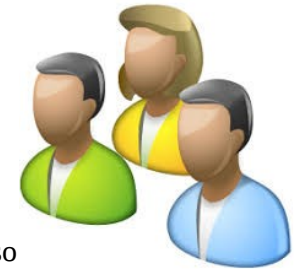
3. GESTIÓN DE USUARIOS Y GRUPOS LOCALES



3.1.- Introducción

Los sistemas Unix/Linux son sistemas multiusuario que aseguran el uso del sistema de forma segura y ordenada. Para poder utilizar un sistema Unix/Linux es necesario disponer de una cuenta de usuario que se compone principalmente de un nombre de usuario o *login* y una contraseña o *password*. Las cuentas de usuario son creadas por el usuario administrador *root*.

Otros componentes de una cuenta de usuario son su grupo primario (grupo obligatorio al que un usuario debe pertenecer), grupos secundarios (grupos extra al que el usuario pertenece), una ruta donde el usuario almacena sus documentos, normalmente un subdirectorio dentro de */home*, un intérprete de comandos o *shell* que le permitirá ejecutar aplicaciones y un uid (identificación de usuario con el que el sistema identifica al usuario de forma única).

Cuando un usuario ejecuta una aplicación, el sistema la carga en memoria y la ejecuta, llamándose proceso. Cada proceso en el sistema se ejecuta como un usuario en concreto y cada archivo o directorio pertenece a un usuario. El acceso a archivos y directorios está restringido por usuario y el usuario asociado a un proceso determina qué archivos y directorios son accesibles para el proceso.

Con el comando *ps aux* veremos los procesos en ejecución del sistema con el usuario que lo ejecutó.

Con el comando *id* sin parámetros vemos la información del usuario con el que estamos conectados (*uid*, *gid* y grupos a los que pertenece) y con *id* <usuario> la información de ese usuario en concreto.

3.2.- Usuarios locales

En el proceso de instalación estándar de las distribuciones Linux como CentOS 7, se crea el usuario *root* y se le da contraseña. En este proceso existe la posibilidad de crear más usuarios pero se suele hacer ya con el sistema en funcionamiento, desde el usuario *root* usando el comando *useradd.*

3.2.1.- Tipos de usuarios locales:

- **Usuario root:** también llamado superusuario o usuario administrador. Su UID es 0 y es la única cuenta con privilegios totales en el sistema. Tiene acceso total a archivos, directorios y procesos independientemente de los permisos que estos tengan. Puede instalar software, ejecutar tareas de mantenimiento del sistema, …

- **Usuarios de sistema**: son usuarios utilizados para ejecutar ciertos procesos de sistema, no tienen todos los privilegios de root pero pueden asumir algunos. No suelen tener home ni contraseña de acceso ya que no inician sesión en el sistema. Se suelen crear en la instalación del sistema o de los paquetes. En CentOS 7 tienen UID de 1-200 para los usuarios que se asignan de forma estática o de 201-999 para procesos de sistema que no tienen archivos en los sistemas de archivos. Usuarios de sistema: *bin, apache, daemon,...*
- **Usuarios regulares**: usuarios individuales. Sólo tienen privilegios totales en su home que está situado bajo /home. En CentOS 7 sus UIDs tienen valores a partir de 1000.

En el archivo */etc/login.defs* están definidos estos rangos de usuarios que se pueden modificar.

3.2.2.- Archivo /etc/passwd

Por defecto, el sistema usa el archivo plano */etc/passwd* para almacenar la información de los usuarios locales. Este archivo almacena en una línea la información de un usuario usando 7 campos separados por el carácter dos puntos (:) con el formato:

<username>:<password>:<uid>:<gid>:<gecos>:<home_dir>:<shell>

donde:

<*username*>: es el nombre del usuario en el sistema. No debe contener mayúsculas.

<*password*>: password del usuario encriptada. Actualmente en este campo aparece una x ya que las password encriptadas están en el archivo */etc/shadow* junto con su caducidad. Si aparece un asterisco, el usuario no se podrá logar en el sistema y si aparece vacío, el usuario podrá entrar al sistema sin necesidad de introducir password.

<*uid*>: identificador único del usuario en el sistema.

<*gid*>: identificador único del grupo primario al que pertenece el usuario.

<*gecos*>: campo optativo usado para almacenar comentarios que suele conteter el nombre completo del usuario.

<*home_dir*>: directorio home del usuario. Si es un usuario de sistema, puede estar vacío.

<*shell*>: intérprete de comandos a utilizar por el usuario, p.e. */bin/bash*. Si aparece */bin/nologin* el usuario no podrá hacer *login* en el sistema.

Este archivo tiene permiso de lectura para todos los usuarios, ya que hay muchos comandos que lo consultan pero sólo tiene permiso de escritura para el usuario root.

