***Imagen que contiene Logotipo

Descripción generada automáticamente***

Alejandro Ortega Martínez

Grado en Ingeniería Informática

21/2/2021

Practica 1

Control de Accesos en Unix

Diseño y Administración de Sistemas y Redes

**1. Consultar los manuales de passwd**

**a) Averiguar las restricciones que se imponen al formato del password**

Si nos vamos al man del comando passwd, podremos leer como se realiza el cambio de contraseña para los usuarios de Unix. En el apartado donde nos dice como debemos introducir nuestra nueva contraseña, nos dice que, como guía general, una contraseña debe ser consistir de 6 a 8 caracteres, y deben incluir al menos uno o más de los siguientes conjuntos:

* Letras minúsculas
* Dígitos del 0 al 9
* Signos de puntuación

También se recalca que la contraseña debe ser lo suficientemente segura, y se recomienda que no se elija una contraseña que pueda estar en algún diccionario, nombres propios, fechas de nacimiento, direcciones, etc.

**b) Resumir los plazos y medidas de envejecimiento aplicables a los passwords en el sistema linux**

En cuanto a expiración de contraseñas con el comando passwd, tenemos las siguientes opciones:

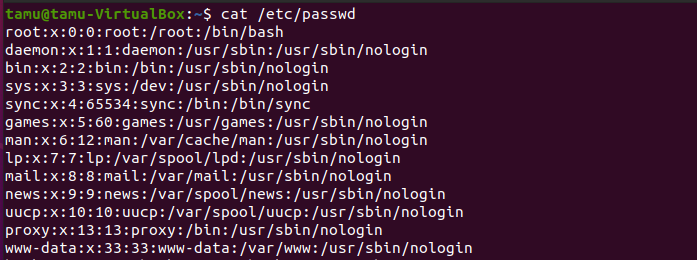
* -e: Expira inmediatamente la contraseña de un usuario. Se puede usar para forzar a un usuario a cambiar la contraseña.
* -i: Se puede usar para deshabilitar una cuenta si su respectiva contraseña ha estado expirada durante un número determinado de días.
* -n: Determina el número mínimo de días que han de pasar entre cambios de contraseña. Si lo ponemos a 0, el usuario podrá cambiar su contraseña en cualquier momento.
* -w: Determina el número de días que avisa el sistema antes de que se requiera un cambio de contraseña.
* -x: Determina el número máximo de días que una contraseña se mantiene valida, después de este numero de días, la contraseña tendrá que ser cambiada. Si especificamos -1 días, se eliminará la comprobación de la validez de la contraseña

**2. Se puede encontrar una descripción de las principales tareas de administración. Entre ellas se pueden encontrar algunas medidas relacionadas con la seguridad del sistema. Resumir las fundamentales.**

Dentro de las principales tareas de la administración, podemos encontrar:

* Definición de políticas: especifican qué queremos conseguir y qué vamos a permitir.
* Escalabilidad: un sistema escalable es el que crece de acuerdo con la política y continúa funcionando de modo previsible incluso cuando aumenta de tamaño.
* Previsibilidad: Es el relacionado con la seguridad. Los sistemas deben ser previsibles, ya que resulta la base para que sean fiables y nos den confianza y seguridad.

**3. Consultar el fichero /etc/passwd y comprobar los usuarios especiales existentes y los números de usuario utilizados**

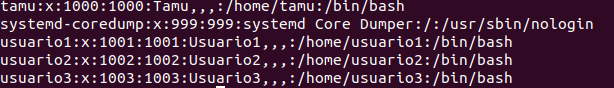
****

Como bien dice la teoría, los usuarios especiales tienen números comprendidos entre el 0 y el 9. En la captura podemos observar los siguientes:

* root (UID: 0) Usuario administrador del sistema
* daemon (UID: 1) Usuario que se encarga de la gestión de demonios
* bin (UID: 2) Usuario que se encarga de la ejecución de algunos binarios
* sys (UID: 3) Usuario que se encarga de tareas del sistema
* sync (UID: 4) Usuario que se encarga de tareas de sincronización

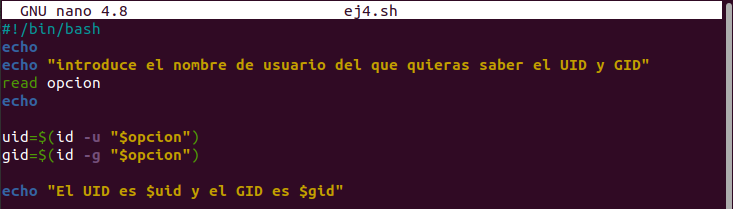
El resto son ya usuarios de diferentes aplicaciones o procesos, como mail (UID 8) o proxy (UID: 13).

