Informe práctica 1
Usuarios y protección en GNU/Linux

Alejandro Samarín Pérez - alu3862
José Lucas Grillo Lorenzo - alu3181

18 de mayo de 2011

Resumen

El objetivo de la práctica consiste en la utilización de los mecanismos que implementa Linux para la gestión de usuarios y grupos en un sistema de estas características. Además se pretende familiarizar a los administradores con los mecanismos de protección habilitados en este sistema operativo. Para ello, se plantea un caso práctico de una organización que utiliza un sistema Linux como soporte informático.

Se pretende crear una serie de grupos y usuarios pertenecientes a los mismos que se relacionan con los recursos del sistema mediante la instauración de diversos permisos y restricciones. La realización de la práctica se llevó a cabo de manera automatizada utilizando scripting bash.

Parte I
Creación de los usuarios

A continuación se listan los requerimientos referentes a las contraseñas:

1. *Los usuarios deben cambiar sus contraseñas cada 3 meses*
2. *Es necesario notificar a los usuarios 1 día antes de que su contraseña caduque.*
3. *Transcurridos 2 días desde la caducidad del password, la cuenta ha de quedar desactivada.*
4. *En la práctica, la contraseña de cada usuario coincidirá con el nombre de usuario*

Se ha implementado una función *"create_user"* para la creación de las cuentas de usuario y asignación de contraseñas. Para los puntos 1, 2 y 3 del primer apartado, se resolvió utilizando la siguiente invocación del comando *chage*:

```
chage –M 90 –W 1 –I 2 <usuario>
```

El parámetro "-M 90" indica el período máximo en días antes del cual los usuarios deben cambiar su contraseña (90 días). El parámetro "-W 1" significa que se avisará al usuario 1 día antes de que caduque la contraseña. Por último, el parámetro "-I 2" desactivará la cuenta del usuario si no ha cambiado aún su contraseña tras 2 días desde su caducidad (período de gracia).

El punto 4 se ha resuelto mediante el comando *chpasswd*, que permite cambiar la contraseña de forma automatizada a diferencia del comando *passwd*. Se utiliza tal que así:

```
echo <usuario>:<usuario> | chpasswd
```

Es decir, se le aplica como contraseña el propio nombre de usuario y se redirecciona al comando *chpasswd* que se encarga de establecerla al usuario especificado en el primer campo (antes del ':')

En cuanto a los requisitos de los directorios de conexión de los usuarios y sus permisos asociados:

1. *Todo usuario del sistema debe poseer un subdirectorio del directorio /home cuyo nombre debe coincidir con el de la cuenta del usuario.*

2. *En este directorio, el usuario debe poder crear y borrar ficheros y directorios, pero no debe poder modificar los permisos de su directorio de conexión.*

3. *Ningún otro usuario del sistema podrá acceder a dicho directorio ni a su contenido.*

4. *Cada usuario tendrá una cuota de disco de 50 Mb que podrá sobrepasar temporalmente (2 días).*

La creación de los usuarios se realiza invocando al comando *useradd*. Para satisfacer la condición de que cada usuario debe poseer un directorio dedicado en /home, se le añaden los siguientes paŕametros:

```
useradd -m -d /home/<usuario> -U -s /bin/bash -G <grupos> <usuario>
```

Para establecer el directorio de conexión, se utilizan conjuntamente las opciones "-m", para crearlo si no existe, y "-d" para asociarlo al usuario. La opción "-U" establece como grupo principal del usuario uno exclusivo con su mismo nombre, siguiendo las recomendaciones y estrategias de muchas distribuciones públicas de Linux (entre ellas Red Hat). A continuación, se especifica la shell que utilizará el nuevo usuario con el parámetro "-s", en este caso una bash estándar. Los grupos suplementarios vendrán establecidos por los proyectos a los que el usuario esté asignado, que habrán sido creados previamente, como se detallará...

La estrategia que se ha seguido para garantizar que el usuario tenga total libertad para crear, borrar y modificar ficheros en su propio home, pero sin poder modificar los permisos del mismo, es:

- Establecer como usuario propietario del directorio a root, ya que es solamente el usuario propietario quien puede modificar los permisos del directorio.

- Establecer como grupo propietario al grupo propio del usuario (razón para indicar la opción "-U" explicada anteriormente), para que así tenga acceso a su directorio de conexión. Esto se realiza por defecto al crear el directorio.

