Laboratorio de
Sistemas Operativos

Práctica 5: SUDO

Autor: Ing. Raúl Fuentes Samaniego
Revisores: Jose I. Icaza

ÍNDICE

Comandos a utilizar

La siguiente es una lista de comandos claves para esta práctica. Aunque algunos se manejarán de forma breve se recomienda que realices una investigación de los mismos:

- useradd
- groupadd
- usermod
- id
- sudo
- chmod
- chown

Manejo de cuentas de usuario

Para esta práctica ya habrás realizado previamente la creación de cuentas y el manejo de UGO para las carpetas y archivos. Habrás creado diferentes carpetas y archivos para diversos usuarios y grupos, permitiendo que algunos usuarios puedan ejecutar el contenido pero restrigiéndoles en su capacidad de modificarlo; o bien que solo puedan leer el contenido, o que no puedan interactuar de ninguna manera con el archivo o carpeta. Pero: **¿qué pasa si queremos que el usuario pueda ejecutar algo que puede alterar los registros del sistema?** Por ejemplo, permitir que cambie la configuración de red, o el montaje o desmontaje de unidades físicas, entre otros. Para que un usuario distinto a Root

pueda realizar esto, hay que otorgarle los suficientes privilegios, agregándolo a algún grupo que tenga autorización para realizar estos cambios; o bien, permitir que el usuario abandone esa cuenta y pase temporalmente a ROOT para realizar estas operaciones.

El modo más ideal podría ser el permitir que el usuario pertenezca a un grupo con los privilegios necesarios, pero resulta complejo encuadrarlo en un marco de seguridad fiable ya que si se maneja mal, el usuario podría poseer demasiados privilegios para otros elementos de la máquina. Con esta perspectiva, obviamente usar a ROOT también resulta inseguro ya que el administrador debe responder a lo que haga este usuario (y cada uno de los que ocupen ROOT cotidianamente) o bien arriesgar la seguridad entregando su contraseña.

Escenario

Durante esta primera parte de la práctica nos enfocaremos en darle diferentes privilegios a usuarios comunes para que ellos puedan tener un control un poco más a modo sin requerir el uso de la cuenta Root. Suponga el siguiente escenario:

> Bob posee un equipo de cómputo bastante importante que utiliza en la oficina, pero simultáneamente es utilizado por sus dos hijos que van a la universidad y por dos empleados de él (un administrador de redes de nombre Delgado, y otro de diseño de procesadores de nombre Vasconcelos, que pertenece al mismo grupo que Bob). Por lo mismo maneja un grupo general de familia y dos grupos de trabajo denominado: WK_IT en el que se encuentra Delgado y otras personas, y WK_MK al que pertenecen Bob mismo y Vasconcelos. **C**omo sus empleados tienen sus propios equipos, se pueden conectar remotamente a la computadora de Bob. De igual manera los hijos pueden conectar a la compu de Bob con sus equipos, o bien usar directamente la computadora de su papá cuando no la esté usando.

Lo anterior se puede representar en la siguiente tabla:

Usuario	**Familia**	**WK_IT**	**WK_MK**
Esteban (Hijo)	Si	No	No
María (Hija)	Si	No	No
Bob	Si	No	Si
Delgado (admin de redes)	No	Si	No
Vasconcenlos (Colega)	No	No	Si

Aunque **Bob** es dueño del equipo de cómputo **debe cumplir con una política administrativa** que involucra lo siguiente:

- El administrador de redes requiere poder acceder vía remota
- Las contraseñas de los miembros del departamento de desarrollo de software (wk-it)

deben ser cambiadas cada 3 meses.

- La contraseña del administrador de redes debe ser compleja y cambiar cada 90 días.
- La cuenta de de Vasconcelos es meramente temporal y expira el 17 de marzo del presente año. Además, el no requiere una carpeta de HOME
- Todos los empleados deben de usar BASH como Shell predeterminado.

Los hijos, al no formar parte de la política de seguridad resultan más sencillos y solo requieren pertenecer al grupo **miFamilia** (y especificar su carpeta Home) pero al igual que el padre tendrán derecho a entrar a la carpeta **/miFamilia/** para tener un repositorio en común de documentos y archivos de toda la familia.

Con el escenario definido empezaremos a trabajar con ello, primero con la creación de las cuentas y posteriormente otorgándoles los privilegios y limitaciones ya mencionadas.