No sería recomendable modificar este archivo con un editor de textos, salvo utilizando el comando *vipw* que utiliza bloqueos para que no se puedan hacer modificaciones en el archivo si lo tenemos abierto.

También los comandos *useradd, userdel* y *usermod* permiten hacen las modificaciones necesarias. Existe el comando *pwck* que verifica la integridad del archivo */etc/passwd* y */etc/shadow*.

3.2.3.-Crear usuarios

Como usuario root utilizaremos el comando ***useradd <opciones> <username>*** que añade una línea en el archivo */etc/passwd* para el usuario con *login* <*username*>, añade una línea en el archivo */etc/shadow* y se modifica el archivo */etc/group*. Se pueden añadir las opciones de:

-c <comentario>, --comment <comentario>: para poner valor al campo *gecos*.

-g <grupo>, --gid <grupo>: para especificar un grupo primario en concreto con *gid* o nombre de grupo. El grupo debe existir previamente. Si no se especifica esta opción, por defecto se creará un grupo con el mismo nombre que el *username* y será el grupo primario del usuario.

-b <base_dir>, --base-dir <base_dir>: se especifica el directorio base del directorio home del usuario, si no se añade la opción *-m*, el directorio debe existir. Si no se usa esta opción, por defecto es */home*.

-d <home_dir>, --home-dir <home_dir>: directorio home del usuario bajo el <base_dir>. No debe existir previamente ya que el comando lo crea.

-e <yyyy-mm-dd>, --expiredate <yyyy-mm-dd>: fecha de expiración de la cuenta.

-G <nombre_grupo1>,<nombre_grupo2>,..., --groups <nombre_grupo1>,<nombre_grupo2>,...: lista de grupos suplementarios a los que pertenece el usuario que deben existir previamente.

-r, --system: crea un usuario de sistema, dándole un *uid* por debajo de 1000. No crea el home del usuario salvo que se utilize la opción -m.

-s <shell>, --shell <shell>: shell que utilizará el usuario, p.e. */bin/bash, /bin/sh, /bin/nologin*.

-u <num>, --uid<num>: si se quiere asignar un *uid* en concreto al usuario.

NOTA: En el archivo */etc/default/useradd* están especificadas los valores por defecto de los diferentes campos si no se especifican con la opción correspondiente.

3.2.4.- Modificar usuarios

Con el usuario *root* y el comando ***usermod <opciones> <username>*** podemos modificar las propiedades del usuario <*username*> almacenadas en */etc/passwd* y */etc/group* y bloquear/desbloquear usuarios. Opciones más comunes:

-c <comentario>, --comment <comentario>: para modificar el valor al campo *gecos*.

-g <grupo>, --gid <grupo>: para modificar el grupo primario en concreto. El grupo debe existir previamente.

-G <grupo1>,<grupo2>,..., --groups <grupo1>,<grupo2>,...: Para sustituir los grupos suplementarios del usuario. Si lo que queremos es añadir grupos secundarios al usuario a los que ya tiene, hay que añadir la opción -a. Esto hace modificaciones en */etc/group*.

-d <home_dir>, --home-dir <home_dir>: modifica el directorio home del usuario. De existir previamente salvo que utilicemos además la opción *-m* que mueve el contenido del home actual al nuevo home y si no existe lo crea.

-l <nuevo_username>, --login <nuevo_username>: modifica el *username* del usuario. Hay que tener precaución con esta opción ya que no modifica el nombre del directorio home del usuario ni el de correo.

-s <shell>, --shell <shell>: modifica la shell que utilizará el usuario.

-L, --lock: bloquea la cuenta del usuario. Esto modifica el archivo */etc/shadow* añadiendo el carácter ! al principio del segundo campo.

-U, --unlock: desbloquea la cuenta del usuario. Esto modifica el archivo */etc/shadow* quitando el carácter ! al principio del segundo campo.

3.2.5.- Borrar usuarios

Con el usuario *root* y el comando ***userdel <username>*** se elimina el usuario del sistema. Esto implica que se borra la línea correspondiente del archivo */etc/passwd* y del archivo */etc/shadow* y se elimina dicho usuario de los grupos en */etc/group*. No se elimina el home del usuario salvo que se añada la opción *-r* o *--remove.*

Los archivos y directorios pertenecientes al usuario que estén en otros sistemas de archivos, permanecen en el sistema pero al haber desaparecido el usuario, en lugar de aparecer en su owner el nombre del usuario y en el grupo el nombre del grupo, aparece el *uid* y gid respectivamente. Dejar estos archivos sin eliminar del sistema, es peligroso ya que si posteriormente creamos un usuario con ese mismo *uid* y un grupo con ese *gid*, los archivos de forma automática se les asignarán.

