ADMINISTRACIÓN DE USUARIOS Y GRUPOS DE LINUX LOCAL

RH124

Capitulo 5

Descripción general	
Meta	Administrar usuarios y grupos de Linux local y administrar directivas de contraseña locales.
Objetivos	• Explicar la función de los usuarios y grupos en un sistema Linux y cómo son entendidos por la computadora. • Ejecutar comandos como superusuario para administrar el sistema Linux. • Crear, modificar, bloquear y eliminar cuentas de usuario definidas a nivel local. • Crear, modificar y eliminar cuentas de grupo definidas a nivel local. • Bloquear cuentas en forma manual o mediante la configuración de una directiva de antigüedad de contraseña en el archivo de contraseña shadow.
Secciones	• Usuarios y grupos (y práctica) • Obtención de acceso de superusuario (y práctica) • Administración de cuentas de usuario local (y práctica) • Administración de cuentas de grupo local (y práctica) • Administración de contraseñas de usuario (y práctica)
Trabajo de laboratorio	• Administración de usuarios y grupos de Linux local

Usuarios y Grupos

RH124

Objetivos

Tras finalizar esta sección, los estudiantes deberían poder explicar el rol y cómo son entendidos, los usuarios y grupos en un sistema Linux.

¿Qué es un usuario?

Cada proceso (programa en ejecución) en el sistema se ejecuta como un usuario particular. Cada archivo es propiedad de un usuario particular. El acceso a los archivos y directorios está restringido por usuario. El usuario asociado con un proceso de ejecución determina los archivos y directorios accesibles para ese proceso.

El comando **`id`** se usa para mostrar información acerca del usuario con sesión iniciada actualmente. También se puede solicitar información básica de otro usuario pasando el nombre de usuario de dicho usuario como primer argumento al comando **`id`**.

```
[student@desktopX ~]$ id
uid=1000(student) gid=1000(student) groups=1000(student),10(wheel)
 context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Para ver el usuario relacionado con un archivo o directorio, use el comando **ls -l**. La tercera columna muestra el nombre de usuario:

```
[student@serverX ~]$ ls -l /tmp
drwx------. 2 gdm     gdm      4096 Jan 24 13:05 orbit-gdm
drwx------. 2 student student  4096 Jan 25 20:40 orbit-student
-rw-r--r--. 1 root    root    23574 Jan 24 13:05 postconf
```

Para ver la información del proceso, use el comando **ps**. La opción predeterminada es mostrar solo los procesos que están en la shell actual. Agregue la opción **a** para ver todos los procesos con un terminal. Para ver el usuario relacionado con un proceso, incluya la opción **u**. La primera columna muestra el nombre de usuario:

```
[student@serverX ~]$ ps au
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root       428  0.0  0.7 152768 14400 tty1     Ss+  Feb03   0:04 /usr/bin/Xorg
root       511  0.0  0.0 110012   812 ttyS0    Ss+  Feb03   0:00 /sbin/agetty
root      1805  0.0  0.1 116040  2580 pts/0    Ss   Feb03   0:00 -bash
root      2109  0.0  0.1 178468  2200 pts/0    S    Feb03   0:00 su - student
student   2110  0.0  0.1 116168  2864 pts/0    S    Feb03   0:00 -bash
student   3690  0.0  0.0 123368  1300 pts/0    R+   11:42   0:00 ps au
```

