



Universidad Nacional
ARTURO JAURETCHE

Seguridad de la Información

TP n° 4

Salvatori Emiliano

Julio del 2020

ÍNDICE	2
--------	---

Índice

1. Introducción	3
2. Objetivos	3
3. Escenario	3
4. Enunciados	3
4.1. Ejercicios teóricos	3
4.2. Ejercicios Prácticos	4
5. Resoluciones	5
5.1. Aclaración a la Práctica	5
5.2. Ejercicios Teóricos	6
5.3. Ejercicios Prácticos	8

1. Introducción

En el siguiente informe se realizarán distintas actividades con respecto al Control de Acceso bajo GNU/Linux para la materia **Seguridad de la información Comisión nº 1**

2. Objetivos

Se familiarizará con los comandos de GNU/Linux y el funcionamiento del sistema de archivos en cuanto al **Control de Acceso**.

3. Escenario

Se utiliza un Sistema Operativo de tipo GNU/Linux, cuya información sobre el núcleo y la distribución, es la siguiente: *Linux usuario 5.6.15-arch1-1 #1 SMP PREEMPT Wed, 27 May 2020 23:42:26 +0000 x86_64 GNU/Linux*.

Los comandos a utilizar en este trabajo serán:

- **ls**: Comando para listar archivo en un directorio.
- **su**: Permite usar la consola de comandos de otro usuario sin necesidad de cerrar la sesión actual.
- **sudo**: Permite ejecutar un comando comandos como otro usuario sin necesidad de cerrar la sesión actual. Normalmente se usa para ejecutar como súper-usuario.
- **chmod**: Modifica los permisos de un archivo.
- **getfacl**: Muestra la ACL de un archivo específico.
- **setfacl**: Modifica la ACL de un archivo específico.
- **rm**: Comando utilizado para remover un archivo.
- **exit**: Cierra la sesión actual.

4. Enunciados

4.1. Ejercicios teóricos

Se tiene un usuario genérico UNAJ (que no pertenece al grupo "matias") y el siguiente directorio:

```
drwxr-xr-x 2 matias matias 4096 jul  3 21:06 pregunta1
```

1. El usuario UNAJ ¿podrá mover archivos a esta carpeta?
2. El usuario UNAJ ¿podrá acceder al directorio *pregunta1*?
3. Describa los permisos que van a brindar los siguientes comandos:
 - **chmod 777**
 - **chmod 444**
 - **chmod 641**
 - **chmod a-w**
 - **chmod u+w**
4. ¿Qué significa el permiso de ejecución en una carpeta?
5. ¿Qué comando se puede utilizar para cambiar el usuario dueño de un archivo?
¿Y el grupo? Dé ejemplos.
6. Si una carpeta con permisos *rw-rw-rw-* es copiada utilizando el comando **cp**,
¿los permisos se mantienen? Justifique.

4.2. Ejercicios Prácticos

1. Logearse como usuario Kali. Intente ejecutar el siguiente comando *./PATH/crearUsuario.sh* para crear el usuario Unaj. ¿Puede ejecutarlo?
2. Intente ejecutar el siguiente comando *sudo ./PATH/crearUsuario.sh* para crear el usuario Unaj. ¿Puede ejecutarlo?
3. Ejecute el siguiente comando *chmod u+x ./PATH/crearUsuario.sh*
Ejecute el siguiente comando *./PATH/crearUsuario.sh* para crear el usuario Unaj
¿Puede ejecutarlo?.
4. Ejecute el siguiente comando *sudo ./PATH/crearUsuario.sh* para crear el usuario Unaj. ¿Puede ejecutarlo?
5. Logearse como Unaj con el comando *su* y crear una carpeta *test* en */home/unaj*.
Utilice *exit* para ir de vuelta al usuario Kali ¿ Intente borrar la carpeta */home/unaj/test*. ¿Puede borrar la carpeta? ¿Porque?
6. Logearse como Unaj con el comando *su* e intente borrar la carpeta */home/unaj/test*.
¿Puede borrar la carpeta? ¿Porque?
7. Crear una carpeta *testacl* en */home/unaj* desde el usuario Unaj. ¿Alguna diferencia en los permisos de la carpeta *testacl* en los comandos *ls -l* y *getfacl*?
8. Utilice *exit* para ir de vuelta al usuario kali e intente crear una carpeta *carpetaacl* en *testacl*. ¿Puede realizarlo? Justifique.