Tal y como se indica en el man del comando *userdel*, es interesante modificar en */etc/login.defs* la variable *USERDEL_CMD* para que apunte a un script que se ejecute al borrar un usuario del sistema y que dicho script haga una búsqueda por todos los sistemas de archivos del sistema de los archivos y directorios del usuario y los elimine además de eliminar también cualquier tarea *at* (*find /var/spool/cron/atjobs -name "[^.]*" -type f -user <usuario> -delete \;*), tarea *cron* (*crontab -r -u <usuario>*) y de impresión (*lprm <usuario>*) que estén pendiente.

3.2.6.- Caso práctico

En ***server1*** vamos a modificar que los usuarios regulares comiencen en 1100 en lugar de 1000 al igual que en los grupos regulares; crear los usuarios regulares con *username* visitor*1* y *visitor2* y crear un usuario de sistema con *username jboss* que no pueda hacer *login* en el sistema y con *home* en */opt/jboss*.

Luego se borrará el usuario *visitor2* eliminando su *home*.

RESOLUCIÓN:

- Comenzamos modificando el valor en el que empezarán a asignarse usuarios y grupos regulares en */etc/login.defs*:

```
[root@server1 ~]#  sed -i 's/UID_MIN                1000/UID_MIN                1100/g' /etc/login.defs
[root@server1 ~]#  sed -i 's/GID_MIN                1000/GID_MIN                1100/g' /etc/login.defs
[root@server1 ~]#  useradd -c 'Usuario visitante 1' visitor1
[root@server1 ~]# useradd -c 'Usuario visitante 2' visitor2
[root@server1 ~]#  tail -2 /etc/passwd
visitor1:x:1100:1100:Usuario visitante 1:/home/visitor1:/bin/bash
visitor2:x:1101:1101:Usuario visitante 2:/home/visitor2:/bin/bash
```

- Creamos el usuario de sistema *jboss*:

```
[root@server1 ~]#  useradd -r -c 'Usuario de sistema para JBoss' -s /bin/nologin -b /opt jboss
[root@server1 ~]#  tail -2 /etc/passwd
visitor2:x:1101:1101:Usuario visitante 2:/home/visitor2:/bin/bash
jboss:x:995:993:Usuario de sistema para JBoss:/opt/jboss:/bin/nologin
```

- Elimino el usuario *visitor2*:

```
[root@server1 ~]#  userdel -r visitor2
[root@server1 ~]#  tail -2 /etc/passwd
visitor1:x:1100:1100:Usuario visitante 1:/home/visitor1:/bin/bash
jboss:x:995:993:Usuario de sistema para JBoss:/opt/jboss:/bin/nologin
```

3.3.- Gestionar contraseñas

Con el comando ***passwd*** cualquier usuario puede modificar su propia contraseña de acceso al sistema. Sólo el usuario *root* podrá modificar las contraseñas de otros usuarios usando ***passwd <usuario>*** siendo <*usuario*> el usuario al que *root* quiere cambiarle la contraseña.

En los usuarios regulares, existen unas restricciones para elegir contraseña: al menos 8 caracteres, no puede estar basada en una palabra existente en el diccionario, no utilizar el *username*, no usar

una contraseña anterior, …

Estas restricciones están definidas en el archivo */etc/login/defs*. El usuario *root* se salta estas restricciones pudiendo asignar cualquier contraseña a los usuarios.

Este comando pide dos veces por la entrada estándar la introducción de la nueva contraseña. Existe una opción del comando, utilizada para scripts, que permite pasarle la contraseña por línea de comandos *--stdin* pero su uso está totalmente desaconsejado ya que la contraseña se ve en texto claro en el terminal y se queda almacenada en el histórico del usuario. Su uso sería: *echo <nueva_pass> | passwd --stdin <usuario>*.

3.3.1.- Archivo /etc/shadow

Las contraseñas se almacenan encriptadas en el archivo */etc/shadow*. A este archivo sólo el usuario *root* tiene acceso ya que no tiene permisos para hacer absolutamente nada en el.

Por cada usuario existe una línea en este archivo, 9 campos separados por el carácter :, de la forma:

<username>:<password>:<ultimo_cambio>:<min>:<max>:<aviso>:<inactiva>:<expirada>:<blanco>

donde:

<*username*>: nombre de usuario en el sistema

<*password*>: contraseña encriptada. Si el primer carácter es ! significa que el usuario está bloqueado.

Si aparece !! significa que el usuario no tiene contraseña asignada y si aparece * significa que el usuario es un usuario *no-login* (usuarios de sistema utilizados para procesos que no se loguearán nunca), en ambos casos, ese usuario no podrá hacer *login* en el sistema.

<*ultimo_cambio*>: días trascurridos desde el 1 de enero de 1970 hasta que se cambió la contraseña por última vez. Si aparece un 0, significa que el usuario debe cambiar la contraseña en el siguiente *login*. Si está vacío, significa que la funcionalidad de caducidad de la contraseña está desactivada.