- El resto de usuarios no tendrá ningún permiso sobre dicho directorio.

Para llevar esto a cabo se han usado los siguiente comandos:

```
chown root /home/<usuario>
chmod 770 /home/<usuario>
```

Con el *chmod 770* aseguramos que el usuario tenga completo acceso a su directorio de conexión, a la vez que restringimos el acceso al resto de usuarios.

En cuanto al requisito referente a las cuotas de usuario, se ha utilizado una variante automatizada del comando *edquota*, denominada *setquota*. Esta nos permite establecer las cuotas para el directorio de conexión del usuario, una vez estén configuradas en el sistema. La invocación del mismo es como sigue:

```
setquota <usuario> 50000 60000 0 0 -a
setquota <usuario> -T 2 0 -a 2&>/dev/null
```

En el primer caso se establecen los soft y hard limit de la cuota a 50MB y 60MB respectivamente, sin límite de inodos (número de ficheros permitidos creados por el usuario). En la segunda línea, indicamos el tiempo de gracia de 2 días para la eventual superación del soft limit de bloques. Ambas restricciones se aplican a todos los sistemas de ficheros con cuotas para ese usuario (en este caso, sólo se trata de */home*), y además en el segundo caso se redirecciona la salida de errores a */dev/null* para eliminar mensajes de aviso relacionados con no haber superado aún el límite blando establecido en ese momento.

A propósito de las cuotas, se ha comentado brevemente que es necesario configurarlas en el sistema de ficheros destino previamente a su aplicación. Para ello, se debe modificar el fichero */etc/fstab* para indicar qué particiones del sistema implementarán las cuotas, en nuestro caso */home*. Esto se ha automatizado mediante el uso de la herramienta estándar *sed* en combinación con la siguiente expresión regular:

```
sed -r 's/^\s*[^#]\S+\s+\/home\s+\S+\s+\S+/
        &,usrjquota=aquota.user,jqfmt=vfsv0/' -i.bkp1 /etc/fstab
```

Aprovechando esta sección del script que se encarga de modificar convenientemente el */etc/fstab* (función *"set_fstab"*), se ha incluido a continuación del código anterior otra invocación a *sed* para, mediante otra expresión regular, incluir asimismo el soporte para ACL's en el mismo sistema de ficheros */home*:

```
sed -r 's/^\s*[^#]\S+\s+\/home\s+\S+\s+\S+/&,acl/' -i.bkp2 /etc/fstab
```

Después de haber modificado el */etc/fstab*, es necesario activar las cuotas para finalizar la implementación:

```
umount /home
fsck /home
mount /home
quotacheck -vnm /home
quotaon -av
```

En primer lugar, se desmonta */home;* a continuación se realiza una comprobación de seguridad del sistema de ficheros ya que se pudo comprobar que en ciertos casos el sistema de cuotas encontraba algunas inconsistencias. Tras ello, se remonta */home* y se puede proceder ahora a activar las cuotas con *quotacheck* y *quotaon*.

Parte II
Proyectos

Los requisitos relacionados con la gestión de los proyectos se detallan a continuación:

1. *Cada proyecto debe tener un directorio bajo el directorio /home/proyectos donde se almacenará la documentación asociada al mismo.*
2. *Todos los usuarios que participan en un proyecto deben tener la posibilidad de leer, modificar, crear y borrar los archivos que forman parte del proyecto.*
3. *Cuando un usuario cree un archivo en el directorio del proyecto, por defecto, éste debe poder ser leído, modificado o borrado por cualquier otro usuario del mismo proyecto.*
4. *Ningún otro usuario podrá acceder a estos directorios.*
5. *Existirá un directorio /home/proyectos/comun donde se almacenará información común a todos los proyectos de tal forma que todos los usuarios puedan añadir y modificar información a este directorio, pero sólo el propietario de cada carpeta pueda eliminarla.*

Como nota preliminar, para cada proyecto se ha definido un grupo, esto es, los grupos "aeropuerto", "cc" (centro comercial) y "parque", con la idea de que cada usuario pertenezca a los grupos correspondientes según sus competencias.