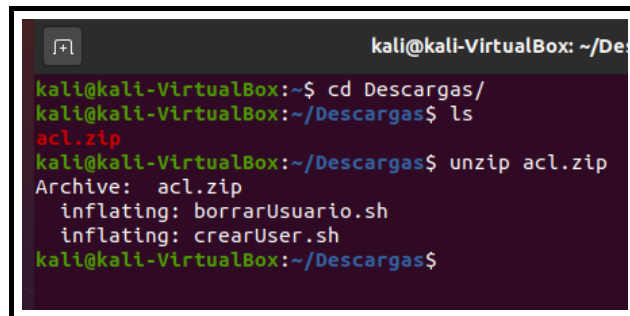
9. Ingrese nuevamente como Unaj y ejecute el comando `setfacl -m u:kali:rw testacl` (Kali es el usuario que ustedes crearon) Utilice `exit` para ir de vuelta al usuario Kali e intente crear una carpeta `carpetaacl` en `testacl`
 - ¿Puede realizarlo? Justifique
 - ¿Alguna diferencia en los permisos de la carpeta `testacl` en el comando `ls -l`?
 - ¿Alguna diferencia en los permisos de la carpeta `testacl` en el comando `getfacl`?

5. Resoluciones

5.1. Aclaración a la Práctica

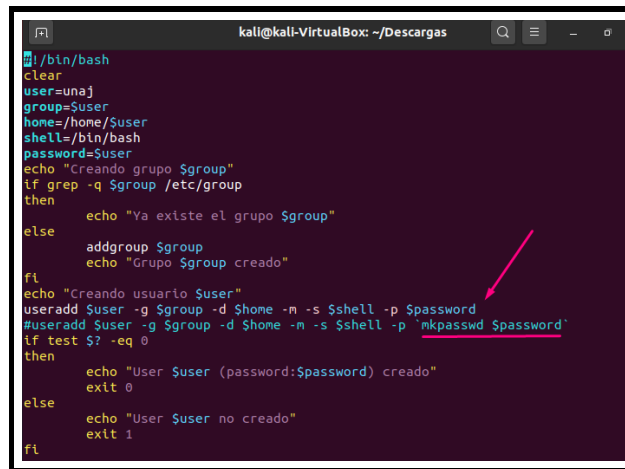
En el presente trabajo, se hizo uso de la instalación del Sistema Operativo Ubuntu, dado que la imagen de la página oficial de Kali Linux, estaba rota. La misma provocaba la interrupción de la instalación cuando requería instalar el gestor de paquetes `apt-get` bajo VirtualBox de Oracle. Para sortear este problema, se instaló el SO alternativo que se nombró al comienzo.

Se crea asimismo un usuario bajo Ubuntu con el nombre de "Kali". Luego se copian los archivos indicados en los enunciados como se muestra a continuación:



```
kali@kali-VirtualBox: ~/De
kali@kali-VirtualBox:~$ cd Descargas/
kali@kali-VirtualBox:~/Descargas$ ls
acl.zip
kali@kali-VirtualBox:~/Descargas$ unzip acl.zip
Archive:  acl.zip
  inflating: borrarUsuario.sh
  inflating: crearUser.sh
kali@kali-VirtualBox:~/Descargas$
```