Si miramos mas abajo dentro del fichero de passwd, podremos observar los usuarios “reales”, con su UID

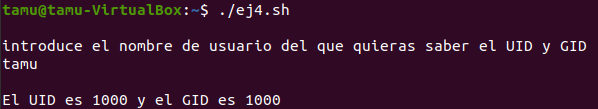


* tamu (UID 1000)
* usuario1 (UID 1001)
* usuario2 (UID 1002)
* usuario3 (UID 1003)

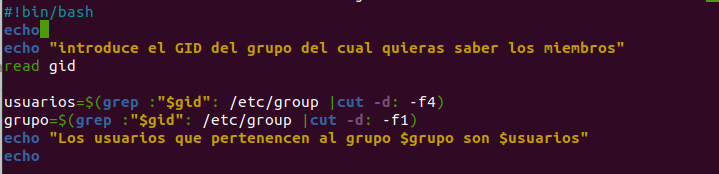
**4) Crear un guion del shell que dado un nombre de usuario (login) obtenga su UID y GID**

****

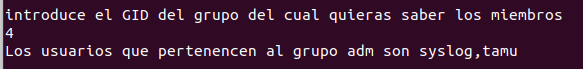
Con este pequeño script, dado un nombre de usuario, sacaremos por pantalla su UID y su GID:



**5) Crear un guion del shell que dado un GID liste los login de todos los usuarios que lo componen**

****

Con este pequeño script, dado un GID, sacaremos por pantalla su nombre y los usuarios que lo componen:



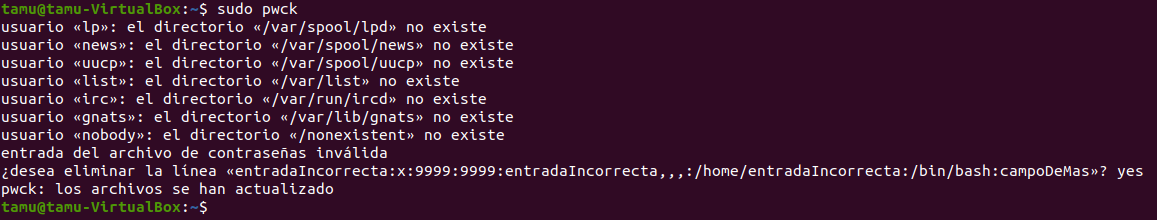
**6) Consultar el manual de la orden pwck y practicar su uso**

Este comando se encarga de comprobar la integridad de los usuarios y la información de autentificación. Comprueba todas las entradas en el archivo passwd y en el de shadow, mirando a ver si tienen el formato correcto y contienen datos inválidos. Si encuentra alguna entrada incorrecta, se le da la opción al usuario de eliminarla.

Para probarlo, vamos a introducir una entrada en el fichero passwd que tenga un formato incorrecto (por ejemplo, con un campo de más)



Al ejecutar el comando pwck, podemos observar como ha detectado la entrada incorrecta y nos ha dejado eliminarla

****

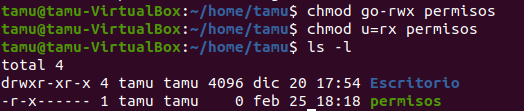
**7)Practicar el uso de la orden chmod en modo simbólico realizando los siguientes ejercicios:**

**a) Crear un fichero con permisos r-x------**

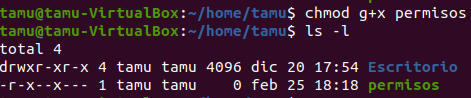
Primero creamos el archivo “permisos”



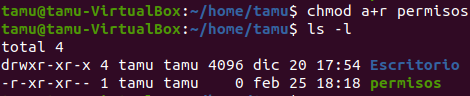
Ahora, usaremos la orden chmod con los parámetros “*go -rwx”*, y “*u=rx”*, y comprobaremos como ha cambiado el archivo



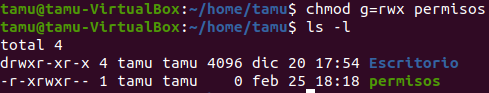
**b) Dar permiso de ejecución a los usuarios del grupo**



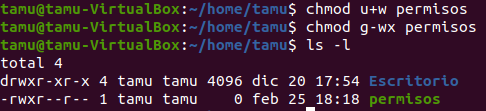
**c) Dar permiso de lectura a todos los usuarios**



