Fundamentos de Usuarios e Permisos

Sumario

Gestión de Usuarios y Permisos

El modelo de seguridad del Sistema Linux basa su simplicidad y robustez en la utilización del concepto de usuario y grupo para controlar este importante aspecto. Cada archivo, o directorio (que no es más que un tipo especial de archivo) tiene un propietario, un usuario. El propietario de un archivo albergará permisos sobre ése archivo, en general dispondrá de los máximos permisos sobre él, y además podrá modificar los permisos de los archivos de los que es propietario.

Los demás usuarios del sistema y miembros del mismo grupo de seguridad que el propietario del archivo podrán ostentar específicamente permisos sobre él, pero nunca modificar ellos mismos los permisos de archivos que no son de su propiedad. En resumen, salvo una excepción, solo el propietario del archivo (o directorio) podrá controlar la asignación de permisos (RWX, Lectura, Escritura, Ejecución) a los demás usuarios. La excepción a la que se refiere el párrafo anterior es el **superusuario o root**. Todo sistema Linux crea por defecto un usuario root y es el único que tiene por defecto permisos globales y no restringidos en el sistema.

Otro aspecto fundamental relacionado con la seguridad de Linux y los usuarios es el concepto de propietario de un proceso. Cuando un programa se ejecuta en el Sistema Operativo, como ya sabemos, se crea un proceso para él. Ese proceso podrá acceder a aquellos recursos a los que el propietario del programa tenga acceso. Según esto, si un programa (archivo ejecutable) es ejecutado, solamente podrá acceder a aquellos recursos del sistema (archivos, dispositivos, etc.) a los que el propietario pueda efectivamente acceder, en caso contrario obtendrá una negativo por parte del Sistema Operativo para operar con esos recursos.

En Linux, al igual que en Windows, se reconoce el concepto de **grupo** de seguridad (o simplemente grupo). Cada usuario puede pertenecer a uno o a varios grupos de usuarios. La utilización de grupos permiten simplificar la gestión de permisos cuando tenemos que manejar grandes cantidades de usuarios, de modo que, se pueden especificar los permisos a nivel de grupos, con lo cual, indirectamente, concedemos permisos sobre los usuarios del grupo.

Permisos

Un Permiso establece un derecho de un usuario sobre un Archivo y Directorio. Linux utiliza un modelo simple, pero potente, de gestión de Permisos de usuario. Como en Linux "todo" es un Archivo (incluídos como ya se dijo los Directorios) este modelo de seguridad afecta directamente a todos los aspectos de uso del Sistema.

Existen 3 tipos de Permisos: **r** (read, lectura), **w** (write, escritura) y **x** (execution, ejecución) Existen 4 tipos de receptores de Permisos: **u** (user, el propietario del Archivo o Propietario), **g** (group, los usuarios miembros del mismo grupo de seguridad del usuario) **o** (others, los usuarios que no son ni propietarios ni pertenecen al grupo del propietario) y **a** (all, todos)

Comandos para cambio de Permisos y Propietario de un Archivo o Directorio:

chown: Esta orden se utiliza para cambiar el propietario de un Archivo o Directorio. La sintaxis básica es:

chown [-R] nuevo_usuario[:grupo_usuario] archivos

Ejemplo:

```
chown www-data:www-data /var/www
```

cambia el propietario del directorio /var/www al usuario www-data y grupo www-data. La **opción -R** se utiliza para cambios de propietario recursivos en Directorios, es decir, modifica el propietario dentro del Directorio y en todos los Subdirectorios

chmod**: Este comando es utilizado para cambiar los permisos de Archivos y Directorios**

chmod [-R] cambio archivos

Ejemplo:

```
chmod a+x /home/javier/mi_programa
```

el comando anterior concede **(+)** el permiso de ejecución **(x)** a todos los usuarios del sistema **(a)**

Esta sintaxis abreviada indica:

- **¿a quién? (u, propietario, g, ,grupo del propietario, o, otros, a, todos)**
- **¿concede o quita? (+, concede, -, quita)**
- **¿qué permisos? (r, lectura, w, escritura, x, ejecución)**

Hay otra sintaxis alternativa del comando chmod basada en **máscaras**. La máscara es un valor numérico de 3 cifras, cada una de las cuales puede tener los valores:

- 0: ningún permiso
- 4: permiso de lectura (r)
- 5: permisos de lectura y ejecución (rx)
- 6: permisos de lectura y escritura (rw)
- 7: todos los permisos, lectura, escritura y ejecución (rwx)

Por tanto los pesos de cada permiso en la expresión numérica son (notad que los valores corresponden con los pesos de las potencias de 2 según la posición de la cifra):

r w x

4 2 1

El resultado de obtener la máscara de permisos es la suma de los permisos concedido. Así, **rw es 6 (4+2=6)**, **rx es 5 (4+1=5)** y **rwx es 7 (4+2+1=7)**

Como necesitamos especificar permisos para 3 identidades (usuario, u, grupo, g, y otros, o) **se utilizan 3 cifras decimales** con los valores anteriores, indicando, de izquierda a derecha, los **permisos para usuario, grupo y otros** respectivamente.

Ejemplo:

u g o (user, group, others)

7 5 4

máscara que indica permisos de rwx (7) para el usuario (u), permisos rx (5) para los miembros del grupo del propietario y r (4) para los otros

La correspondencia entre máscaras octales y permisos se ve en la siguiente tabla

- - -	= 0	no se tiene ningún permiso
- - x	= 1	solo permiso de ejecución
- w -	= 2	solo permiso de escritura
- w x	= 3	permisos de escritura y ejecución
r - -	= 4	solo permiso de lectura
r - x	= 5	permisos de lectura y ejecución
r w -	= 6	permisos de lectura y escritura
r w x	= 7	todos los permisos establecidos, lectura, escritura y ejecución

Aplicado a las 3 identidades básicas del sistema en un listado de directorio (user, group, others):

Permisos	Valor	Descripción
rw-------	600	El propietario tiene permisos de lectura y escritura.
rwx--x--x	711	El propietario lectura, escritura y ejecución, el grupo y otros solo ejecución.
rwxr-xr-x	755	El propietario lectura, escritura y ejecución, el grupo y otros pueden leer y ejecutar el archivo.
rwxrwxrwx	777	El archivo puede ser leido, escrito y ejecutado por quien sea.
r--------	400	Solo el propietario puede leer el archivo, pero ni el mismo puede modificarlo o ejecutarlo y por supuesto ni elgrupo ni otros pueden hacer nada en el.
rw-r-----	640	El usuario propietario puede leer y escribir, el grupo puede leer el archivo y otros no pueden ha nada.

En la siguiente tabla vemos un resumen con las máscaras de permisos más utilizadas

chmod Value	Owner	Group	Public/Other
755	R-W-X	R-X	R-X
744	R-W-X	R	R
766	R-W-X	R-W	R-W
777	R-W-X	R-W-X	R-W-X

Comandos sudo y su

sudo: Este comando se utiliza para ejecutar un programa con permisos de otro usuario. En ocasiones un programa necesita ejecutarse con más permisos que los que posee el usuario invocador para poder acceder a ciertos recursos del sistema para el que no tiene permisos suficientes. Por ejemplo, si ejecutamos con nuestro usuario:

```
cat /etc/shadow
```

veremos que nos da un error de permiso denegado. Esto es porque ese archivo solo puede ser leído por root y los miembros del grupo shadow (haced un ls -la /etc para comprobarlo). Por tanto para poder visualizar el contenido de ese archivo podemos invocar el comando con permisos de root, utilizando para ello la orden sudo:

```
sudo cat /etc/shadow
```

veréis que ahora sí os dejará listar el contenido de ese archivo. El motivo es que al utilizar sudo se invoca el comando al que precede con privilegios de root. Sin embargo sudo no está restringido solamente a ejecutar comandos como root, también pueden usarse otros usuarios y grupos del sistema, para ello usamos los comandos

```
sudo -u usuario comando
sudo -g grupo comando
```

Esta sintaxis permitiría ejecutar comandos en nombre de un usuario o grupo especificados

El comportamiento de sudo está definido en el archivo /**etc**/**sudoers**, el cual debe ser solamente modificado mediante el comando **visudo** ejecutado como root (o mediante **sudo visudo** con un sudoer). La sintaxis de cada línea de asignación de privilegios es:

```
USERNAME/GROUPNAME SERVERNAME=(USERNAME_RUNAS:GROUPNAME_RUNAS) COMMAND
```