También se tuvo que hacer un cambio en el script `crearUsuario.sh` dado que utilizaba un comando `mkpasswd` que no era posible interpretar de forma correcta. Simplemente se eliminó este comando y se dejó la creación de la contraseña de una forma más directa, pero menos segura, como se puede ver a continuación:



```

kali@kali-VirtualBox: ~/Descargas
#!/bin/bash
clear
user=unaj
group=Suser
home=/home/Suser
shell=/bin/bash
password=Suser
echo "Creando grupo $group"
if grep -q $group /etc/group
then
    echo "Ya existe el grupo $group"
else
    addgroup $group
    echo "Grupo $group creado"
fi
echo "Creando usuario $user"
useradd $user -g $group -d $home -m -s $shell -p $password
#useradd $user -g $group -d $home -m -s $shell -p `mkpasswd $password`
if test $? -eq 0
then
    echo "User $user (password:$password) creado"
    exit 0
else
    echo "User $user no creado"
    exit 1
fi

```

Una vez realizado estos cambios, se pudo proceder con normalidad a realizar los puntos de la práctica.

5.2. Ejercicios Teóricos

Se debe recordar que la disposición de los permisos en los archivos y directorios bajo un Sistema Operativo de tipo GNU/Linux, siguen la siguiente disposición:



1. El usuario genérico UNAJ no podrá mover los archivos a esta carpeta, porque el directorio no tiene seteados los permisos de escritura (w) ni para el grupo Group al que pueda llegar a pertenecer el usuario, ni para el grupo Others.
2. El usuario genérico UNAJ si podrá acceder a la carpeta, debido que en esta ocasión el directorio sí tiene seteados los permisos de lectura (r) para el grupo de Others.
3. Los permisos en orden correspondiente serán:

- a) **chmod 777**: Los argumentos de este comando permitirán setear los permisos de Lectura (r), Escritura (w) y Ejecución (x) a las ternas: User, Group, Others.
 - b) **chmod 444**: Los argumentos de este comando permitirá setear sólo el permiso de Lectura (r) a las ternas: User, Group, Others.
 - c) **chmod 641**: Los argumentos de este comando permitirá setear los permisos de Lectura(r) y Escritura(w) para el Usuario, el de Lectura(r) para los permisos de Grupo y el de Ejecución(x) para la terna Others.
 - d) **chmod a-w**: Los argumentos de este comando permite quitarle (-) a todos(a) los permisos de Escritura(w).
 - e) **chmod u+w**: Los argumentos de este comando permite agregarle(+) a la terna User (u) el permiso de Escritura(w).
4. El permiso de ejecución en un directorio permite que se puedan ejecutar los archivos que se encuentran dentro de esta carpeta.
5. Para cambiar el dueño de un archivo, se puede utilizar el comando **chown** el cual soporta los siguientes argumentos:

```
NAME
    chown - change file owner and group

SYNOPSIS
    chown [OPTION]... [OWNER][:[GROUP]] FILE...
    chown [OPTION]... --reference=RFILE FILE...
```

Para cambiar el grupo al que pertenece el archivo se utiliza el comando **chgrp**, el cual soporta los siguientes argumentos:

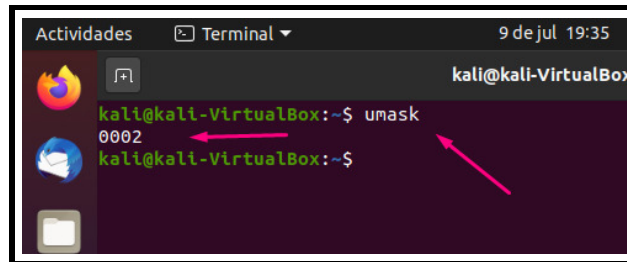
```
NAME
    chgrp - change group ownership

SYNOPSIS
    chgrp [OPTION]... GROUP FILE...
    chgrp [OPTION]... --reference=RFILE FILE...
```

6. Los permisos no se mantendrían en caso de copiarse el archivo seteado como *rw-rw-rw-*, y esto es así ya que existe una característica en los sistemas de tipo GNU/Linux que permiten definir con qué permisos iniciales se crea ese archivo. Esto se realiza en función del valor de *umask* del usuario. El valor de *umask* especifica qué permisos no se deben establecer. Por ejemplo, en el SO Ubuntu, el valor predeterminado de *umask* para un usuario normal es 002, mientras que

el valor predeterminado para el usuario root es 022. Se puede averiguar el valor actual de umask (o configurarlo) utilizando el comando `umask`.