Configuración de usuarios

Primero procederemos a crear los 3 grupos de forma similar a la práctica de administración:

```
# groupadd familia
# groupadd wk_it
# groupadd wk_mk
```

Una vez creado los grupos **crearemos las cuentas** de usuario con los requisitos indicados pero **no deben olvidar que se están solicitando tiempos de expiración de contraseñas y de vida de las cuentas**. Bien, toda cuenta creada en sistemas basados en Unix es finita al igual que su tiempo de vida. Para poder corroborarlo debemos revisar diferentes archivos. En el caso de Linux es necesario consultar el archivo **/etc/login.defs** del cual mostramos una captura parcial:

```
#
# Password aging controls:
#
#       PASS_MAX_DAYS   Maximum number of days a password may be used.
#       PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#       PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7
```

En esta captura toda las cuentas creadas duran más de 273 años, para motivos prácticos tienen una vida eterna si Bob y su compañía (y la humanidad…) todavía existen. Si nosotros cambiamos el valor de esa línea impactará a *todas las cuentas de usuario* que se creen. Sin embargo esta no es la opción más adecuada para nuestro escenario ya que hay cuentas con diferente tiempos de expiración, así que tenemos que buscar otra alternativa.

NOTA: Los tiempos mínimos de cambio de contraseña pueden resultar

contraproducentes si se está intentando recuperar control de una cuenta robada.

Exploremos ahora todas las opciones que poseemos al crear un usuario. La siguiente tabla muestra los argumentos y opciones aceptadas por el comando **useradd**:

```
#useradd --help
Usage: useradd [options] LOGIN

Options:
  -b, --base-dir BASE_DIR        base directory for the home directory of the
                                 new account
  -c, --comment COMMENT          GECOS field of the new account
  -d, --home-dir HOME_DIR        home directory of the new account
  -D, --defaults                 print or change default useradd configuration
  -e, --expiredate EXPIRE_DATE   expiration date of the new account
  -f, --inactive INACTIVE        password inactivity period of the new account
  -g, --gid GROUP                name or ID of the primary group of the new
                                 account
  -G, --groups GROUPS            list of supplementary groups of the new
                                 account
  -h, --help                     display this help message and exit
  -k, --skel SKEL_DIR            use this alternative skeleton directory
  -K, --key KEY=VALUE            override /etc/login.defs defaults
  -l, --no-log-init              do not add the user to the lastlog and
                                 faillog databases
  -m, --create-home              create the user's home directory
  -M, --no-create-home           do not create the user's home directory
  -N, --no-user-group            do not create a group with the same name as
                                 the user
  -o, --non-unique               allow to create users with duplicate
                                 (non-unique) UID
  -p, --password PASSWORD        encrypted password of the new account
  -r, --system                   create a system account
  -s, --shell SHELL              login shell of the new account
  -u, --uid UID                  user ID of the new account
  -U, --user-group               create a group with the same name as the user
  -Z, --selinux-user SEUSER      use a specific SEUSER for the SELinux user mapping
```

Como puedes observar, este comando, ya utilizado en la práctica anterior, ofrece una respuesta aceptable para nuestro requerimiento de fecha de expiración (cuando el usuario deja de tener permiso para usar el sistema) pero no para el manejo de las contraseñas; por lo tanto analizaremos ahora el comando **passwd** que manipula el archivo /etc/passwd que también ya fue utilizado en la práctica pasada para cambiar o asignar contraseñas .

```
# passwd --help
Usage: passwd [options] [LOGIN]
Options:
  -a, --all                      report password status on all accounts
  -d, --delete                   delete the password for the named account
  -e, --expire                   force expire the password for the named account
  -h, --help                     display this help message and exit
  -k, --keep-tokens              change password only if expired
  -i, --inactive INACTIVE        set password inactive after expiration
                                 to INACTIVE
  -l, --lock                     lock the password of the named account
  -n, --mindays MIN_DAYS         set minimum number of days before password
                                 change to MIN_DAYS
  -q, --quiet                    quiet mode
  -r, --repository REPOSITORY    change password in REPOSITORY repository
```

```
  -S, --status                  report password status on the named account
  -u, --unlock                  unlock the password of the named account
  -w, --warndays WARN_DAYS      set expiration warning days to WARN_DAYS
  -x, --maxdays MAX_DAYS        set maximum number of days before password
                                change to MAX_DAYS
```

(Fragmento de **man passwd**)

```
 -S, --status
       Display account status information. The status information consists
       of 7 fields. The first field is the user's login name. The second
       field indicates if the user account has a locked password (L), has no
       password (NP), or has a usable password (P). The third field gives
       the date of the last password change. The next four fields are the
       minimum age, maximum age, warning period, and inactivity period
       for the password. These ages are expressed in days.
```