Para el primer punto, simplemente se han creado los directorios correspondientes vía *mkdir*. El punto 2 se ha controlado de manera conjunta gracias a la creación de los grupos comentados anteriormente y a la asignación de permisos totales al grupo propietario:

```
mkdir -v -p -m 2770 /home/proyectos/aeropuerto
chgrp aeropuerto /home/proyectos/aeropuerto
mkdir -v -p -m 2770 /home/proyectos/cc
chgrp cc /home/proyectos/cc
mkdir -v -p -m 2770 /home/proyectos/parque
chgrp parque /home/proyectos/parque
```

En cuanto al tercer punto, también se han empleado dos acciones conjuntas: por un lado, se ha establecido el bit SETGID en el directorio del proyecto (ya especificado anteriormente en las llamadas a *mkdir*), para que cuando un usuario cree un nuevo fichero o directorio herede el grupo del directorio raíz (es decir, el grupo "aeropuerto", "cc", "parque", etc... en lugar del propio de cada usuario); por otro lado se han empleado ACL's, de nuevo, sobre el directorio del proyecto, para imponer que los permisos que dichos ficheros o directorios adquieren por defecto en el momento de su creación sean los adecuados, permitiendo a los demás usuarios del grupo hacer lecturas y modificaciones sobre los mismos (descartando así las directrices de *umask*).

```
setfacl -d -m g:aeropuerto:rw /home/proyectos/aeropuerto
setfacl -d -m g:cc:rw /home/proyectos/cc
setfacl -d -m g:parque:rw /home/proyectos/parque
setfacl -d -m o::rw /home/proyectos/comun
```

Nótese además que, en el caso del directorio *"comun"*, esta misma propiedad se aplica a "otros", ya que todos los usuarios y ejecutivos accederán por esta vía a dicho directorio y de esta forma se asegura que todos puedan asimismo modificar a placer los nuevos ítems creados.

El cuarto punto, relativo a que ningún otro usuario puede acceder a los directorios de los proyectos, ya ha sido controlado mediante el uso de los grupos exclusivos para cada directorio y una adecuada asignación de permisos con *chmod*, en este caso restringiendo a "otros" todos los privilegios posibles, de forma que no puedan acceder, consultar ni modificar el contenido de dichos directorios.

En lo tocante al punto número 5, en el momento de la creación del directorio común se establece el bit Sticky en el mismo, precisamente para asegurar que sólo el propietario del fichero puede eliminarlo:

```
mkdir -v -p -m 1777 /home/proyectos/comun
```

Parte III

Ejecutivos

1. *Los ejecutivos asociados a un determinado proyecto podrán leer la información de ese proyecto.*
2. *Los ejecutivos que no pertenezcan a un proyecto no deben poder acceder directamente a los directorios de los proyectos.*
3. *Para que estos ejecutivos puedan controlar el estado de cada proyecto, deben existir en el directorio /usr/local/bin tantos programas como proyectos existan.*
4. *Estos programas internamente han de realizar un "ls" sobre el directorio del proyecto correspondiente.*
5. *El programa que permite evaluar cada proyecto, debe cumplir lo siguiente:*
   a) *Debe poder ser ejecutado únicamente por los ejecutivos de la organización.*
   b) *Debe tener asignado los permisos suficientes para poder ejecutar el "ls" sobre el directorio correspondiente.*

La creación de un grupo ejecutivos facilita la ejecución de los programas por parte de los ejecutivos únicamente, así como simplificar las tareas de asignación de permisos a nuevos ejecutivos que pudieran ser añadidos posteriormente.

El primer punto se resolvió mediante el uso de ACLs, otorgando exclusivamente permisos de lectura y ejecución a cada ejecutivo perteneciente a cada proyecto.

En el segundo se optó por denegar los permisos para leer, ejecutar, o manipular el contenido de los directorios de las proyectos para el resto de usuarios (que no sean propietarios ni miembros del grupo propietario), incluyendo de este modo a los ejecutivos que no tuviesen asignado un determinado proyecto.

Por otra parte, para que los ejecutivos puedan controlar el estado de cada proyecto, deben existir en el directorio /usr/local/bin tantos programas como proyectos existan. Estos programas consisten en un pequeño

código en C que realiza una llamada a la función *execl()* y que internamente realizan un *ls* sobre el directorio del proyecto especificado. Además, a los ejecutables ya compilados se les activa el bit SETUID mediante *chmod*, y se establece como usuario propietario de los mismos a root, de forma que al realizarse la suplantación de EUID se tienen suficientes permisos para efectuar el *ls* sobre los proyectos. Para que los ejecutivos puedan ejecutarlo (y sólamente ellos), el grupo propietario de dichos binarios es el grupo ejecutivos, y tiene asignados permisos de lectura y ejecución. De esta forma, los ejecutivos pueden acceder a los proyectos de manera controlada por los administradores.