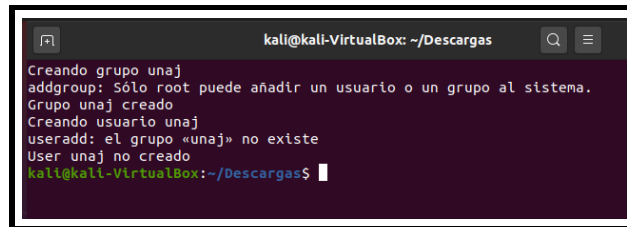
Se procede a ejecutar el comando `umask`:



```
Actividades Terminal 9 de jul 19:35
kali@kali-VirtualBox
kali@kali-VirtualBox:~$ umask
0002
kali@kali-VirtualBox:~$
```

5.3. Ejercicios Prácticos

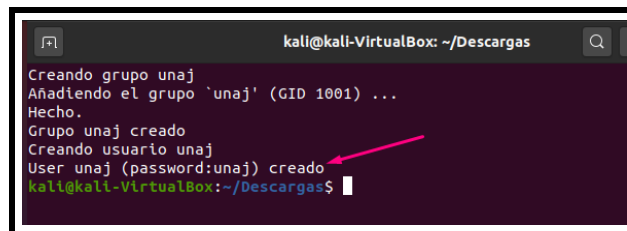
1. Se intenta ejecutar el script `crearUsuario.sh`:



```
kali@kali-VirtualBox: ~/Descargas
Creando grupo unaj
addgroup: Sólo root puede añadir un usuario o un grupo al sistema.
Grupo unaj creado
Creando usuario unaj
useradd: el grupo «unaj» no existe
User unaj no creado
kali@kali-VirtualBox:~/Descargas$
```

Como se puede visualizar, no se puede crear el usuario "unaj" dado que el usuario Kali, no tiene los permisos para realizar este tipo de operación.

2. Se procede a ejecutar el script mediante la utilidad `sudo`:



```
kali@kali-VirtualBox: ~/Descargas
Creando grupo unaj
Añadiendo el grupo 'unaj' (GID 1001) ...
Hecho.
Grupo unaj creado
Creando usuario unaj
User unaj (password:unaj) creado
kali@kali-VirtualBox:~/Descargas$
```

Como se puede observar, dado que la utilidad `sudo` permite ejecutar los programas con los privilegios de un súper usuario, la creación del nuevo grupo y el nuevo usuario fue creado de forma satisfactoria.

3. Se procede a ejecutar el comando `chmod` al script y se muestran los permisos seteados para tal archivo:


```
Creando grupo unaj
Añadiendo el grupo 'unaj' (GID 1001) ...
Hecho.
Grupo unaj creado
Creando usuario unaj
User unaj (password:unaj) creado
kali@kali-VirtualBox:~/Descargas$ chmod u+x crearUser.sh
kali@kali-VirtualBox:~/Descargas$ ls -la crearUser.sh
-rwxrwx-rw- 1 kali kali 518 jul  9 17:11 crearUser.sh
kali@kali-VirtualBox:~/Descargas$
```

Al querer ejecutar nuevamente el script, nos indica que el usuario ya existe, por lo tanto no puede crearlo nuevamente ni tampoco puede sobrescribir la información existente:

```
kali@kali-VirtualBox: ~/Descargas
Creando grupo unaj
Ya existe el grupo unaj
Creando usuario unaj
useradd: el usuario «unaj» ya existe
User unaj no creado
kali@kali-VirtualBox:~/Descargas$
```

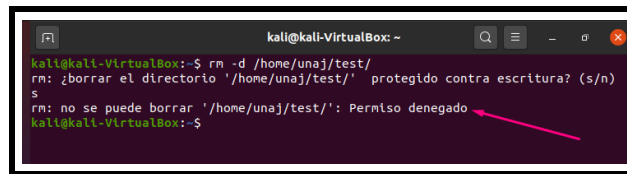
4. Se ingresa con el usuario Unaj y se crea dentro del directorio otorgado por el sistema en el directorio `/home/unaj` la carpeta denominada como `test`:

```
unaj@kali-VirtualBox: ~
unaj@kali-VirtualBox:~$ pwd
/home/unaj
unaj@kali-VirtualBox:~$ mkdir test
unaj@kali-VirtualBox:~$ ls
test
unaj@kali-VirtualBox:~$
```