De estos argumentos uno muy importante es **-S** o **--status** que despliega el estado de la contraseña; y que añadido al argumento **--all,** nos puede generar una salida similar a la siguiente:

```
# passwd -S -a
root L 08/06/2012 0 99999 7 -1
daemon L 04/25/2012 0 99999 7 -1
bin L 04/25/2012 0 99999 7 -1
sys L 04/25/2012 0 99999 7 -1
sync L 04/25/2012 0 99999 7 -1
games L 04/25/2012 0 99999 7 -1
man L 04/25/2012 0 99999 7 -1
lp L 04/25/2012 0 99999 7 -1
mail L 04/25/2012 0 99999 7 -1
news L 04/25/2012 0 99999 7 -1
uucp L 04/25/2012 0 99999 7 -1
proxy L 04/25/2012 0 99999 7 -1
www-data L 04/25/2012 0 99999 7 -1
backup L 04/25/2012 0 99999 7 -1
list L 04/25/2012 0 99999 7 -1
irc L 04/25/2012 0 99999 7 -1
gnats L 04/25/2012 0 99999 7 -1
nobody L 04/25/2012 0 99999 7 -1
libuuid L 04/25/2012 0 99999 7 -1
syslog L 04/25/2012 0 99999 7 -1
messagebus L 04/25/2012 0 99999 7 -1
colord L 04/25/2012 0 99999 7 -1
lightdm L 04/25/2012 0 99999 7 -1
whoopsie L 04/25/2012 0 99999 7 -1
avahi-autoipd L 04/25/2012 0 99999 7 -1
avahi L 04/25/2012 0 99999 7 -1
usbmux L 04/25/2012 0 99999 7 -1
kernoops L 04/25/2012 0 99999 7 -1
pulse L 04/25/2012 0 99999 7 -1
rtkit L 04/25/2012 0 99999 7 -1
speech-dispatcher L 04/25/2012 0 99999 7 -1
hplip L 04/25/2012 0 99999 7 -1
saned L 04/25/2012 0 99999 7 -1
Bob P 08/06/2012 0 99999 7 -1
```

Por lo mismo la mayor parte de las cuentas están protegidas al no permitir que un usuario pueda simular ser ellas (la mayoría de estas cuentas corresponden en realidad a “usuarios-aplicación” que vimos en la práctica pasada).

En conclusión, para poder cumplir con los requisitos básicos al crear las cuentas de los usuarios, debemos utilizar opciones adicionales en los comandos **useradd**, **groupadd** y **passwd**.

UGO (avanzado)

Durante la práctica pasada se vió cómo manejar los permisos ***U(ser) G(roups) O(ther)*** o UGO, con el cual se pueden controlar los permisos de control de lectura, escritura y ejecución de todos los usuarios. El mejor ejemplo es ROOT quien posee todos los permisos para todo lo que se halle en el sistema; todos los demás usuarios generalmente logran sus permisos mediante los grupos a los que se hallan inscritos.

Pero existen ocasiones en que el uso de UGO se vuelve un poco más complejo y requiere más

seguridad. Para muestra veamos una de las necesidades del administrador de redes Delgado y la cual consiste en utilizar un *sniffer* denominado Wireshark. Este programa debe de realizar algo atípico en el Sistema Operativo: se trata de capturar los paquetes de datos que le llegan por una o más interfaz física y desplegar dicha información. Lo normal es que el S.O., usando sus propias rutinas (que no pueden ejecutar los usuarios) entregue los paquetes que van para cada programa sin permitir intervención de terceros, por motivos de seguridad. Pero un administrador de redes puede requerir utilizar este programa debido a los requisitos de su propio trabajo.

Por lo mismo, el usuario que ejecuta ese programa (o el mismo programa que tendrá su propio usuario) debe tener los privilegios necesarios para la captura de paquetes.

Para poder cumplir de forma segura y eficiente este requisito para los administradores de red, se realizará la instalación de la aplicación, y a continuación se alterarán ciertos elementos para alterar los privilegios de UGO en ellos. Específicamente trabajaremos en un subprograma llamado "Dumpcap" que ya viene instalado en Linux. Ese subprograma es el encargado de capturar los paquetes del S.O. y pasarlo a la aplicación Wireshark. Para estos cambios tenemos dos opciones:

```
                                                OPCIÓN 1
# groupadd wireshark
# usermod -a -G wireshark delgado
# chgrp wireshark /usr/bin/dumpcap
# chmod 4750 /usr/bin/dumpcap

                                                OPCIÓN 2
# sudo apt-get install libcap2-bin
# groupadd -g wireshark
# usermod -a -G wireshark delgado
# chmod 750 /usr/bin/dumpcap
# setcap cap_net_raw,cap_net_admin=eip /usr/bin/dumpcap
```