<*min*>: número mínimo de días transcurridos desde que se puso la última contraseña para que pueda ser cambiada. Si aparece un cero, no hay número mínimo de días.

<*max*>: número máximo de días antes de que una contraseña tenga que cambiarse obligatoriamente. Trascurridos estos días, obligará a un cambio de contraseña en el siguiente *login*. Si aparece un -1, no hay número máximo de días. Si *max* es menor que *min*, el usuario no podrá cambiar la contraseña.

<*aviso*>: días antes de que una contraseña expire en los cuales se avisa al usuario cuando hace *login*. Si aparece un cero, no hay aviso.

<*inactiva*>: número de días tras la expiración de una contraseña antes de que se bloquee. En este periodo se pueden entrar con el usuario y la contraseña pero obliga a cambiarla. Puede estar vacío.

<*expirada*>: días pasados desde el 1 de enero de 1970 hasta la fecha de expiración de la cuenta. Si está vacío significa que la cuenta nunca expira.

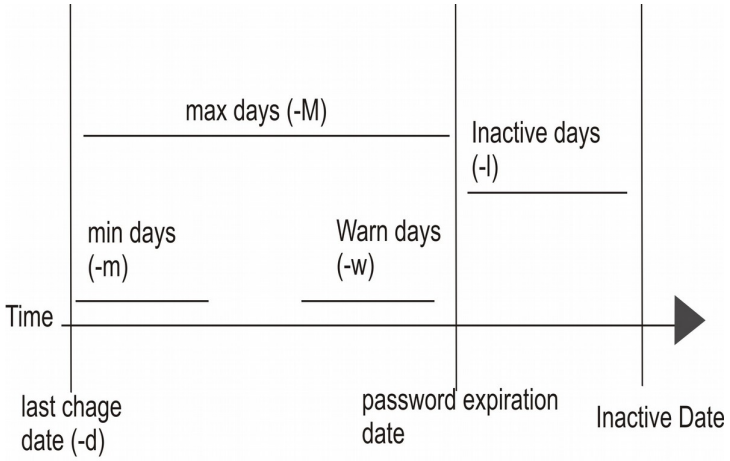
<*blanco*>: campo en blanco reservado para usos futuros.

Los usuarios a los que no se les ha asignado contraseña nunca, el campo *password* aparece con !!.

No sería recomendable modificar este archivo con un editor de textos, salvo utilizando el comando *vipw -s* que utiliza bloqueos para que no se puedan hacer modificaciones en el archivo si lo tenemos abierto. También los comandos *passwd* y *chage* permiten hacen las modificaciones necesarias. Existe el comando *pwck* que verifica la integridad del archivo */etc/passwd* y */etc/shadow*.

3.3.2.- Gestionar caducidad de la contraseña de un usuario

En el siguiente diagrama se ven los parámetros de caducidad que se ajustan con el comando *chage*.



Parámetros del comando chage

Con el comando ***chage <opciones> <username>*** podemos ajustar la caducidad de la contraseña de un usuario. Sólo puede ser ejecutado como *root*. Las opciones a usar:

-d <dia>, --last-day <dia>: días desde 1 de enero de 1970 cuando la contraseña fue cambiada. Se puede poner una fecha con el formato *yyyy-mm-dd* o poner 0 para obligar a cambiar la contraseña en el siguiente *login*.

-l: lista la configuración de caducidad de contraseña actual del usuario.

-E <dia>, - -expiredate <dia>: días desde 1 de enero de 1970 cuando expirará la contraseña. Se puede poner una fecha con el formato *yyyy-mm-dd*. Si se pone -1, borrará la fecha de

expiración de la cuenta.

-*I* <*dias*>, --*inactive* <*dias*>: días de inactividad tras caducar la contraseña (pasados los max-days).

-*m*<*dias*>, --*mindays* <*dias*>: días a transcurrir desde el último cambio de contraseña para poder volver a cambiarla.

-*M* <*dias*>, --*maxdays* <*dias*>: días que puede durar una contraseña.

-*W* <*dias*>, --*warndays* <*dias*>: días antes de que caduque la contraseña que avisa al usuario al hacer *login*.

3.3.3.- Caso prático

En ***server1***, al usuario *visitor1* creado anteriormente, se le asigna la contraseña de acceso al sistema *passvisitor*. Obligamos a que la primera vez que el usuario haga *login*, cambie la contraseña, que debe cambiar cada 60 días, dando un aviso tres días antes y permitiendo un periodo de gracia de cinco días. El usuario podrá cambiar la contraseña siempre que quiera y la cuenta del usuario expirará el 31 de diciembre de 2017.