```
[student@serverX ~]$ ps au
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root       428  0.0  0.7 152768 14400 tty1     Ss+  Feb03   0:04 /usr/bin/Xorg
root       511  0.0  0.0 110012   812 ttyS0    Ss+  Feb03   0:00 /sbin/agetty
root      1805  0.0  0.1 116040  2580 pts/0    Ss   Feb03   0:00 -bash
root      2109  0.0  0.1 178468  2200 pts/0    S    Feb03   0:00 su - student
student   2110  0.0  0.1 116168  2864 pts/0    S    Feb03   0:00 -bash
student   3690  0.0  0.0 123368  1300 pts/0    R+   11:42   0:00 ps au
```

El resultado de los comandos anteriores muestra a los usuarios por nombre, pero internamente, el sistema operativo realiza el seguimiento de los usuarios por *número de UID*. La asignación de nombres a números se define en las bases de datos de la información de la cuenta. De forma predeterminada, los sistemas usan un "archivo plano o sin formato", el archivo **`/etc/passwd`**, para almacenar información sobre los usuarios locales. El formato de **`/etc/passwd`** es el siguiente (siete campos separados por dos puntos):

❶`username:`❷`password:`❸`UID:`❹`GID:`❺`GECOS:`❻`/home/dir:`❼`shell`

1. El *username* es una asignación de ID de usuario (UID) a un nombre para beneficio de los usuarios humanos.
2. *password* es donde se guardaban las contraseñas en formato cifrado tradicionalmente. Actualmente, se guardan en un archivo aparte con el nombre **`/etc/shadow`**.
3. *UID* es una ID de usuario, un número que identifica al usuario en el nivel más básico.
4. *GID* es el número de ID de grupo principal del usuario. Los grupos se analizarán más adelante.
5. El campo *GECOS* es un texto arbitrario que, por lo general, incluye el nombre real del usuario.
6. */home/dir* es la ubicación donde se encuentran los datos personales del usuario y los archivos de configuración.
7. La *shell* es un programa que se ejecuta cuando el usuario inicia sesión. Para un usuario habitual, por lo general, este es el programa que proporciona el aviso de línea de comando del usuario.

¿Qué es un grupo?

Al igual que los usuarios, los grupos tienen un nombre y un número (GID). Los grupos locales están definidos en **`/etc/group`**.

Grupos principales

- Cada usuario tiene exactamente un *grupo principal*.
- Para los usuarios locales, el grupo principal está definido por el número de GID del grupo indicado en el cuarto campo de **`/etc/passwd`**.
- Generalmente, el grupo principal es propietario de los nuevos archivos creados por el usuario.
- Normalmente, el grupo principal de un usuario creado recientemente es un grupo creado con el mismo nombre que el del usuario. El usuario es el único miembro de este *grupo privado de usuarios* (UPG).

Grupos suplementarios

- Los usuarios pueden ser miembros de ninguno o más *grupos adicionales*.
- Los usuarios que son miembros adicionales de grupos locales se enumeran en el último campo de la entrada del grupo en **`/etc/group`**: Para grupos locales, la membresía del usuario se determina por una lista de usuarios separados por comas que se encuentran en el último campo de la entrada del grupo en **`/etc/group`**:

```
groupname:password:GID:list,of,users,in,this,group
```

Referencias

Páginas del manual: **id**(5), **passwd**(5) y **group**(1)

info libc (*Manual de referencia de la biblioteca GNU C*)

- Sección 29: Usuarios y grupos

(Tenga en cuenta que el paquete *glibc-devel* se debe haber instalado para que estos nodos de información estén disponibles).

Práctica: Conceptos de usuario y grupo

Obtención de acceso de superusuario

RH124

Objetivos

Tras finalizar esta sección, los estudiantes deberían poder ejecutar comandos como superusuario para administrar un sistema Linux.

El usuario root

La mayoría de los sistemas operativos tienen una especie de *superusuario*, un usuario que tiene todo el poder sobre el sistema. Este usuario en Red Hat Enterprise Linux es el usuario **root**. Este usuario tiene el poder de anular los privilegios normales del sistema de archivos y se utiliza para manejar y administrar el sistema. Para poder realizar tareas, como la instalación o eliminación de software, y para administrar los directorios y los archivos del sistema, debe aumentar los privilegios al usuario **root**.

La mayoría de los dispositivos solo pueden ser controlados por el usuario **root**, pero existen algunas excepciones. Por ejemplo, los dispositivos desmontables, como los dispositivos USB, pueden controlarse mediante un usuario normal. Por lo tanto, se le permite a un usuario que no sea root que agregue y elimine archivos, y administre de otro modo un dispositivo desmontable, pero solo el usuario root puede administrar los discos duros "fijos" de manera predeterminada.

Sin embargo, este privilegio ilimitado viene acompañado de una responsabilidad. El usuario **root** tiene poder ilimitado para dañar el sistema: eliminar archivos y directorios, eliminar cuentas de usuarios, agregar puertas traseras, etc. Si la cuenta **root** está comprometida, alguien más tendrá control administrativo del sistema. A lo largo de este curso, se les indicará a los administradores que inicien sesión como usuario normal y que escalen los privilegios a **root** solo cuando sea necesario.

La cuenta **root** en Linux es casi equivalente a la cuenta de administrador local en Windows. En Linux, la mayoría de los administradores del sistema inicia sesión en una cuenta de usuario sin privilegios y utiliza distintas herramientas para obtener privilegios de usuario root temporalmente.

Advertencia

Una práctica habitual en Windows en el pasado es que el usuario administrador inicie sesión en forma directa para que realice las tareas de administrador del sistema. Sin embargo, en Linux se recomienda que los administradores de sistema *no* inicien sesión directamente como **root**. En su lugar, los administradores de sistema deben iniciar sesión como usuario no root y utilizar otros mecanismos (**su**, **sudo** o **PolicyKit**, por ejemplo) para obtener privilegios de superusuario temporalmente.

Mediante el inicio de sesión como usuario administrativo, todo el entorno de escritorio se ejecuta sin necesidad con privilegios administrativos. En esa situación, cualquier vulnerabilidad de la seguridad, que normalmente pudiera comprometer solo la cuenta del usuario, tiene el potencial de comprometer a todo el sistema.

En versiones recientes de Microsoft Windows, el administrador está inhabilitado de manera predeterminada y se usan funciones como el control de cuenta de usuario (UAC) para limitar los privilegios administrativos de los usuarios hasta que se necesiten. En Linux, el sistema **PolicyKit** es el equivalente más cercano a UAC.

Intercambio de usuarios con su

El comando **su** le permite al usuario cambiar a una cuenta de usuario diferente. Si no se especifica el nombre de usuario, se supone qe es la cuenta de usuario *root*. Al ser invocado como usuario común, aparecerá un aviso que le solicitará la contraseña de la cuenta a la que cambiará, mientras que al ser invocado como usuario *root*, no deberá ingresar la contraseña de la cuenta.

su [-] *<username>*