> **NOTA**: La diferencia entre las dos opciones es que la primera es un mecanismo tradicional de modificar UGO y la segunda utiliza una capacidad granular relativamente nueva de los Sistemas GNU/Linux (implementada en Ubuntu posterior al 2010). En todo caso la segunda opción se puede realizar mediante el mecanismo:
> **`dpkg-reconfigure wireshark-common`**

Durante la práctica utilizaremos la primera opción,. Debe notar que el usuario **Delgado** no recibe ningún nuevo privilegio, sino que se añade al grupo al cual el programa dumpcap ahora pertenece (y es este último quien sufrió cambios en sus privilegios).

Substitute User DO

Mientras que las configuraciones de UGO nos pueden permitir eventualmente ejecutar todo con los privilegios más altos (o una mezcla de ello) exige un tiempo y conocimientos elevados para lograrlo. Una alternativa altamente empleada en sistemas GNU/Linux basados en las distribuciones de Debian es el denominado "**SUDO**" a veces referido como "super-user DO"

aunque su término correcto es “Substitute User, Do!”.

SUDO permite que usuarios que se encuentren en el archivo **/etc/sudoers** puedan tomar temporalmente privilegios de otros usuarios y/o grupos para la ejecución de ciertos programas instalados en el sistema. La sintaxis usada en el archivo es estricta **y por lo mismo no se utilizan directamente los editores de texto como NANO o GEDIT**; en su lugar se utiliza un editor de propósito específico denominado: **visudo**.

```
root@usuario-VirtualBox:/home/usuario# whatis visudo
visudo (8)              - edit the sudoers file

root@usuario-VirtualBox:/home/usuario# ls -l /etc/sudoers
-r--r----- 1 root root 723 Jan 31  2012 /etc/sudoers
```

Es muy recomendable **editar /etc/sudoers desde la cuenta Root** para poder recuperar el acceso a ese archivo o incluso volverlo a crear si queda mal configurado ya que un mal manejo puede derivar en que quede bloqueada para siempre dicha cuenta.

El archivo **/etc/sudoers** por defecto tiene la siguiente sintaxis:

```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

Toda línea que inicie con el símbolo gato “ **#** “ es un comentario, por lo mismo podemos ver que se intenta separar el archivo en las porciones **“Alias”** ” y “**privilegios especificados para cada los usuarios**”. Estos *alias* son básicamente variables y los últimos son especificaciones directas para los usuarios, aunque ya hay 3 configurados por defecto.

Los alias se separan en "**User**", "**Runas**", "**Host**" y "**Cmdn**" (**runas** no aparece por defecto en **/etc/sudoers**) y en general permiten el agrupar en una lista identificada por un nombre (primera letra mayúscula siempre) y cuyo contenido depende del tipo de alias.

Con "**user alias**" se trata de identificar usuarios (o grupos identificados con el signo " **%** " del sistema ya sea mediante sus grupos, el GId, uID o el dominio al que pertenecen. En el siguiente ejemplo identificamos en 3 alias del tipo usuario a los usuarios creados hasta ahora.

```
##
# User alias specification
# % indica que es un grupo (existente en /etc/group) en vez de un usuario
# (Existente en /etc/passwd )
##
User_Alias  FAMLIARES  =  %familia
User_Alias  COLABORADORES  = bob, vasconcelos
User_Alias  ADMINISTRADORES = delgado
```

El caso de "**runas alias**" es idéntico al anterior en sintaxis pero su objetivo es identificar cual es el usuario que ejecutará el comando dado por otro usuario. Un ejemplo sería que María ejecute un comando como si fuese Bob y en cuyo caso Bob estaría definido en un alias de runas.

```
##
# RunAs alias specification
##
Runas_Alias     OP = root
Runas_Alias DUENIO = bob #quien es el propietario de la maquina
Runas_Alias NETADMIN = delgado
```

El siguiente alias "**host alias**" define hosts en la red, por lo mismo puede agrupar nombres de dominios de hosts o directamente las direcciones IPv4 o IPv6 e incluso bloques de estas direcciones. Para este laboratorio no haremos uso de estos alias.

El último tipo de alias, el de comandos o "**cmnd_alias**" tiene como objetivo identificar y agrupar los diferentes comandos que queremos permitir que se ejecuten por los usuarios. Dichos comandos deben ser escritos con sus direcciones absolutas (El comando **`whereis`** puede ayudarles si no recuerdan la ubicación del comando a añadir en el alias).