RESOLUCIÓN:

- Asigno contraseña por línea de comandos al usuario *visitor1:*

```
[root@server1 ~]# echo 'passvisitor' | passwd visitor1 --stdin
```

- Veo la caducidad de la contraseña que tiene por defecto el usuario antes de hacer cambios:

```
[root@server1 ~]# chage -l visitor1
Last password change                                    : Aug 21, 2017
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
```

- El usuario debe cambiar la contraseña en el primer *login*, cambiarla cada 60 días, con aviso tres días antes de caducar, con 5 días de gracia y que la fecha de expiración de la cuenta sea 31 de diciembre de 2017:

```
[root@server1 ~]# chage -d 0  -M 60 -W 3 -I 5 -E 2017-12-31 visitor1
```

- Veo la caducidad de la contraseña tras los cambios:

```
[root@server1 ~]# chage -l visitor1
Last password change                                    : password must be changed
```

```
Password expires                                    : password must be changed
Password inactive                                   : password must be changed
Account expires                                     : Dec 31, 2017
Minimum number of days between password change      : 0
Maximum number of days between password change      : 60
Number of days of warning before password expires   : 3
```

Una vez que el usuario ha entrado al sistema y ha establecido una nueva contraseña:

```
[admin@server1 ~]$ su - visitor1
Password:                          ← passvisitor
You are required to change your password immediately (root enforced)
Changing password for visitor1.
(current) UNIX password:           ← passvisitor
New password:                      ← Vi23Valencia
Retype new password:               ← Vi23Valencia
```

la caducidad de la contraseña queda así:

```
[root@server1 ~]#  chage -l visitor1
Last password change                                : Aug 21, 2017
Password expires                                    : Oct 20, 2017
Password inactive                                   : Oct 25, 2017
Account expires                                     : Dec 31, 2017
Minimum number of days between password change      : 0
Maximum number of days between password change      : 60
Number of days of warning before password expires   : 3
```

Y la línea correspondiente en el archivo */etc/shadow*:

```
[root@server1 ~]# cat /etc/shadow | grep visitor1
visitor1:$6$OVk.0nh9$OJYYSs.75WQQzoCcUYTW9QmS0Z9sgpCkl5eAaOfiT8VqHuXx/iOS5hz7q
Xairo4qAO3O73QiGLtkI.4.XDekm1:17399:0:60:3:5:17531:
```

3.4.- Grupos locales

Un grupo debe existir antes de que un usuario pueda ser añadido a él, salvo el grupo primario del usuario que se crea por defecto con el mismo nombre que el del usuario, al usar *useradd* sin usar la opcion *-g*.

Con el comando *groupmems -g <nombre_grupo> -l* obtengo una lista con los usuarios que pertenecen a ese grupo.

3.4.1.- Archivo /etc/group

Al igual que el archivo */etc/passwd* almacena la información de los usuarios locales, el archivo */etc/group* almacena la información de los grupos locales del sistema.

La información de un grupo es una línea de 4 campos separados por el carácter dos puntos (:) con el formato:

<groupname>:<password>:<gid>:<username1>,<username2>,..., <uernameN>

donde:

<*groupname*>: es el nombre del grupo en el sistema. No debe contener mayúsculas.

<*password*>: contraseña del grupo encriptada. No se suele utilizar, aparece vacío.

<*gid*>: identificador único del grupo en el sistema.

<username1>,...: lista de usuarios que pertenecen al grupo separados por comas. Puede estar vacía bien porque es un grupo primario o porque aún no se le han asignado usuarios.

Este archivo tiene permiso de lectura para todos los usuarios, ya que hay muchos comandos que lo consultan pero sólo tiene permiso de escritura para el usuario *root*.

Al igual que ocurre con el archivo */etc/passwd*, no es recomendable modificar este archivo con un editor de textos, salvo utilizando el comando *vigr* que utiliza bloqueos para que no se puedan hacer modificaciones en el archivo si lo tenemos abierto. También los comandos *groupadd*, *groupdel* y *groupmod* permiten hacen las modificaciones necesarias. Con el comando *grpck* chequeamos la integridad del archivo */etc/group*.

3.4.2.- Crear grupos

Con el comando ***groupadd <opciones> <nombre_grupo>*** creamos un nuevo grupo secundario en el sistema. Las opciones usadas:

-g <gid>: dar un gid en concreto, opción nada recomendada, es mejor dejar que el sistema los asigne de forma automática para que no haya colisiones en los *gid*.

-r: para crear un grupo de sistema que tendrá *gid* por debajo de 1000.

Una vez creado el grupo, añado un usuario al grupo con el comando *useradd -aG <grupo> <usuario>*. Importante el uso de la opción *-a* junto con *-G* para que añada a los grupos secundarios a los que ya pertenece el usuario, este nuevo grupo. Si sólo usamos *-G*, eliminamos el usuario de TODOS los grupos secundarios a los que pertenece y lo añadimos al nuevo grupo. Esto no afecta a la pertenencia al grupo primario.