Volvemos a la sesión con el usuario Kali:

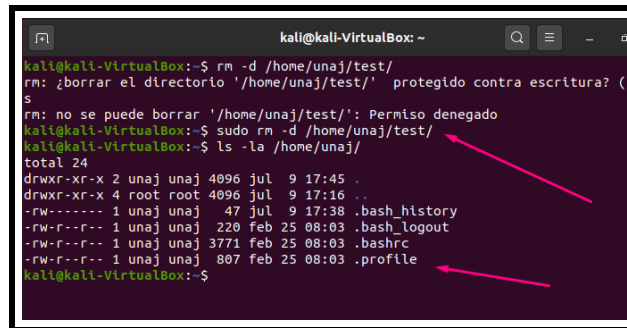
```
kali@kali-VirtualBox: ~/Descargas
root@kali-VirtualBox:~# cerrar sesión
kali@kali-VirtualBox:~/Descargas$
```

Se procede a tratar de borrar la carpeta `/home/unaj/test` pero esto no es posible dado que el usuario Kali, no tiene los permisos necesarios para realizar tal acción. La carpeta sólo tiene los permisos establecidos para que de forma exclusiva, tanto el usuario Unaj como el Root, puedan hacer modificaciones:



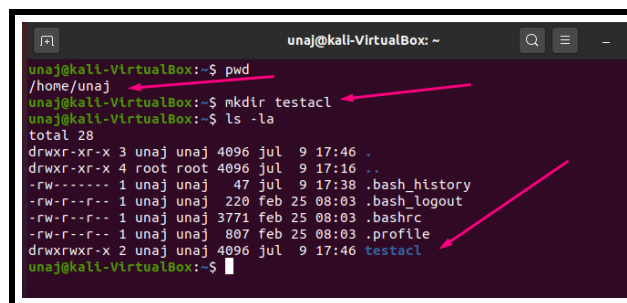
```
kali@kali-VirtualBox: ~  
kali@kali-VirtualBox:~$ rm -d /home/unaj/test/  
rm: ¿borrar el directorio '/home/unaj/test/' protegido contra escritura? (s/n)  
s  
rm: no se puede borrar '/home/unaj/test/': Permiso denegado  
kali@kali-VirtualBox:~$
```

5. Se procede a utilizar nuevamente la utilizad *sudo* para poder proporcionarnos los permisos necesarios para borrar tal carpeta que no pertenece al espacio de usuario de Kali. Cabe aclarar que cuando se invoca a la aplicación sudo, la misma genera temporalmente el cambio de permisos a súper usuario para ejecutar el comando que se le administra. Ahora si es posible poder eliminar la carpeta:



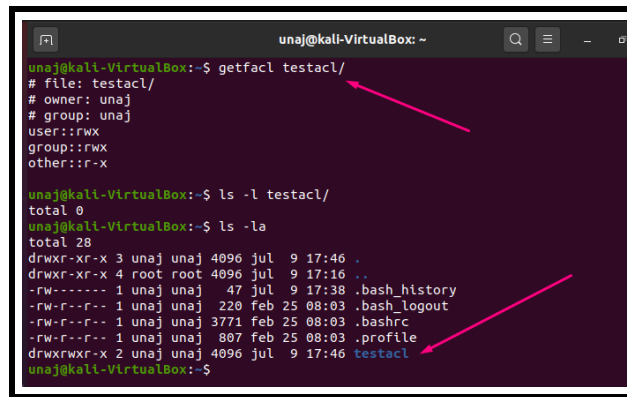
```
kali@kali-VirtualBox: ~  
kali@kali-VirtualBox:~$ rm -d /home/unaj/test/  
rm: ¿borrar el directorio '/home/unaj/test/' protegido contra escritura? (s/n)  
s  
rm: no se puede borrar '/home/unaj/test/': Permiso denegado  
kali@kali-VirtualBox:~$ sudo rm -d /home/unaj/test/  
kali@kali-VirtualBox:~$ ls -la /home/unaj/  
total 24  
drwxr-xr-x 2 unaj unaj 4096 jul  9 17:45 .  
drwxr-xr-x 4 root root 4096 jul  9 17:16 ..  
-rw----- 1 unaj unaj  47 jul  9 17:38 .bash_history  
-rw-r--r-- 1 unaj unaj 220 feb 25 08:03 .bash_logout  
-rw-r--r-- 1 unaj unaj 3771 feb 25 08:03 .bashrc  
-rw-r--r-- 1 unaj unaj 807 feb 25 08:03 .profile  
kali@kali-VirtualBox:~$
```

