenido: líneas con campos separados por dos puntos:

  $login:passwd:a:b:c:d:e:f:g$

  a: momento en que la *passwd* fue cambiada por última vez.
  b: días que deben pasar antes de que pueda cambiarse.
  c: días después de los cuales la *passwd* debe cambiarse.
  d: días antes de la expiración para avisar al usuario.
  e: días después de la expiración para desactivar la cuenta.
  f: momento en que la cuenta se ha desactivado.

  g: campo reservado.

Para mejorar la seguridad se añade un "salt"

salt es un tipo de *nounce*: Number used once. 2 bytes aleatorios que se añaden a la contraseña
Sin salt

```
password  -->  hash (password)
"sesamo" --> zv/coRb$PjGToGEqNZF434TmQ7bAH.rVi3i.o7IWQAI9qqzeGKe/JkJq
bDfQE2gBFYzBTDNCHyoxpZvSLhenkPT3L6aZN0
```

El atacante puede usar una *rainbow table*: El resultado de aplicar hash a un diccionario completo. Si encuentra la hash en la tabla, conoce la contraseña que fue usada

```
rainbow table
hash(palabra1)
hash(palabra2)
hash(palabra3)
```

Con salt

```
password+salt  -->  hash (password+salt)

rainbow table:
hash(palabra1+salt1)
hash(palabra1+salt2)
hash(palabra1+salt3)
```

- *salt* se guarda en abierto: se añade al hash, son los primeros dos bytes
- Esto obliga a que la rainbow table sea mucho mayor, puede hacerla inviable

Desactivar un usuario del sistema

- Bloquear su contraseña en el `/etc/passwd` o `/etc/shadow` (añadiendo un carácter "-" o "*", por ejemplo).
- Eliminar sus tareas periódicas (`/var/spool/cron`).
- Revisar `/etc/aliases` y `.forward` por si el usuario tuviera acciones a realizar con el correo recibido.

Eliminar un usuario

- `userdel`
- `userdel -r` también borra su correo y su *home*

Usuarios especiales

- No todas las líneas del `/etc/passwd` corresponden con usuarios físicos.
- Super-usuario: uid=0 (su *login* es normalemente `root`).
- Otros usuarios del sistema: se utilizan para:
  - tareas específicas de administración
  - propietarios de determinados ficheros del sistema
  - ejecución de determinadas aplicaciones (bases de datos, servidores de web, ftp, e-mail, noticias, etc)
- Normalmente, los usuarios normales tienen UIDs entre el 1000 y el 30000.

Cambio de contraseña

Para cambiar la contraseña y otros datos se utilizan las órdenes `passwd` (contraseña), `chfn` (info/gecos), `chsh` (*shell*):

- Estas órdenes tienen *set-uid* para que un usuario normal pueda modificar información privilegiada.
- Antes de nada, piden la *passwd* del usuario para verificar que es quien dice ser.
- Bloquean cada fichero a modificar para asegurar exclusión de accesos.
- Realizan las modificaciones.
- Desbloquean ficheros.

Para crear o cambiar la contraseña de un usuario desde un script

- `echo "jperez:sesamo" | chpasswd`

Cambios de usuario y grupo

- `su` ejecuta otra *shell* bajo un usuario distinto.
  - `su jperez`
    ejecuta otra shell, perteneciente al usuario *jperez*
    No hace falta ser root para ejecutar esto, si la invoca un usuario ordinario, le pedirá la contraseña de *jperez*
  - `su`
    ejecuta shell con uid=0 (`root`).
  - Pide la contraseña del usuario destino, excepto salvo si el origen es `root`.
- `newgrp` ejecuta una *shell* con distinto GID.
  - Tiene *set-uid*
  - `newgrp` permite cambiar el GID a otro grupo al que pertenezcamos (cambia el grupo primario)

Mandatos que sólo pueden ejecutarse como root

- `groupadd` *grupo*
  crea un grupo
- `adduser` *usuario*
  añade un usuario. Copia en su home el directorio `/etc/skel`
  `adduser` *usuario grupo*
  añade un usuario a un grupo
- `usermod -g` *grupo_primario usuario*
  Cambia el grupo primario por omisión del usuario
- `passwd` *usuario*
  Cambia la contraseña de un usuario

- `chown` *dueño fichero(s)*
  cambia el dueño de un fichero
- `chgrp` *dueño fichero(s)*
  cambia el grupo de un fichero

Mandatos para cualquier usuario

- `passwd`
  Cambia la contraseña del usuario
- `newgrp` *grupo*
  Entre los grupos de un usuario, elige el grupo primario

Añadir usuarios desde un script

- Como hemos visto, el comando `adduser` permite al `root` añadir usuarios [1].
- El uso más sencillo es interactivo, *a mano*. El comando va preguntando la contraseña y los datos personales (nombre completo, número de despacho, etc). Todo esto se lee desde la entrada estándar.
- Para crear el usuario desde un script, podemos usar opciones en *adduser*. Pero para añadir la contraseña, es necesario otro comando: `chpasswd`

[1] Otra alternativa es el comando `useradd`, pero es algo más laborioso

```
#!/bin/bash
USERNAME="jperez"
PASSWORD="sesamo"
GECOS="Juan Perez,Oficina 101,555-1234,555-5678,Admin"
# Por motivos históricos, a los datos personales se
# les llama "campo GECOS"
# Nombre, oficina, telef trabajo, telef casa, otros
# Se puede dejar en blanco. Hoy pueden resultar obsoletos

adduser --gecos "$GECOS" --disabled-password "$USERNAME"

echo "$USERNAME:$PASSWORD" | chpasswd
```