3.4.3.- Modificar grupos

Con el comando *groupmod <opciones> <nombre_grupo>* modifico las propiedades del grupo como pueden ser nombre del grupo o *gid*. Opciones usadas:

-g <gid>: cambio el *gid* del grupo. Importante saber que no se hacen comprobaciones en el archivo */etc/login.defs* para los parámetros GID_MIN, GID_MAX, SYS_GID_MIN, SYS_GID_MAX con lo que es posible que estemos infringiendo esos rangos establecidos. Los archivos/directorios que tuviesen el antiguo *gid*, deben ser cambiados de forma manual al nuevo *gid*.

-n <nuevo_nombre>: para cambiar el nombre de grupo.

3.4.4.- Borrar grupos

Se utiliza el comando *groupdel <nombre_grupo>*. El grupo no será borrado si es un grupo primario de algún usuario. Al igual que con el comando *userdel*, hay que asegurarse de que en los sistemas de archivos del sistema no existen archivos que pertenezcan al grupo.

3.4.5.- Caso práctico

En **server1** se van a crear una serie de grupos para los distintos departamentos de la empresa: *desarrollo*, *sistemas*, *redes* y *seguridad*. Existe un director por departamento que además de pertenecer al grupo de su departamento, pertenece al grupo *directiva*.

También se van a crear los usuarios pertenecientes a los departamentos, usando como *username* la inicial del nombre seguida del apellido. La distribución de los usuarios por departamentos, siendo el primer usuario que aparece el director, es:

- *desarrollo*: Ada Lovelace, Grace Hopper.
- *redes*: Hedy Lamarr, Tim Berners, Robert Kahn.
- *seguridad*: Alan Turing y Tatu Ylonen.
- *sistemas*: Linus Torvalds, Dennis Ritchie.

A todos los usuarios se les dará la contraseña de *changeme* que deberán cambiar en el primer *login*.

RESOLUCIÓN:

Creamos los grupos de los departamentos utilizando un bucle *for* de Bash:

```
[root@server1 ~]#  for dpto in {desarrollo,redes,seguridad,sistemas,directiva};do
                                        groupadd $dpto;done
[root@server1 ~]# tail -5 /etc/group
desarrollo:x:1101:
redes:x:1102:
```

```
seguridad:x:1103:
sistemas:x:1104:
directiva:x:1105:
```

Creamos los usuarios de los departamentos:

```
[root@server1 ~]#  useradd -G desarrollo -c 'Ada Lovelace' alovelace
[root@server1 ~]#  useradd -G desarrollo -c 'Grace Hopper' ghopper
[root@server1 ~]#  useradd -G redes -c 'Hedy Lamarr' hlamarr
[root@server1 ~]#  useradd -G redes -c 'Tim Berners' tberners
[root@server1 ~]#  useradd -G redes -c 'Robert Kahn' rkahn
[root@server1 ~]#  useradd -G seguridad -c 'Alan Turing' aturing
[root@server1 ~]#  useradd -G seguridad -c 'Tatu Ylonen' tylonen
[root@server1 ~]#  useradd -G sistemas -c 'Linus Torvalds' ltorvalds
[root@server1 ~]#  useradd -G sistemas -c 'Dennis Ritchie' dritchie
```

Añadimos los directores al grupo directiva:

```
[root@server1 ~]#  for dir in {alovelace,hlamarr,aturing,ltorvalds};do
                       usermod -aG directiva $dir;done
[root@server1 ~]# tail -14 /etc/group | head -5
desarrollo:x:1101:alovelace,ghopper
redes:x:1102:hlamarr,tberners,rkahn
seguridad:x:1103:aturing,tylonen
sistemas:x:1104:ltorvalds,dritchie
directiva:x:1105:alovelace,hlamarr,aturing,ltorvalds
```

Les pongo a todos la contraseña *changeme* y obligo a que la cambien en el primer *login*:

```
[root@server1 ~]# for dpto in {desarrollo,redes,seguridad,sistemas,directiva}; do
for user in $(groupmems -g $dpto -l);do echo changeme | passwd --stdin $user;
chage -d 0 $user;done;done
```

3.5.- Obteniendo acceso root

El usuario *root* es el usuario con privilegios totales en el sistema. Para realizar determinadas tareas, como instalar paquetes, gestionar usuarios, etc., un usuario necesita escalar privilegios.

Se puede obtener privilegios de root de dos formas, bien cambiado del usuario con el que hayamos hecho *login* al usuario *root* usando el comando *su* o bien ejecutando el comando privilegiado con *sudo*.

3.5.1.- Comando su

Permite a un usuario hacer *login* con otro usuario del mismo sistema. Si no se especifica usuario, se cambia al usuario root. Si se ejecuta con un usuario regular, preguntará la contraseña del usuario al que queremos pasar pero si se ejecuta como *root*, pasaremos a hacer *login* con ese usuario sin necesidad de meter la contraseña del nuevo usuario.