Parte IV

Anexo: Script de automatización

```
#!/bin/bash

#############################################################
#                ADMINISTRACION DE SISTEMAS                 #
#      PRACTICA 1: Usuarios y proteccion en Linux           #
# ------------------------------------------------------- #
# ALUMNOS     : Alejandro Samarin Perez   (alu3862)         #
#             : Jose Lucas Grillo Lorenzo (alu3181)         #
# FECHA       : 14/03/2011                                  #
# DESCRIPCION: Este script crea varios usuarios en          #
#              el sistema teniendo en cuenta diver-         #
#              sas medidas de seguridad y tecnicas          #
#              de trabajo en grupo.                         #
#############################################################

# ---------------------------------------------------------------------------

# FUNCION     : create_user
# DESCRIPCION: Crea en el sistema un nuevo usuario
# ARGUMENTOS :
#    $1 => Nombre de usuario
#    $2 => Lista de grupos suplementarios del usuario (Ej:
#          "cc,aeropuerto,parque")
function create_user {
    echo "[+] Creando usuario '$1'"
    # Crear el usuario obtenido en el primer argumento de la funcion, con
    # su directorio de conexion establecido en /home, grupo propio (-U),
    # shell bash estandar y los grupos suplementarios especificados en el
    # segundo argumento de la funcion
    useradd -m -d /home/$1 -U -s /bin/bash -G $2 $1

    # Cambiar la password del nuevo usuario (igual que su username)
    echo $1:$1 | chpasswd

    # Cambiar los parametros de caducidad de su password, segun se indica:
    #   -M 90 => La contrasenya caduca en 3 meses
    #   -W 1  => Se informara al usuario 1 dia antes de la caducidad de
    #            la misma
    #   -I 2  => 2 dias tras la expiracion, la cuenta del usuario sera
    #            deshabilitada
    chage -M 90 -W 1 -I 2 $1
```

```
    # Root sera el propietario de su directorio de conexion, para que el
    # usuario no pueda modificar los permisos del mismo
    chown root /home/$1

    # Estableciendo permisos "rwxrwx---" se asegura que el usuario
    # puede crear, modificar y borrar ficheros y directorios en su
    # 'home', a la vez que se previene que otros usuarios puedan acceder
    chmod 770 /home/$1

    # Establecer las cuotas de usuario:
    #   Limite blando (por bloques) => 50MB
    #   Limite duro (por bloques)   => 60MB
    #   Sin limite de inodos
    setquota $1 50000 60000 0 0 -a

    # Establecer el periodo de gracia a la hora de rebasar el limite
    # blando (2 dias)
    setquota $1 -T 2 0 -a 2&>/dev/null
}

# ------------------------------------------------------------------------

# FUNCION     : set_fstab
# DESCRIPCION: Modifica el fichero /etc/fstab para activar en el sistema
#              el uso de cuotas de usuario y listas de control de acceso
#              (ACL); en caso necesario, remonta la particion /home y
# ARGUMENTOS : Ninguno
function set_fstab {
    local remount_needed=false

    # Comprobar si estaban seteadas las cuotas en la particion de /home
    if [[ -z `grep -v "^\s*#" /etc/fstab | grep usrjquota` ]]
    then
        echo "[*] Cuotas no establecidas previamente en fstab, modificando..."
        sed -r 's/^\s*[^#]\S*\s+\/home\s+\S+\s+\S+/
                &,usrjquota=aquota.user,jqfmt=vfsv0/' -i.bkp1 /etc/fstab
        remount_needed=true
    else
        echo "[+] Cuotas establecidas previamente"
    fi

    # Idem para las listas de control de acceso (ACL)
    if [[ -z `grep -v "^\s*#" /etc/fstab | grep acl` ]]
    then
        echo "[*] ACL's no establecidas previamente en fstab, modificando..."
        sed -r 's/^\s*[^#]\S*\s+\/home\s+\S+\s+\S+/&,acl/' -i.bkp2 /etc/fstab
        remount_needed=true
    else
        echo "[+] ACL's establecidas previamente"
    fi