6. Se crea la carpeta *testacl* con el usuario Unaj:



```
unaj@kali-VirtualBox: ~  
unaj@kali-VirtualBox:~$ pwd  
/home/unaj  
unaj@kali-VirtualBox:~$ mkdir testacl  
unaj@kali-VirtualBox:~$ ls -la  
total 28  
drwxr-xr-x 3 unaj unaj 4096 jul  9 17:46 .  
drwxr-xr-x 4 root root 4096 jul  9 17:16 ..  
-rw----- 1 unaj unaj  47 jul  9 17:38 .bash_history  
-rw-r--r-- 1 unaj unaj 220 feb 25 08:03 .bash_logout  
-rw-r--r-- 1 unaj unaj 3771 feb 25 08:03 .bashrc  
-rw-r--r-- 1 unaj unaj 807 feb 25 08:03 .profile  
drwxrwxr-x 2 unaj unaj 4096 jul  9 17:46 testacl  
unaj@kali-VirtualBox:~$
```

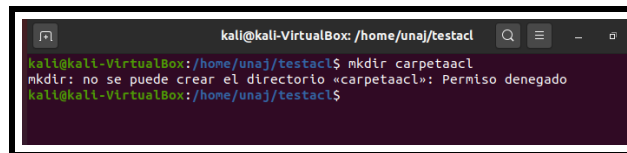
Se procede a visualizar lo expuesto por los comandos *getfacl* y *ls -l*:



```
unaj@kali-VirtualBox: ~  
unaj@kali-VirtualBox:~$ getfacl testacl/  
# file: testacl/  
# owner: unaj  
# group: unaj  
user::rwx  
group::rwx  
other::r-x  
  
unaj@kali-VirtualBox:~$ ls -l testacl/  
total 0  
unaj@kali-VirtualBox:~$ ls -la  
total 28  
drwxr-xr-x 3 unaj unaj 4096 jul  9 17:46 .  
drwxr-xr-x 4 root root 4096 jul  9 17:16 ..  
-rw----- 1 unaj unaj  47 jul  9 17:38 .bash_history  
-rw-r--r-- 1 unaj unaj 220 feb 25 08:03 .bash_logout  
-rw-r--r-- 1 unaj unaj 3771 feb 25 08:03 .bashrc  
-rw-r--r-- 1 unaj unaj 807 feb 25 08:03 .profile  
drwxrwxr-x 2 unaj unaj 4096 jul  9 17:46 testacl  
unaj@kali-VirtualBox:~$
```

Se puede notar que la información es la misma, pero la utilidad *getfacl* visualiza la información de una forma más elegante para el usuario, a diferencia de como lo hace *ls -l*:

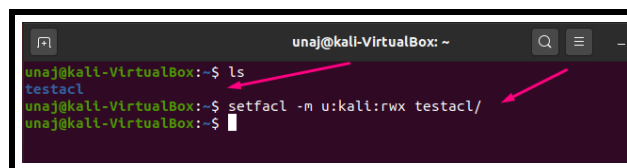
7. Volvemos al usuario Kali y creamos la carpeta *carpetaacl*:



```
kali@kali-VirtualBox: /home/unaj/testacl  
kali@kali-VirtualBox:/home/unaj/testacl$ mkdir carpetaacl  
mkdir: no se puede crear el directorio «carpetaacl»: Permiso denegado  
kali@kali-VirtualBox:/home/unaj/testacl$
```

Como se puede ver en la imagen anterior, no es posible crear la carpeta dado que no es el espacio de usuario correcto para el usuario Kali, sino que ese espacio determinado a partir de */home/unaj/testacl* es propiedad y accesible sólo por Unaj y el súper usuario Root.