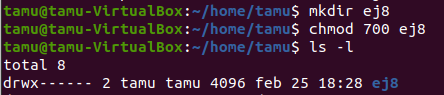
**d) Modificar los permisos de grupo para que sean rwx**



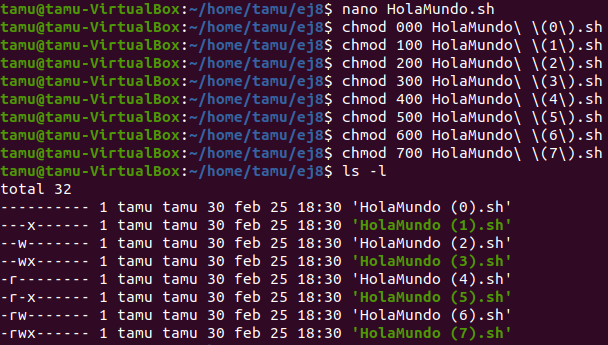
**e) Hacer que los permisos del fichero sean rwxr--r—**



**8) Crear un directorio con todos los permisos para el propietario**

****

**a) Situarse dentro y crear un fichero cuya ejecución diga "Hola Mundo" o cualquier frase de tu interés. Hacer ocho copias de ese fichero y darle a cada una de las 8 combinaciones de permisos para el propietario (0 a 7).**



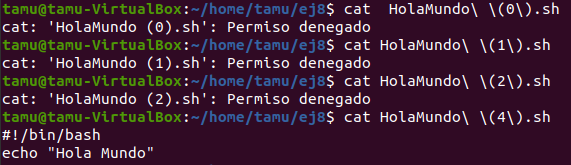
**b) Intentar deducir el funcionamiento de las siguientes ordenes con cada uno de los ficheros: cat, cp, vi(lectura), vi(escritura), rm, ejecución.**

Las ordenes de lectura (cat y el vi de lectura) solo se podrán ejecutar en los archivos con permisos de lectura.

Las ordenes de escritura (cp y vi de escritura) solo se podrán ejecutar sobre archivos que tengan tanto permisos de lectura como de escritura; y borrado (rm) solo se podrán ejecutar en los archivos con permisos de escritura.

Las ordenes de ejecución solo se podrán ejecutar sobre los archivos con permisos de ejecución.

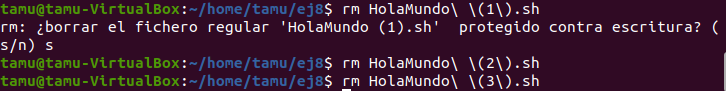
**c) Comprobar el funcionamiento de cada una de las ordenes anteriores.**

****

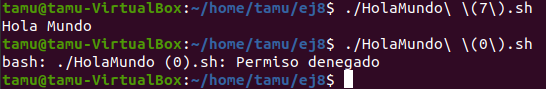
Como podemos observar, la orden cat solo la hemos podido ejecutar en el archivo que tenia permisos de lectura (400). Lo mismo nos pasaría con una orden de lectura de vi.



La orden cp (y por extrapolación, cualquier orden de escritura) no se ha podido ejecutar sobre un archivo solo con permisos de escritura (200); sino que se necesitaba también permisos de lectura (600)

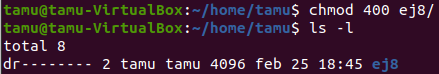


La orden rm nos deja eliminar cualquier fichero, pero si estuviera protegido contra escritura, nos pedirá aprobación.



A la hora de ejecutar, solo nos ha dejado ejecutar el que tenía permisos de ejecución (700)

**9) Modifica los permisos del directorio creado en el ejercicio anterior eligiendo una de las 8 combinaciones de permisos posibles.**

****

**a) Intentar deducir el funcionamiento de cada una de las siguientes ordenes en base al significado de los permisos sobre directorios:**

* Ordenes de listado (ls, Ll): Ambas se pueden ejecutar sin problema, ya que para ver los objetos que hay en el interior de un fichero, solo necesitamos permisos de lectura.
* Orden de acceso (cd): No podemos ejecutarla y entrar en el directorio, porque no tenemos permisos de ejecución.
* Orden de modificación del directorio (cp hacia el directorio): Tenemos el permiso denegado también.
* Orden de borrado (rm): No tenemos permisos para borrar
* Otras ordenes (cp desde el directorio): Al no poder entrar, no podemos ejecutarla.
* Lectura de un fichero del directorio (cat): Igual que el anterior
* ¿Qué ordenes dependerán de los permisos de los ficheros afectados, además de los permisos del directorio que los contiene?: Para borrar ficheros, es necesario tener permisos en la carpeta donde está contenido dicho fichero.