```
[student@desktopX ~]$ su -
Password: redhat
[root@desktopX ~]#
```

El comando **su *username*** inicia una *shell de no inicio de sesión*, mientras que el comando **su - *username*** inicia una shell de *inicio de sesión* . La diferencia principal es que **su -** establece el entorno de la shell como si iniciara la sesión como ese usuario, mientras que **su** simplemente inicia una shell como ese usuario con la configuración de entorno actual.

En la mayoría de los casos, los administradores quieren ejecutar **su -** para obtener la configuración normal del usuario. Si desea obtener más información, consulte la página del manual **bash**(1).

Ejecución de comandos como usuario root con sudo

Fundamentalmente, Linux implementa un modelo de permisos muy general: los usuarios *root* pueden realizar todo, mientras que los demás usuarios no pueden realizar nada (relacionado con el sistema). Una solución común es permitir que los usuarios estándares "se conviertan en usuarios *root*" temporalmente con el comando **su**. La desventaja es que, mientras sea un usuario *root*, se otorgan todos los privilegios (y las responsabilidades) de un usuario *root*. El usuario no solo puede reiniciar el servidor web, sino que también puede eliminar el directorio **/etc** completo. Además, todos los usuarios que requieran privilegios de superusuario de esta manera deben conocer la contraseña de usuario **root**.

El comando **sudo** permite al usuario ejecutar un comando como usuario root o como otro usuario, en función de la configuración del archivo **/etc/sudoers**. A diferencia de otras herramientas, como **su**, **sudo** requiere que un usuario ingrese su propia contraseña para la autenticación y no la contraseña de la cuenta a la que intenta acceder. Esto le permite a un administrador repartir los permisos específicos a los usuarios para delegar las tareas de administración del sistema sin tener que repartir la contraseña *root*.

Por ejemplo, cuando **sudo** se configura para permitir al usuario *student* ejecutar el comando **usermod** como *root*, el usuario *student* puede ejecutar el siguiente comando a fin de bloquear una cuenta de usuario:

```
[student@serverX ~]$ sudo usermod -L username
[sudo] password for student: password
```

Un beneficio adicional de usar **sudo** es que todos los comandos ejecutados con **sudo** se registran de manera predeterminada en **/var/log/secure**.