8. Se procede a invocar al comando *setfacl* para otorgarle al usuario Kali, los permisos de Lectura (r), Escritura (w) y Ejecución (x) al directorio *testacl*:



```
unaj@kali-VirtualBox: ~  
unaj@kali-VirtualBox:~$ ls  
testacl  
unaj@kali-VirtualBox:~$ setfacl -m u:kali:rwx testacl/  
unaj@kali-VirtualBox:~$
```

Se procede a visualizar mediante *getfacl* y *ls -l* los permisos para la carpeta:

```

una@kali-VirtualBox:~$ ls
testacl
una@kali-VirtualBox:~$ setfacl -m u:kali:rwx testacl/
una@kali-VirtualBox:~$ ls -la
total 28
drwxr-xr-x  3 una una 4096 jul  9 17:46 .
drwxr-xr-x  4 root root 4096 jul  9 17:16 ..
-rw-r----- 1 una una 153 jul  9 17:48 .bash_history
-rw-r--r--  1 una una 220 feb 25 08:03 .bash_logout
-rw-r--r--  1 una una 3771 feb 25 08:03 .bashrc
-rw-r--r--  1 una una 807 feb 25 08:03 .profile
drwxrwxr-x+ 2 una una 4096 jul  9 17:46 testacl
una@kali-VirtualBox:~$ getfacl testacl/
# file: testacl/
# owner: una
# group: una
user::rwx
user:kali:rwx
group::rwx
mask::rwx
other::r-x
una@kali-VirtualBox:~$

```

9. Se procede a crear la carpeta *carpetaacl* bajo el directorio */home/una/testacl*. Esto es posible gracias al otorgamiento de permisos realizado en el punto anterior:

```

kali@kali-VirtualBox: /home/una/testacl
kali@kali-VirtualBox:~$ cd /home/una/testacl/
kali@kali-VirtualBox:/home/una/testacl$ pwd
/home/una/testacl
kali@kali-VirtualBox:/home/una/testacl$ mkdir carpetaacl
kali@kali-VirtualBox:/home/una/testacl$ ls -la
total 12
drwxrwxr-x+ 3 una una 4096 jul  9 17:55 .
drwxr-xr-x  3 una una 4096 jul  9 17:46 ..
drwxrwxr-x  2 kali kali 4096 jul  9 17:55 carpetaacl
kali@kali-VirtualBox:/home/una/testacl$

```

10. Se procede a visualizar mediante *getfacl* y *ls -l* los permisos para la carpeta creada en el punto anterior:

```

kali@kali-VirtualBox: /home/una
kali@kali-VirtualBox:/home/una$ getfacl testacl/
# file: testacl/
# owner: una
# group: una
user::rwx
user:kali:rwx
group::rwx
mask::rwx
other::r-x

kali@kali-VirtualBox:/home/una$ ls -la
total 28
drwxr-xr-x  3 una una 4096 jul  9 17:46 .
drwxr-xr-x  4 root root 4096 jul  9 17:16 ..
-rw-r----- 1 una una 224 jul  9 17:53 .bash_history
-rw-r--r--  1 una una 220 feb 25 08:03 .bash_logout
-rw-r--r--  1 una una 3771 feb 25 08:03 .bashrc
-rw-r--r--  1 una una 807 feb 25 08:03 .profile
drwxrwxr-x+ 3 una una 4096 jul  9 17:55 testacl
kali@kali-VirtualBox:/home/una$

```

Como se dijo anteriormente, se puede notar que la información es la misma, pero la utilidad *getfacl* visualiza la información de una forma más elegante para el usuario, a diferencia de como lo hace *ls -l*.