La línea:

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

Significa: El usuario **root** puede ejecutar desde cualquier host (primer **ALL**, antes del =), como cualquier usuario, segundo **ALL** (primero dentro del paréntesis), como cualquier grupo, tercer **ALL**, todos los comandos, último **ALL** de la línea. Cuando indiquemos una lista de privilegios de ejecución para miembro de un grupo la línea debe empezar por %

```
%admin ALL=(ALL:ALL) ALL
```

Los miembros del grupo admin pueden ejecutar cualquier comando

También es posible definir en el archivo sudoers **alias** para usuarios, máquinas comandos y runas. Los alias deben definirse en letra mayúscula (Ver **man sudo** o **sudo --help** para más información)

```
User_Alias  OFICINA1 = paco, luis, carlos
Cmnd_Alias  POWER = /sbin/shutdown, /sbin/halt, /sbin/reboot, /sbin/restart
Runas_Alias WEB = www-data, apache
```

Una vez definidos los alias puedo utilizarlos en las reglas de ejecución del archivo

```
OFICINA1 ALL = (WEB) ALL
```

Los miembros del grupo alias OFICINA1 pueden ejecutar como usuarios **www-data, apache** todos los comandos con permisos de ejecución para esos usuarios

En versiones recientes se puede conseguir que el usuario pueda **invocar a sudo** simplemente haciéndolo miembro del **grupo sudo**

Un comando interesante es **sudo -i** que permite iniciar una sesión interactiva de root sin necesidad de la invocación recursiva **sudo su**

su: Este comando se utiliza para cambiar el usuario de la sesión de consola actual: Sintaxis:

su [-] [usuario [argumentos]]

Si se pone el - se ejecutarán los archivos de inicio del usuario (para iniciación de variables de entorno por ejemplo), en caso de que no se ponga se mantendrán las de la sesión de usuario anterior al cambio. Si no se especifica el **usuario** entonces se cambiará por defecto al usuario root. En algunos sistemas, como Debian o Ubuntu, el usuario root viene deshabilitado por defecto, es decir no se podrá iniciar sesión con él. Esto se hace por motivos de seguridad. Los **argumentos** también son opcionales y se refieren a la shell, por ejemplo -c comando ejecutaría el comando indicado en nombre del usuario indicado

```
su -
```

el comando anterior cambiaría la sesión al usuario root, ejecutando sus archivos de inicio y situándolo en el home de root

Creación y Administración de Usuarios

Linux posee, como la mayoría de Sistemas, una gestión integrada de usuarios y grupos. El soporte en el Sistema Operativo para las operaciones de definición y mantenimiento de usuarios se basa en los archivos del Sistema siguientes:

/etc/passwd: Archivo de texto en el que se almacenan los usuarios del Sistema. Cada línea de este archivo representa la información relativa a un usuario determinado y su sintaxis es la siguiente: **usuario:contraseña:id:gid:descripcion:directorio_trabajo:shell** dónde:

- **usuario**: nombre del usuario
- **contraseña**: contraseña del usuario. Actualmente, por motivos de seguridad, las contraseñas no se almacenan en este archivo, sino en /etc/shadow. Por este motivo aparecerá una "x" en este campo
- **id**: id del usuario en el Sistema
- **gid**: id del grupo principal del usuario en el Sistema
- **descripción**: breve descripción
- **directorio_trabajo**: directorio home del usuario
- **shell**: programa de intérprete de comandos, o shell, asociado al usuario

/etc/shadow: Archivo de texto en el que se almacena la información de contraseñas (cifradas) de los usuarios. A diferencia de /etc/passwd este archivo solo puede ser leído por root, de modo que solamente el superusuario puede ver su contenido. Con esta medida se fortaleza la seguridad de los sistemas Linux. De nuevo hay una línea por cada usuario del sistema, de modo que toda línea en /etc/passwd tendría su correspondiente en

/etc/shadow. La sintaxis es: **usuario:contraseña:ultimo_cambio:min:max:aviso:inactivo:expiración:indicador** dónde:

- **usuario**: tiene el mismo significado que en /etc/passwd
- **contraseña**: contraseña del usuario en formato cifrado. Si el valor en este campo es **LK** indicará que la cuenta está bloqueada, si el campo está vacío se indicará que no hay contraseña para el usuario
- **ultimo_cambio**: número de días entre el 1 de enero de 1970 y el último cambio de contraseña
- **min**: número de días mínimo antes de cambiar contraseña
- **max**: número de días máximo de vigencia de la contraseña
- **aviso**: número de días antes de que expire la contraseña, para avisar al usuario
- **inactivo**: número de días permitidos de inactividad de la cuenta
- **expiración**: fecha en la que la cuenta ya no podrá ser usada
- **indicador**: para uso futuro, valor por defecto 0

Veréis que muchas de las líneas de este archivo tienen un **"' en el campo de contraseña. Esto indica que corresponden a cuentas de usuario con los que no se va a inicar sesión, como por ejemplo los usuarios que se crean al instalar programas, éstos tendrán los permisos mínimos necesarios para ejecutar el programa, pero no se iniciará sesión con ellos.** En caso de root, el cual en Ubuntu está deshabilitado como usuario de inicio de sesión por defecto, veréis que hay un "!" en el campo de la contraseña. Para poder iniciar sesión con root directamente en Ubuntu debemos asignarle una contraseña, para ello podríamos usar los comando

```
sudo su
passwd
```

Las líneas de /etc/shadow correspondientes a usuarios con inicio de sesión habilitado veremos que en el campo de la contraseña se almacena un código hash de la misma, identificado con los 2 caracteres iniciales, como por ejemplo $6, el cual indica SHA-512. Más concretamente:

Concentrándonos en la parte de la contraseña, tenemos que este hash se compone de tres partes separadas por el caracter $:

- **Id**: Identifica el método de encriptación utilizado: 1 para MD5; 2 o 2a para Blowfish; 3 para NT Hash; 5 para SHA-256; y 6 para SHA-512.
- **Salt**: Es utilizado por los algoritmos de encriptación, regularmente son 16 caracteres o más. El salt es una semilla de aleatorización utilizada con el propósito de que una misma contraseña en claro no genere siempre la misma cadena de caracteres cifrados, distintos salt producirán distintas versiones cifradas de la contraseña, lo cual dificulta los ataques de fuerza bruta.
- **Hash**: Esto es la "contraseña" (o hash). MD5 utiliza 22 caracteres, SHA-256 usa 43, y SHA-512 usa 86.

Veamos un ejemplo:

Supongamos que en /etc/shadow tenemos la línea:

```
oracle:$6$MvOuttAz$5ZUMuRkx8b2kGJ/jQvTszUQz73R1G9wM78kh1SogNRnSNARUtH9YFbRX/
E9iSkcokC4Djyo86DDj39Tq5ebw4/:15057:0:99999:7:::
```

Se interpretaría así:

- Id=6 (SHA-512)
- Salt=MvOuttAz
- Hash= 5ZUMuRkx8b2kGJ/jQvTszUQz73R1G9wM78kh1SogNRnSNARUtH9YFbRX/E9iSkcokC4Djyo86DDj39Tq5ebw4/

/etc/group: Archivo de texto en el que se definen los grupos de usuarios. Todo usuario tiene un grupo principal asociado, además de otros de los que también puede ser miembros. De nuevo, se define en cada línea del archivo un grupo, con la sintaxis siguiente: **grupo:gid:lista_usuarios** dónde:

- grupo: nombre del grupo
- gid: id de Sistema para el grupo
- lista_usuarios: usuarios del grupo separados por comas

Pasos para crear un usuario manualmente

- Crear el grupo en /etc/group
- Añadir el usuario en el fichero /etc/passwd y /etc/shadow, dejando la contraseña en blanco
- Ejecutar el comando passwd para asignarle contraseña al usuario
- Crear el directorio HOME del usuario y asignarle como propietario al usuario

Uso de comandos relacionados con la gestión de usuarios

Existen varios comandos en Linux para automatizar, o al menos simplificar, las tareas de creación de usuarios y grupos:

- **useradd**: Comando para añadir usuario especificando toda la información como parámetros

```
useradd -m -s /bin/bash prueba
```