```
[student@serverX ~]$ sudo tail /var/log/secure
...
Feb 19 15:23:36 localhost sudo: student : TTY=pts/0 ; PWD=/home/student ; USER=root ;
 COMMAND=/sbin/usermod -L student
Feb 19 15:23:36 localhost usermod[16325]: lock user 'student' password
Feb 19 15:23:47 localhost sudo: student : TTY=pts/0 ; PWD=/home/student ; USER=root ;
 COMMAND=/bin/tail /var/log/secure
```

En Red Hat Enterprise Linux 7, todos los miembros del grupo **wheel** pueden usar **sudo** para ejecutar comandos como cualquier usuario, que incluye al usuario **root**. Se le pedirá al usuario que ingrese su propia contraseña. Este es un cambio con respecto a Red Hat Enterprise Linux 6 y las versiones anteriores. Los usuarios que fueron miembros del grupo **wheel** no obtuvieron este acceso administrativo de manera predeterminada en RHEL 6 y en versiones anteriores.

Para habilitar comportamientos similares en versiones anteriores de Red Hat Enterprise Linux, use **visudo** a fin de editar el archivo de configuración y eliminar el comentario de la línea que permite al grupo **wheel** ejecutar todos los comandos.

```
[root@desktopX ~]# cat /etc/sudoers
...Output omitted...
## Allows people in group wheel to run all commands
%wheel        ALL=(ALL)       ALL

## Same thing without a password
# %wheel  ALL=(ALL)       NOPASSWD: ALL
...Output omitted...
```

Advertencia

RHEL 6 no otorgó ningún privilegio especial al grupo **wheel** de manera predeterminada. Es probable que los sitios que estuvieron usando este grupo se sorprendan cuando RHEL 7 otorgue en forma automática y a todos los miembros de **wheel** privilegios totales de **sudo**. Esto podría provocar que usuarios no autorizados obtengan acceso de superusuario a los sistemas RHEL 7.

Históricamente, la membresía en el grupo **wheel** se ha usado por sistemas parecidos a Unix para otorgar o controlar el acceso como superusuario.

La mayoría de las aplicaciones de administración del sistema con un GUI usan **PolicyKit** para solicitar autenticación a los usuarios y administrar el acceso como usuario root. En Red Hat Enterprise Linux 7, **PolicyKit**también puede pedir a los miembros del grupo **wheel** su propia contraseña para obtener privilegios como **root** cuando usen herramientas gráficas. Esto es parecido a la forma en que pueden usar **sudo** para obtener esos privilegios en el aviso de la shell. **PolicyKit** otorga estos privilegios según sus propios parámetros de configuración, aparte de **sudo**. Es posible que los estudiantes avanzados estén interesados en las páginas del manual **pkexec**(1) y **polkit**(8) para obtener detalles sobre cómo funciona este sistema, pero eso está fuera del alcance de este curso.

Práctica: Ejecución de comandos como usuario root

Administración de cuentas de usuarios locales

RH124

Objetivos

Tras finalizar esta sección, los estudiantes deberían poder crear, modificar, bloquear y eliminar cuentas de usuarios definidas localmente.

Administración de usuarios locales

Se puede utilizar una serie de herramientas de la línea de comandos para administrar cuentas de usuarios locales.

useradd permite crear usuarios.

- **useradd *username*** define valores predeterminados razonables para todos los campos en **/etc/passwd** cuando se ejecuta sin opciones. El comando **useradd** no permite definir ninguna contraseña válida de manera predeterminada, y el usuario no puede iniciar sesión hasta que se defina una.

- **useradd --help** permite ver las opciones básicas que pueden usarse para anular los valores predeterminados. En la mayoría de los casos, las mismas opciones pueden usarse con el comando **usermod** para modificar un usuario existente.

- Algunos valores predeterminados, como el rango de números UID válidos y las reglas de vigencia de contraseñas predeterminadas, se leen desde el archivo **/etc/login.defs**. Los valores incluidos en este archivo solo se utilizan durante la creación de usuarios nuevos. Si se modifica dicho archivo, ningún usuario existente se verá afectado.

usermod permite modificar usuarios existentes.

- **usermod --help** mostrará las opciones básicas que pueden usarse para modificar una cuenta. Algunas opciones comunes incluyen las siguientes:

usermod opciones:	
-c, --comment COMMENT	Añadir un valor, como un nombre completo, al campo GECOS.
-g, --gid GROUP	Especificar el grupo principal para la cuenta del usuario.
-G, --groups GROUPS	Especificar una lista de grupos complementarios para la cuenta de usuario.
-a, --append	Se utiliza con la opción **-G** para anexar el usuario a los grupos complementarios mencionados sin quitarlo de otros grupos.
-d, --home HOME_DIR	Especificar un nuevo directorio de inicio para la cuenta de usuario.
-m, --move-home	Mover un nuevo directorio de inicio de usuario a una nueva ubicación. Debe usarse con la opción **-d**.
-s, --shell SHELL	Especificar una nueva shell de inicio de sesión para la cuenta de usuario.

usermod opciones:	
-L, --lock	Bloquear una cuenta de usuario.
-U, --unlock	Desbloquear una cuenta de usuario.

Advertencia

Cuando se elimina un usuario con **`userdel`** sin la opción **`-r`** especificada, el sistema tendrá archivos que pertenecen a un número de ID de usuario no asignado. Esto también puede suceder cuando los archivos creados por un usuario eliminado existen fuera de su directorio de inicio. Esta situación puede hacer que se filtre información y causar otros problemas de seguridad.

En Red Hat Enterprise Linux 7, el comando **`useradd`** asigna a los usuarios nuevos el primer número de UID disponible en el rango, a partir de la UID 1000 en adelante (a menos que se especifique uno explícitamente con la opción **`-u`** ***`UID`***). Es así como puede filtrarse información: si el primer número UID disponible ha sido asignado previamente a una cuenta de usuario que ha sido eliminada del sistema, el número de UID del usuario anterior se reasignará al nuevo usuario y le dará la propiedad de los archivos restantes del usuario anterior. A continuación se demuestra esta situación:

```
[root@serverX ~]# useradd prince
[root@serverX ~]# ls -l /home
drwx------. 3 prince  prince    74 Feb  4 15:22 prince
[root@serverX ~]# userdel prince
[root@serverX ~]# ls -l /home
drwx------. 3    1000    1000   74 Feb  4 15:22 prince
[root@serverX ~]# useradd bob
[root@serverX ~]# ls -l /home
drwx------. 3 bob     bob       74 Feb  4 15:23 bob
drwx------. 3 bob     bob       74 Feb  4 15:22 prince
```

Observe que **bob** es ahora propietario de todos los archivos que, en otra ocasión, pertenecían a **prince**. Según la situación, una solución a este problema es eliminar todos los archivos "que no pertenecen a nadie" del sistema cuando se elimina el usuario que los creó. Otra solución es asignar manualmente los archivos "que no pertenecen a nadie" a otro usuario. El usuario root puede encontrar los archivos y directorios "que no pertenecen a nadie" al ejecutar: **find / -nouser -o -nogroup 2> /dev/null**.

passwd permite definir las contraseñas.

- **passwd *username*** se puede usar para establecer la contraseña inicial o cambiar la contraseña del usuario.
- El usuario *root* puede definir una contraseña en cualquier valor. Aparecerá un mensaje si la contraseña no cumple con los criterios mínimos recomendados, seguido de un aviso para que vuelva a ingresar la contraseña nueva y todos los símbolos se actualizarán correctamente.

```
[root@serverX ~]# passwd student
Changing password for user student.
New password: redhat123
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary
 word
Retype new password: redhat123
passwd: all authentication tokens updated successfully.
```

- Un usuario regular debe elegir una contraseña que tenga al menos 8 caracteres y que no sea una palabra que figure en el diccionario, el nombre de usuario ni la contraseña anterior.

Rangos de UID

Red Hat Enterprise Linux utiliza números y rangos de números de UID específicos con fines específicos.

- *UID 0* siempre se asigna a la cuenta de superusuario: **root**.
- *UID 1-200* es un rango de "usuarios del sistema" que Red Hat asignó estadísticamente a procesos del sistema.
- *UID 201-999* es un rango de "usuarios del sistema" utilizado por procesos del sistema que no tienen archivos en el sistema de archivos. Por lo general, se asignan dinámicamente de la agrupación disponible cuando el software que los necesita está instalado. Los programas se ejecutan como estos usuarios del sistema "sin privilegios" para limitar el acceso que tienen a solo los recursos que necesitan para funcionar.
- *UID 1000+* es el rango disponible para la asignación a usuarios regulares.

nota

Antes de Red Hat Enterprise Linux 7, la convención consistía en que UID 1-499 se utilizaba para usuarios del sistema y UID 500+ para usuarios regulares. Los rangos predeterminados utilizados por **useradd** y **groupadd** pueden modificarse en el archivo **/etc/login.defs**.

Referencias

Páginas del manual: **useradd**(8), **usermod** (8), **userdel** (8)

Práctica: Creación de usuarios usando herramientas de la línea de comandos

Administración de cuentas de grupos locales

RH124

Objetivos

Tras finalizar esta sección, los estudiantes deberían poder crear, modificar y eliminar cuentas de grupos definidas localmente.

Administración de grupos adicionales

Para que un usuario pueda agregarse a un grupo, primero debe crearse el grupo. Se emplean diversas herramientas de la línea de comandos para administrar cuentas de grupos locales.

El comando crea grupos.groupadd

- **groupadd *groupname*** sin opciones emplea la siguiente GID disponible de un rango especificado en el archivo **/etc/login.defs**.
- La opción **-g *GID*** se utiliza para especificar una GID particular.