```
##
# cmnd alias specification
##
Cmnd_Alias      REDES = /sbin/ifconfig, /sbin/route ,  /sbin/dhclient
```

Finalmente vienen las especificaciones. La primera es la de usuarios y se indica qué usuarios pueden ejecutar ciertos comandos bajo el manto de otro usuario en ciertas máquinas (hosts). Lo podemos hacer directamente o bien utilizando alias previamente definidos. En general la sintaxis es :

who where = (as_whom) what
<user_alias> <hosts_alias> = (runas_alias:runas_alias) [tag_spec:]<cmd_alias>
root ALL=(ALL:ALL) ALL

Las dos primeras líneas son ejemplos de la sintaxis, siendo la segunda la más específica. La tercera es un ejemplo el cual utiliza un tag especial que es "**ALL**" que dependiendo de su lugar es lo que significa. El primer ALL es principalmente para indicar la máquina (y todos los posibles hosts que utilicen dicho comando) mientras que la combinación **ALL:ALL** involucra todos los usuarios de todos los grupos y el último todo los comandos. *De esta forma lo que se esta diciendo es que Root puede ejecutar en esta y cualquier otra máquina todos los comandos como si fuese cualquier usuario o miembro de cualquier grupo*.

Además existe una especificación que se puede añadir opcionalmente a cada uno de los comandos (o alias de comandos), estas especificaciones son denominadas "**Tag_Spec**" y tiene 8 posibles valores:

- **PASSWD y NOPASSWD -** (Por defecto esta PASSWD), que decide si la ejecución del comando con SUDO solicite o no las credenciales del usuario (la contraseña).
- **NOEXEC y EXEC**
- **NOSETENV y SETENV**
- **NOLOG_OUTPUT y LOG_OUTPUT**

Por último, existe una alternativa para no trabajar en el archivo original y se trata del directorio `/etc/sudoers.d` (que esta comentando en la última línea del archivo /etc/sudoers). El objetivo de este directorio es permitir que el administrador no altere el archivo original (y sobre todo no lo borre...) y en su lugar usar uno secundario o bien utilizar uno genérico para las computadoras y luego archivos adicionales por regiones.

En caso de utilizar esta alternativa, la revisión se hace con visudo. Por ejemplo, observe el siguiente cuadro después de retirar el símbolo "#" de la última línea:

```
usuario@usuario-VirtualBox:~$ sudo visudo
[sudo] password for usuario:
visudo: >>> /etc/sudoers: syntax error near line 28 <<<
What now?
Options are:
  (e)dit sudoers file again
  e(x)it without saving changes to sudoers file
  (Q)uit and save changes to sudoers file (DANGER!)

What now? x
usuario@usuario-VirtualBox:~$
```

Lo que ha ocurrido es que se ha activado la línea, se ha salvado y el visudo marca un error porque no ha podido terminar el archivo. Esto es natural ya que visudo busca documentos en dicha carpeta y al no hallar ninguno entonces termina con error. ES muy importante que todos

los archivos en dicha carpeta **le peternezcan a root:root** y respecto a UGO tenga **el valor de 0440**. En caso de no cumplirlo Visudo pasará de ellos.

= = = = Laboratorio = = = =

- Actividades a realizar en esta práctica se encuentran descritas en este documento, sus respuestas deben registrarse en :

    - M**aterial de Apoyo**: Lab-05: SUDO
    - E**nlace al formulario:** Práctica 5: SUDO .

Bibiliografías:

Basic administration - User accounts explanation
www.linux-tutorial.info/modules.php?name=MContent&pageid=69
What is user nobody and why is it needed
http://en.wikipedia.org/wiki/Nobody_%28username%29
Running wireshark as you
https://blog.wireshark.org/2010/02/running-wireshark-as-you/
Sudoers sample file
http://www.gratisoft.us/sudo/sample.sudoers

Quick HOWTO : Ch09 : Linux Users and Sudo

http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO_:_Ch09_:_Linux_Users_and_Sudo

Chmod UGO permissions quick reference
http://www.omnisecu.com/gnu-linux/redhat-certified-engineer-rhce/how-to-use-chmod-command-to-change-linux-file-permissions.php
Setting default editor to visudo
https://www.garron.me/en/linux/visudo-command-sudoers-file-sudo-default-editor.html
Using sudoers.d folder
https://luiseth.wordpress.com/2012/04/15/in-a-nutshell-add-permissions-with-configuration-files-in-etcsudoers-d/
Basic Linux Permissions part 6: sudo and sudoers demo video

https://www.youtube.com/watch?v=YSSIm0g00m4