El comando anterior crea el usuario prueba con shell /bin/bash y **creando también automáticamente su directorio home** (opción -m). Pueden verse más opciones de este comando ejecutando

```
useradd --help
```

- **groupadd**: Comando para añadir grupos especificando toda la información como parámetros

```
groupadd test
```

El comando anterior crea el grupo test (Más opciones con la opción **--help** del comando)

- **usermod**: Comando para modificar un usuario existente

```
usermod -a -G test prueba
```

El comando anterior añade al usuario **prueba** como miembro del grupo **test** Tras ejecutar el comando podemos comprobar que efectivamente el usuario se ha añadido al grupo con el comando

```
id prueba
```

El siguiente comando:

```
usermod -p $(openssl passwd -1 abc123.) prueba
```

establece la contraseña **abc123.** para el usuario prueba. Debemos pasar como parámetro la **contraseña cifrada**, en este caso utilizando el algoritmo MD5, al comando **usermod**, por ese motivo se utiliza el comando openssl para generarla. Para obtener ayuda adicional para creación de passwords del comando **openssl**

```
openssl passwd --help
```

- **vipw**: Comando que permite editar el archivo /etc/passwd de forma segura pues trabaja con una copia de seguridad
- **adduser**: Comando para **añadir usuarios pero de un modo asistido**, no es necesario especificar la información como parámetros, solamente el nombre del usuario. También se puede utilizar para establecer el grupo de un usuario con la sintaxis **adduser usuario grupo**. Adicionalmente permite la creación de grupos con la sintaxis **adduser --group nombre_grupo**. La opción **--system** indica que se creará un usuario del sistema, con inicio de sesión deshabilitado.
- **addgroup**: Comando para añadir grupos pero de un modo asistido, simplemente especificando el nombre del grupo
- **gpasswd**: Comando para añadir usuarios a grupos. Ejemplo (añade el usuario uadmin al grupo wireshark)

```
gpasswd -a uadmin wireshark
```

Sería equivalente a

```
usermod -G wireshark uadmin
```

Estos comandos añadirían el usuario al grupo pero no como grupo principal. Para cambiar el grupo principal del usuario podemos usar

```
usermod -g wireshark uadmin
```

El comando **gpasswd también se puede utilizar para eliminar a usuarios de grupos**

```
gpasswd -d usuario sudo
```

Elimina a usuario del grupo sudo

Eliminar usuarios

Podemos hacerlo de varios formas:

- Poniendo en la definición de shell el valor /bin/false
- Poniendo una contraseña inválida, como **LK**. Pensad que en /etc/shadow se almacenan las contraseñas cifradas, por lo tanto cualquier cosa que pongáis "a voleo" no coincidirá con ninguna contraseña "real"
- Eliminar al usuario eliminando la línea de /etc/passwd y /etc/shadow

Ahora bien, hay que tener en cuenta que, aunque eliminemos al usuario, todos sus archivos y directorios permanecen en el sistema. Por tanto mucho cuidado si tratáis de **reutilizar identificadores de usuario (UID)** pues todos esos archivos y directorios aparecerían asociados al nuevo usuario.

- Utilizar comandos como **userdel** para eliminar usuarios o **groupdel** para eliminar grupos (ejecutados con sudo)

El comando id

Este comando es útil para visualizar la información relativa a un usuario. La sintaxis es

```
id usuario
```

La salida del comando muestra el **uid, gid, y la lista de grupos adicionales del usuario**

El comando getent

Permite visualizar información de entorno relacionada con los usuarios del sistema. Si, por ejemplo el equipo está conectado a un dominio, permitirá ver, además de los usuarios locales, todos los usuarios y grupos del dominio. Veamos algún ejemplo de uso:

```
getent passwd
```

Muestra todas las cuentas de usuarios en el sistema

```
getent group
```

Muestra todas las cuentas de grupos en el sistema

Referencias

Tutoriales sobre sudo https://www.digitalocean.com/community/tutorials/how-to-edit-the-sudoers-file-on-ubuntu-and-centos

Ejemplos sudo http://www.courtesan.com/sudo/sample.sudoers

Calculador de permisos chmod: http://www.wwwwdot.com/chmod-calculator/

Volver

JavierFP 16:31 08 ene 2019 (CET)