```
[student@serverX ~]$ sudo groupadd -g 5000 ateam
```

nota

Dada la creación automática de grupos privados de usuarios (GID 1000+), generalmente se recomienda establecer aparte un rango de números de GID para su uso con los grupos adicionales. Un rango más alto evitará una colisión con un grupo del sistema (GID 0-999).

- La opción **-r** creará un grupo del sistema usando una GID del rango de números de GID del sistema válido incluidos en el archivo **/etc/login.defs**.

```
[student@serverX ~]$ sudo groupadd -r appusers
```

El comando **groupmod** modifica grupos existentes.

- El comando **groupmod** se utiliza para cambiar el nombre de un grupo por una asignación de GID. La opción **-n** se usa para especificar un nombre nuevo.

```
[student@serverX ~]$ sudo groupmod -n javaapp appusers
```

- La opción **-g** se usa para especificar una GID nueva.

```
[student@serverX ~]$ sudo groupmod -g 6000 ateam
```

El comando **groupdel** elimina un grupo.

- El comando **groupdel** quita un grupo.

```
[student@serverX ~]$ sudo groupdel javaapp
```

- Es posible que un grupo no se quite si es el grupo principal de cualquier usuario existente. Como en el caso de **userdel**, controle todos los sistemas de archivos para asegurarse de que ningún archivo siga siendo propiedad del grupo.

El comando **usermod** modifica la pertenencia a grupos.

- La pertenencia a un grupo se controla con la administración de usuarios. Cambie el grupo principal de un usuario con **usermod -g *groupname***.

```
[student@serverX ~]$ sudo usermod -g student student
```

- Añada un usuario a un grupo adicional con **usermod -aG *groupname username***.

```
[student@serverX ~]$ sudo usermod -aG wheel elvis
```

Importante

El uso de la opción **-a** hace que **usermod** funcione en modo "adición". Sin esta, el usuario se eliminaría de *todos los demás* grupos adicionales.

Referencias

Páginas del manual: **group**(5), **groupadd**(8), **groupdel**(8) y **usermod**(8)

Práctica: Administración de grupos utilizando herramientas de línea de comandos

Administración de contraseñas de usuarios

RH124

Objetivos

Tras finalizar esta sección, los estudiantes deberían poder bloquear cuentas manualmente o definiendo una política de vigencia de contraseñas en el archivo de contraseña "shadow".

Contraseñas shadow y política de contraseñas

Hace muchos años, las contraseñas cifradas se almacenaban en el archivo /etc/passwd de lectura global. Se pensaba que esta ubicación era bastante segura hasta que los ataques de diccionarios a contraseñas cifradas se volvieron frecuentes. En ese momento, las contraseñas cifradas o "hashes de contraseña", se trasladaron al archivo /etc/shadow más seguro. Este nuevo archivo también permitió la implementación de características de vigencia y caducidad de la contraseña.

Un hash de contraseña moderno almacena tres datos:

`$1$gCjLa2/Z$6Pu0EK0AzfCjxjv2hoLOB/`

1. **`1`**: el algoritmo hash. El número 1 indica un hash MD5. El número 6 aparece cuando se usa un hash SHA-512.

2. **`gCjLa2/Z`**: el valor *aleatorio* utilizado para cifrar el hash. Originalmente, se elige al azar. El valor aleatorio y la contraseña no cifrada se combinan y se cifran para crear el hash de contraseña cifrado. El uso del valor aleatorio evita que dos usuarios con la misma contraseña tengan entradas idénticas en el archivo **`/etc/shadow`**.

3. **`6Pu0EK0AzfCjxjv2hoLOB/`**: el hash cifrado.

nota

Red Hat Enterprise Linux 6 y 7 admiten dos nuevos algoritmos de hash de contraseñas sólidos: SHA-256 (algoritmo **5**) y SHA-512 (algoritmo **6**). Tanto la cadena del valor aleatorio como el hash cifrado son más extensos para estos algoritmos. El usuario **root** puede cambiar el algoritmo predeterminado que se utiliza para hashes de contraseñas ejecutando el comando **authconfig --passalgo** con alguno de los argumentos **md5**, **sha256** o **sha512**, según corresponda.

Red Hat Enterprise Linux 7 utiliza el cifrado SHA-512 de manera predeterminada.

/etc/shadow formato

El formato de **/etc/shadow** es el siguiente (nueve campos separados por dos puntos):