**b) Comprobar el funcionamiento de las ordenes anteriores y resumirlo en una tabla.**

|  |  |
| --- | --- |
| **Orden** | **Funcionamiento** |
| ls | Muestra la información de los ficheros del directorio donde estamos. Si queremos un formato mas largo y completo, usaremos la opción -l |
| ll | Similar a ls, pero utiliza un formato mas largo por defecto, como hace ls -l. Muestra, en este orden: tipo de fichero, permisos, enlaces a este, nombre del propietario, grupo del fichero, tamaño, ultima modificación y nombre del fichero |
| cd | Navega a un directorio |
| cp | Copia de origen a destino |
| rm | Elimina el fichero seleccionado |
| cat | Concatena ficheros, o la entrada estándar en la salida estandar |

**10) ¿Como puedes conocer el método de creación por defecto de un fichero?**

Gracias a la máscara, que conoceremos con el comando umask. Se hace un AND del complemento a1 de esta mascara con los permisos de creación por defecto de los archivos, y el resultante es el valor de los permisos con los que se crean los ficheros.



**11)Si el modo de creación es 0666, ¿Qué efecto tienen las mascaras 0077, 0022 y 0002? ¿Y si el modo de creación es 0644?**

* Con el modo de creación 0666
  + Mascara 0077

0666 🡪 000 110 110 110

0077🡪 000 000 111 111

(0077)🡪111 111 000 000

0600 🡪000 110 000 000 Tendria efecto 600

* + Mascara 0022

0666 🡪 000 110 110 110

0022 🡪 000 000 010 010

(0022)🡪111 111 101 101

0644 🡪000 110 100 100 Tendría un efecto 644

Mascara 0002

0666 🡪 000 110 110 110

0002 🡪 000 000 000 010

(0002) 🡪 111 111 111 101

0664 🡪 000 110 110 100 Tendría un efecto 664

* Con el modo de creación 0644
  + Mascara 0077

0644 🡪 000 110 100 100

0077🡪 000 000 111 111

(0077)🡪111 111 000 000

0600🡪 000 110 000 000 Tendría un efecto 600

* + Mascara 0022

0644 🡪 000 110 100 100

0022 🡪 000 000 010 010

(0022)🡪111 111 101 101

0644 🡪 000 110 100 100 Tendría un efecto 644

Mascara 0002

0644 🡪 000 110 100 100

0002 🡪 000 000 000 010

(0002)🡪 111 111 111 101

0644 🡪 000 110 100 100 Tendría un efecto 644

**12)¿Cuál sería la máscara que crearía los ficheros sin permiso de escritura ni ejecución para los usuarios no propietarios de los mismos?**

Una máscara que podríamos usar es 0433.

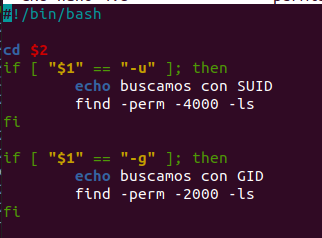
Modo de creación 0??? 🡪 000 ??? ??? ???

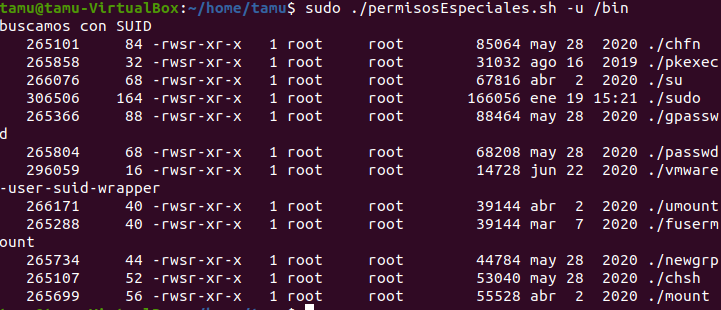
Mascara 0433 🡪 000 100 011 011

Complemento a1 🡪 111 011 100 100

Permiso 🡪 000 0?? ?00 ?00

**13)(Optativo a valorar) Escribir un guión con dos parámetros. Si el primer parámetro es -u se listarán los ficheros SUID y si es -g si listarán los SGID. El segundo parámetro será el nombre de un directorio en el que se buscarán ficheros SUID o SGID en función del primer parámetro.**