    # Si hubo algun cambio en las comprobaciones anteriores, es necesario
    # remontar /home. Tambien se activan aqui las cuotas
```

```
    if ($remount_needed)
    then
        echo "[+] Remontando /home y comprobando cuotas..."
        umount /home
        # Una vez desmontado /home, se realiza una comprobacion del sistema
        # de ficheros por seguridad, ya que en ocasiones quotacheck puede
        # encontrar errores
        fsck /home
        mount /home
        quotacheck -vnm /home
        quotaon -av
    fi
}

# ------------------------------------------------------------------------

# FUNCION     : create_ls
# DESCRIPCION: Crea un pequenyo programa ejecutable en C que emula un
#              'ls', pensado para ser utilizado por los ejecutivos que
#              desean observar el desarrollo de proyectos a los cuales no
#              estan asignados; posteriormente se compila este codigo fuente
#              y se coloca el ejecutable en '/usr/local/bin'. El nombre del
#              ejecutable generado tiene la forma 'ls<proyecto>'
# ARGUMENTOS :
#   $1 => Nombre del proyecto (Ej: "aeropuerto" genera
#         '/usr/local/bin/lsaeropuerto')
function create_ls {
    # Crear en /tmp el codigo fuente en C
    echo "
#include <stdio.h>
#include <unistd.h>

void main() {
   execl(\"/bin/ls\", \"/bin/ls\", \"-l\", \"/home/proyectos/${1}\", NULL);
}" > /tmp/ls$1.c

    # Compilar el programa C creado y depositar el ejecutable resultante
    # en el directorio destino '/usr/local/bin'
    gcc /tmp/ls$1.c -o /usr/local/bin/ls$1

    # Cambiar UGO y grupo del nuevo ejecutable para que solamente pueda
    # ser utilizado por los ejecutivos
    chgrp ejecutivos /usr/local/bin/ls$1
    chmod -v 4770 /usr/local/bin/ls$1

    # Eliminar el codigo fuente temporal, ya que no es necesario en este
    # punto
    rm /tmp/ls$1.c
}

# ------------------------------------------------------------------------

# FUNCION     : set_pam
# DESCRIPCION: Modifica los ficheros /etc/pam.d/login y
```

```
#               /etc/security/time.conf  para activar en el sistema las
#               restricciones de acceso a usuarios mediante PAM
# ARGUMENTOS : Ninguno
function set_pam {
   # Invocar a sed para incluir el soporte para el control de tiempos
   # en los ficheros de configuracion de las PAM, si este no habia sido
   # establecido anteriormente
   # sed -r 's/(account\s+required.*)+/account     required
   #                                 pam_time.so\n&/m' -i.bkp /etc/pam.d/login
   if [[ -z `egrep "^\s*account\s+required\s+pam_time.so" /etc/pam.d/login` ]]
   then
      sed -r '0,/^\s*account.*/s/^\s*account.*/
              account required pam_time.so\n&/' -i.bkp /etc/pam.d/login
   fi

   # Crear backup del fichero /etc/security/time.conf si no existia
   # previamente
   if [[ ! -x /etc/security/time.conf.bkp ]]
   then
      cp /etc/security/time.conf /etc/security/time.conf.bkp
   fi
   # usu1 y usu2 pueden logearse por cualquier tty entre las 9:00 y las
   # 15:00 independientemente del dia
   echo "login;tty*;usu1|usu2;Al0900-1500" >> /etc/security/time.conf
   # usu3 puede logearse por cualquier tty de lunes a jueves entre las
   # 16:00 y las 21:00
   echo "login;tty*;usu3;FrWk1600-2100" >> /etc/security/time.conf
   # usu5 solo puede logearse por la tty5 los dias laborables entre las
   # 9:00 y las 15:00
   # (En dos pasos, no esta muy claro el por que)
   echo "login;!tty5;usu5;*" >> /etc/security/time.conf
   echo "login;tty*;usu5;Wk0900-1500" >> /etc/security/time.conf
}

# ------------------------------------------------------------------------
# MAIN: Punto de entrada del script
# ------------------------------------------------------------------------

# PASO 1: Crear los nuevos grupos. uno para cada proyecto y otro para los
# ejecutivos
echo "[+] Creando grupos necesarios para los proyectos y ejecutivos..."