```
❶name:❷password:❸lastchange:❹minage:❺maxage:❻warning:❼inactive:❽expire:❾blank
```

❶ El *nombre* de inicio de sesión. Debe ser un nombre de cuenta válido en el sistema.

❷ La *contraseña* cifrada. Si un campo de contraseña comienza con un signo de admiración, la contraseña está bloqueada.

❸ La fecha de la última *modificación de la contraseña*, que se representa como la cantidad de días desde 1970.01.01.

❹ La cantidad *mínima* de días que deben transcurrir para que una contraseña pueda modificarse; 0 significa "ningún requisito mínimo de vigencia".

❺ La cantidad *máxima* de días que deben transcurrir para que una contraseña deba modificarse.

❻ El período de *advertencia* de que una contraseña está a punto de caducar. Se representa en días; 0 significa que "no se proporciona ninguna advertencia".

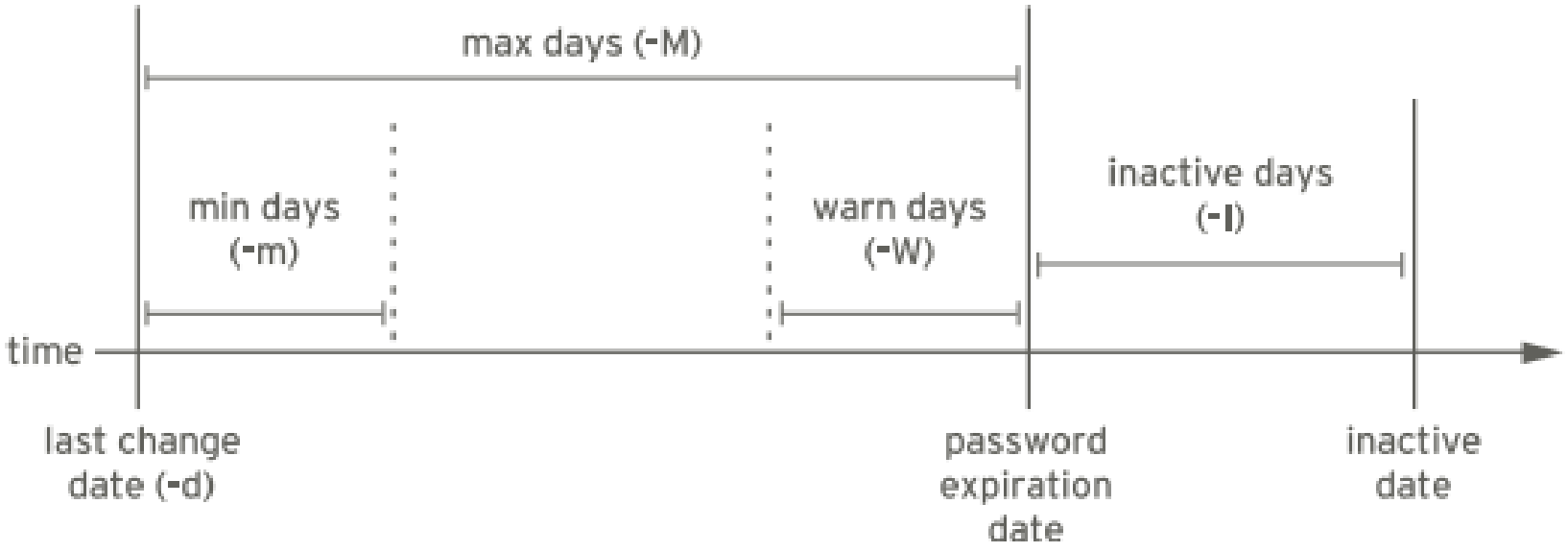
❼ La cantidad de días que una cuenta permanece activa después de que una contraseña caduca. Un usuario aún puede iniciar sesión en el sistema y modificar la contraseña durante ese período. Una vez transcurridos los días especificados, la cuenta se bloquea y se vuelve *inactiva*.

❽ La fecha de *caducidad* de la cuenta, que se representa como la cantidad de días desde el 1970.01.01.

❾ Este campo en *blanco* se reserva para su uso en el futuro.

Vigencia de contraseñas

En el siguiente diagrama, se indican los parámetros de vigencia de contraseñas relevantes que pueden ajustarse mediante **chage** para implementar una política de vigencia de contraseñas.