Hay dos formas de hacer *login* a otro usuario:

- *no-login shell*: usando el comando *su* <*usuario*>. Con esto en el nuevo usuario heredamos las variables de entorno y lo especificado en los archivos de inicialización del usuario de origen en lugar del usuario al que hemos cambiado.
- *login shell*: usando *su -* <*usuario*> o *su -l* <*usuario*> o *su –login* <*usuario*>. Así cargaremos todo el entorno del nuevo usuario, es decir, se cargan los archivos *.bashrc* y *.bash_profile* del usuario al que queremos cambiar. Esta opción es la recomendable.

3.5.2.- Comando sudo

Con el comando sudo permitimos ejecutar ciertas cosas con privilegios de otro usuario sin necesidad de conocer la contraseña de dicho usuario. Se suele usar para delegar tareas de administración del sistema sin proporcionar la contraseña de *root*. El comando *sudo* pide la contraseña del propio usuario.

Sudo proporciona un control muy exhaustivo de que puede hacer un usuario con permisos de otro además de registrar todo lo que hace un usuario usando sudo en el archivo */var/log/secure*.

→ Uso de sudo

Para usar sudo, se ejecuta:

```
sudo -u <usuario> <comando>
```

donde <*usuario*> *e*s el usuario con el que se quiere ejecutar el comando <*comando*>. Si no se pone usuario, se hará como usuario *root*.

Sudo con redirecciones y/o pipes no funciona de forma adecuada, en su lugar hay que ejecutar con *sudo* el comando ***bash -c*** y a continuación el comando/s con redirecciones y/o pipes. Por ejemplo, introducir una nueva linea en el archivo */etc/hosts:*

```
sudo bash -c “echo '10.11.1.101 server1 server1.example.com' >> /etc/hosts”
```

Opciones interesantes del comando *sudo*:

-l: veo los comandos que puedo utilizar con *sudo* con el usuario con el que he hecho *login*.

-e <*archivo*>*:* me permite editar un archivo como si fuese root.

-V: veo las opciones por defecto establecidas para todos los usuarios, comandos, equipos, ...

→ Configuración de sudo

Para configurar sudo se modifica el archivo principal */etc/sudoers* o se añade un archivo con el mismo formato a */etc/sudoers.d*/. Estos archivos se deben editar con el comando *visudo* ejecutado como *root* que bloquea el archivo para que sólo haya un usuario modificándolo además de hacer una verificación de la sintaxis del archivo antes de guardar los cambios. Si hubiese errores de sintaxis, tenemos tres opciones:

e: editar de nuevo el archivo

x: salir sin guardar los cambios

Q: guardar los cambios y salir.

Para verificar la sintaxis de *sudoers* sin hacer cambios, basta con hacer *visudo -c* para que chequee */etc/sudoers* y *visudo -c -f <archivo>* en el caso de otro archivo.

Existe un grupo predefinido llamado *wheel* cuyos usuarios pueden usar sudo para ejecutar cualquier comando como si fuesen root. En CentOS 7/RHEL 7 la configuración para que los miembros del grupo *wheel* puedan ejecutar comandos privilegiados en el */etc/sudoers* ya viene por defecto activada, no así en las versiones anteriores que habría que descomentar la línea:

```
# %wheel ALL=(ALL)   NOPASSWD: ALL
```

El archivo *sudoers* está basado en la forma EBNF que describe la gramática de un lenguaje que se va creando a través de reglas de producción que a la vez son la base para ser referenciadas por otras reglas.

Existen tres secciones, ninguna obligatoria, en el archivo /*etc/sudoers*:

Alias: engloban bajo un nombre varios usuarios, equipos o comandos. Cada línea tiene el formato

```
<tipo_alias> <NOMBRE_ALIAS> = elemento1, elemento2, elemento3, ... elementoN
```

donde:

<*tipo_alias*> es *Cmnd_Alias* (alias de comandos), *User_Alias* (alias de usuarios normales), *Runas_Alias* (alias de usuarios administradores o con privilegios) y *Host_Alias* (alias de hosts).

<*NOMBRE_ALIAS*>: formado por caracteres alfanuméricos o _ y debe comenzar por mayúscula. Se suelen poner en mayúsculas.

<*elemento1*>, ….: dependiendo del tipo de alias que sea serán comandos, *usernames*, %gruponames, *hostnames*, IPs, …

Opciones o *defaults*: permiten definir ciertas características de comportamiento para los alias previamente creados, para usuarios, usuarios privilegiados, para equipos o de manera global para todos.

Reglas de acceso: definen qué usuarios ejecutan qué comandos bajo que usuario y en qué equipos. Se configura la regla en una línea con el formato:

<usuario> <host> = (<usuario_ejecucion>:<grupo_ejecucion>)<comando>

donde:

<*usuario*>*:* puede ser un *username* o un alias de usuario. También puede aparecer un *groupname* anteponiendo el carácter !.