groupadd aeropuerto
groupadd cc
groupadd parque
groupadd ejecutivos

# PASO 2: Crear los directorios necesarios de los proyectos, y asignarle
# el grupo primario correspondiente a los mismos
# Opciones de mkdir:
#   -v => Modo verbose (muestra un mensaje por pantalla por cada
#         directorio creado)
#   -p => Crea directorios padres si son necesarios (evita un 'mkdir
#         /home/proyectos')
#   -m XXXX => Establece los permisos especificados en el nuevo directorio,
```

```
#           estilo chmod
echo "[+] Creando directorios de proyectos..."
mkdir -v -p -m 2770 /home/proyectos/aeropuerto
chgrp aeropuerto /home/proyectos/aeropuerto

mkdir -v -p -m 2770 /home/proyectos/cc
chgrp cc /home/proyectos/cc

mkdir -v -p -m 2770 /home/proyectos/parque
chgrp parque /home/proyectos/parque

mkdir -v -p -m 1777 /home/proyectos/comun

# PASO 3: Llamar a la funcion 'set_fstab' para permitir cuotas y ACL's en
# el sistema
echo "[+] Comprobando fichero /etc/fstab y ajustando parametros necesarios..."
set_fstab

# PASO 4: Establecer las ACL para adecuar los accesos a cada proyecto, de
# forma que cada  nuevo fichero creado por un usuario del proyecto pueda
# ser automaticamente leido, modificado o borrado (esto ultimo mediante
# UGO) por otro usuario del mismo proyecto
echo "[+] Estableciendo ACL's de proyectos"
setfacl -d -m g:aeropuerto:rw /home/proyectos/aeropuerto
setfacl -d -m g:cc:rw /home/proyectos/cc
setfacl -d -m g:parque:rw /home/proyectos/parque
# En el caso de '/home/proyectos/comun', se desea que este misma propiedad
# la tengan todos
setfacl -d -m o::rw /home/proyectos/comun

# PASO 5: Crear los usuarios y ejecutivos del sistema
echo "[+] Creando usuarios y ejecutivos..."
create_user usu1 cc
create_user usu2 aeropuerto
create_user usu3 aeropuerto,cc
create_user usu4 aeropuerto,cc
create_user usu5 aeropuerto,cc,parque
create_user usu6 parque
# Notese que a los ejecutivos se les agrega al grupo destinado a tal fin,
# relacionado con el uso de los ejecutables 'ls<proyecto>'
create_user ejec1 ejecutivos
create_user ejec2 ejecutivos

# PASO 6: Permisos de solo lectura para los ejecutivos, de forma que no
#         puedan modificar ni borrar archivos de los proyectos a los que
#         pertenecen
echo "[+] Estableciendo ACL's de ejecutivos..."
# Ejecutivo 1: Notese que es necesario establecer ACLs tanto de acceso
#              como por defecto (flag '-d' en este ultimo caso), ya que
#              si se establecen solo las de acceso, no se heredan los
#              permisos deseados en los nuevos ficheros y directorios
#              que se creen a partir del directorio raiz del proyecto;
#              y en el caso contrario, esto es, que se establezcan solo
#              las ACLs por defecto, se aplicaran a todos los items
```

```
#               dependientes del directorio raiz pero NO se aplican al
#               propio directorio, por lo que los ejecutivos seguirian
#               sin disfrutar de las nuevas reglas para acceder al
#               directorio del proyecto correspondiente y sus items
#               asociados
setfacl -m u:ejec1:rx /home/proyectos/aeropuerto
setfacl -d -m u:ejec1:rx /home/proyectos/aeropuerto
setfacl -m u:ejec1:rx /home/proyectos/parque
setfacl -d -m u:ejec1:rx /home/proyectos/parque
# Ejecutivo 2:
setfacl -m u:ejec2:rx /home/proyectos/aeropuerto
setfacl -d -m u:ejec2:rx /home/proyectos/aeropuerto
setfacl -m u:ejec2:rx /home/proyectos/cc
setfacl -d -m u:ejec2:rx /home/proyectos/cc

# PASO 7: Crear los ficheros ejecutables 'ls<proyecto>' para los
#         ejecutivos
echo "[+] Creando programas para ejecutivos..."
create_ls aeropuerto
create_ls cc
create_ls parque

# PASO 8: [Opcional] Establecer las restricciones de acceso mediante
#         PAM login
echo "[+] Estableciendo restricciones PAM..."
set_pam
```