```
[root@serverX ~]# chage -m 0 -M 90 -W 7 -I 14 username
```

chage -d 0 ***username*** forzará que se actualice la contraseña en el próximo inicio de sesión.

chage -l ***username*** enumerará los valores de configuración actuales del nombre de usuario.

chage -E ***YYYY-MM-DD username*** expirará una cuenta un día específico.

nota

El comando **date** puede usarse para calcular una fecha en el futuro.

```
[student@serverX ~]$ date -d "+45 days"
Sat Mar 22 11:47:06 EDT 2014
```

Restricción del acceso

Con el comando **chage**, puede definirse la caducidad de una cuenta. Cuando se alcanza la fecha, el usuario no puede iniciar sesión en el sistema de manera interactiva. El comando **usermod** puede "bloquear" una cuenta con la opción **-L**.

```
[student@serverX ~]$ sudo usermod -L elvis
[student@serverX ~]$ su - elvis
Password: elvis
su: Authentication failure
```

Cuando un usuario se va de una empresa, el administrador puede bloquear una cuenta y determinar su caducidad con el comando **usermod** solamente. La fecha debe indicarse como la cantidad de días desde 1970.01.01.

```
[student@serverX ~]$ sudo usermod -L -e 1 elvis
```

El bloqueo de la cuenta evita que el usuario logre la autenticación con una contraseña en el sistema. Esta es la forma recomendada de evitar que un empleado que se fue de la empresa acceda a su cuenta. Si el empleado regresa, la cuenta puede desbloquearse con **usermod -U *USERNAME***. Si la cuenta también caducó, asegúrese de modificar, además, la fecha de caducidad.

La shell nologin

En ocasiones, un usuario necesita una cuenta con una contraseña para realizar la autenticación en un sistema, pero no necesita una shell interactiva en el sistema. Por ejemplo, un servidor de correo puede necesitar una cuenta para el almacenamiento de correo y una contraseña para que el usuario realice la autenticación con un cliente de correo utilizado para recuperar correo. Dicho usuario no debe iniciar sesión directamente en el sistema.

Ante una situación como la anterior, una solución común es definir la shell de inicio de sesión del usuario en **/sbin/nologin**. Si el usuario intenta iniciar sesión en el sistema directamente, la "shell" **nologin** simplemente cerrará la conexión.

```
[root@serverX ~]# usermod -s /sbin/nologin student
[root@serverX ~]# su - student
Last login: Tue Feb  4 18:40:30 EST 2014 on pts/0
This account is currently not available.
```

Importante

El uso de la shell **`nologin`** evita el uso interactivo del sistema, pero no evita todo el acceso. Un usuario puede, de todas maneras, realizar la autenticación y cargar o recuperar archivos a través de aplicaciones, como aplicaciones web, programas de transferencia de archivos o lectores de correo.

Referencias

Páginas del manual: **`chage`**(8), **`usermod`**(5), **`shadow`**(3), **`crypt`**(1)

Práctica: Administración de la antigüedad de la contraseña de usuario