<*host*> puede ser ALL (cualquier equipo), un hostname, un alias de equipo, una dirección IP o una definición de red IP/máscara.

<*usuario_ejecución*> *y* <*grupo_ejecución*>*:* usuario y/o grupo con el que se ejecutará el comando. Los grupos se indican poniendo el carácter % delante del *groupname*. Si no aparece, el usuario/grupo de ejecución será *root*.

<*comando*>,...: cualquier comando dando su ruta completa. Si se termina en '/' indica todos los archivos dentro de ese directorio. Se pueden poner varios comandos separados por comas. Si el comando aparece entre comillas dobles, quiere decir que se permite el uso del comando pero sin parámetros ni argumentos.

A los comandos se puede anteponer una etiqueta que modifica como se ejecuta el comando. La etiqueta se hereda en una lista de comandos. Las posibles etiquetas son:

- ***NOPASSWD* y *PASSWD***: no requiere contraseña y requiere contraseña respectivamente. Por defecto si no aparece es PASSWD.
- *NOEXEC* y *EXEC*: importante cuando el comando permite escapes a shell, p.e. vi usando !. Con NOEXEC no permitimos que esto suceda.
- *SETENV* y *NOSETENV:* permite al usuario cambiar el entorno de variables a las del usuario con el que se ejecutará el comando.

→ Ejemplos de reglas:

admin ALL=(ALL:ALL) ALL

El usuario con *username admin* en cualquier *host*, puede ejecutar cualquier comando de cualquier usuario incluyendo a *root*.

fperez ALL = /sbin/iptables

El usuario con username *fperez* en cualquier *host* puede utilizar *iptables*.

ADMIN ALL = ALL

Los usuarios definidos en el alias *ADMIN* desde cualquier *host* pueden ejecutar cualquier comando.

%developers dbserver = (jboss) /opt/jboss/bin/standalone, (root) /var/log/*

Los usuarios que pertenezcan al grupo con *groupname developers* pueden en el sistema con *hostname dbserver* ejecutar como si fueran el usuario *jboss* el comando *standalone* y además como usuario *root* pueden ver los archivos del directorio */var/log*.

agarcia ALL = /usr/bin/passwd *, !/usr/bin/passwd root

El usuario con *username agarcia* desde cualquier sistema puede cambiar la contraseña a cualquier usuario excepto al usuario *root*.

flopez ALL = "/sbin/lsmod"

El usuario con *username flopez* puede ejecutar el comando *lsmod* pero sin argumentos luego sólo podrá listar los módulos del kernel.

pedro webserver = NOPASSWD: /bin/kill, /usr/bin/lprm, PASSWD:/etc/httpd/conf/, NOEXEC:/usr/bin/vi

El usuario con username *pedro* en el sistema con hostname *webserver* no necesita contraseña para los comandos *kill*, *lprm*, permite modificar todos los archivos que están en el directorio */etc/httpd/conf* necesitando contraseña y no permite ejecutar shell dentro del comando *vi*.

3.5.3.- Caso práctico

En **server1** vamos a permitir al usuario *admin* que pueda cambiar las password de cualquier usuario excepto *root*:

A los usuarios del grupo *redes*, les permitiremos usar los comandos de red *ip*, *ss*, *nmcli* y *firewall-cmd*, además de acceso a los archivos de red en */etc/sysconfig/network-scripts/*.

A los usuarios del grupo *sistemas*, permisos totales de *root* en el sistema.

RESOLUCIÓN

- Para permitir al usuario *admin* que pueda cambiar las contraseñas de todos los usuarios excepto *root*, en la parte de reglas del archivo */etc/sudoers* (editándolo con *visudo)* añado la línea:

  admin ALL = /usr/bin/passwd *, !/usr/bin/passwd root

- Para permitir a los usuarios del grupo *redes* que puedan ejecutar *ip*, *nmcli* y *firewall-cmd* y acceso al directorio, en la parte de alias del archivo *sudoers*, añado la línea:

  Cmnd_Alias RED=usr/sbin/ip,/usr/bin/nmcli,/usr/sbin/ss,/usr/bin/firewall-cmd, /etc/sysconfig/network-scripts/

y en la parte de reglas:

```
%sistemas ALL=RED
```

- Para dar acceso root a los usuarios del grupo *sistemas*, basta con añadirlos al grupo *wheel* y verificar que la configuración en el */etc/sudoers* para *wheel* está descomentada:

```
[root@server1 ~]# for user in $(groupmems -g sistemas -l);do usermod -aG wheel $user;done
[root@server1 ~]# less /etc/sudoers | grep wheel
## Allows people in group wheel to run all commands
%wheel        ALL=(ALL)    ALL
```