****

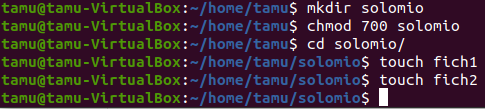
****

**14) Ejercicio de manejo de programas SUID**

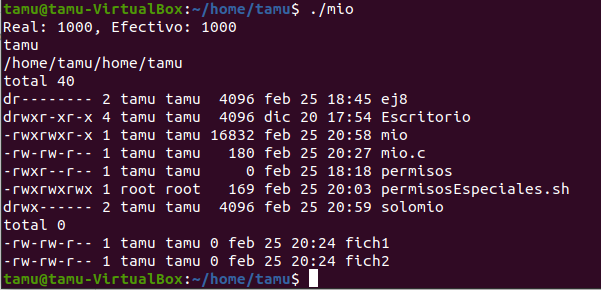
**a) Estudia el objetivo de las llamadas al sistema getuid() y geteuid() y system()**

* Getuid(): El comando getuid() devuelve el UID del usuario real del proceso que lo llama.
* Geteuid(): El comando geteuid() devuelve el UID del usuario efectivo del proceso que lo llama.
* System(): Ejecuta un comando de la shell

**b) Crea un directorio llamado solomio con permisos 700. Dentro del mismo crea los ficheros fich1 y fich2.**

****

**c) Copia el siguiente código. Denomina al fichero mio.c. Compílalo mediante la orden make para obtener el ejecutable mio. Ejecútalo e interpreta el resultado.**

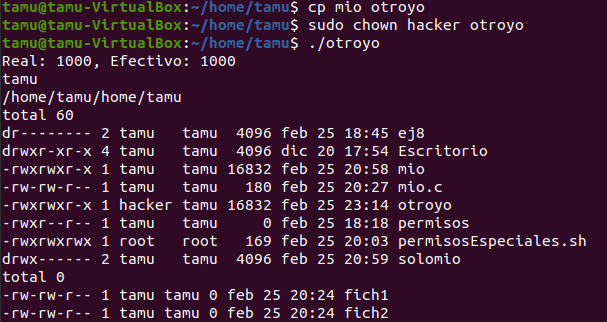
****

Lo primero que hace este script es mostrarnos el UID tanto del usuario real como del usuario efectivo que esta haciendo la llamada. En este caso el UID es 1000, que se corresponde a mi usuario.

A continuación imprime por pantalla el nombre de usuario, y el directorio donde estoy.

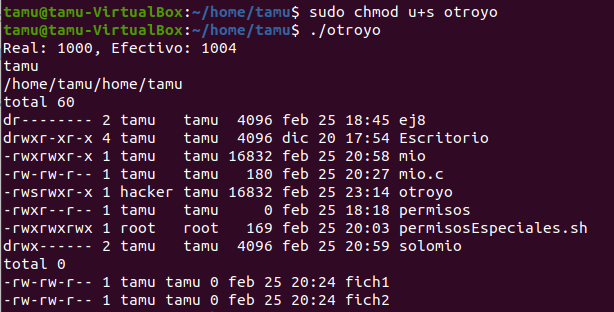
Por último, imprime los documentos que hay en el directorio desde el que se ha llamado al script, y los documentos que hay en el directorio “solomio”.

**d) Copia en tu directorio el fichero del apartado b como fichero otroyo propiedad del usuario hacker. Ejecútalo e interpreta los resultados**

****

La salida es la misma, ya que quien el UID de la persona que lo ejecuta, tanto real como efectivo, es el mismo de antes (tamu🡪UID1000)

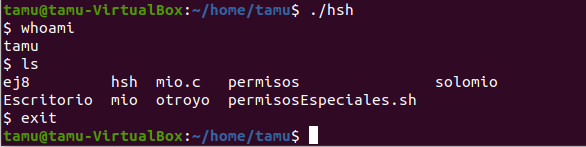
**e) También en tu directorio, copia el ejecutable suid, propiedad de hacker y SUID desde su ubicación original. Ejecútalo e interpreta los resultados.**

****

Podemos observar que ahora el UID real, que es el nuestro (tamu🡪1000) difiere del efectivo (hacker🡪1004)

**15)Copia en tu directorio el fichero el fichero /bin/sh (del interprete de ordenes) como hsh y como propiedad de hacker y SUID. Ejecútalo y comprueba que ocurre con las ordenes, whoami, ls, ls ../hacker y exit. Interpreta el resultado.**

****

****

**16) Estudiar el manual de la orden write. Estudiar lo que ocurre cuando las líneas escritas comienzan con el carácter !. ¿Supone esto algún peligro para el sistema?. Estudiar los permisos del programa /bin/write**

El comando write sirve para comunicarse con otros usuarios del sistema, mandándoles mensajes.

Si el mensaje empieza por !, lo que venga despues se ejecutara en la shell, siendo esto un grave problema de seguridad.

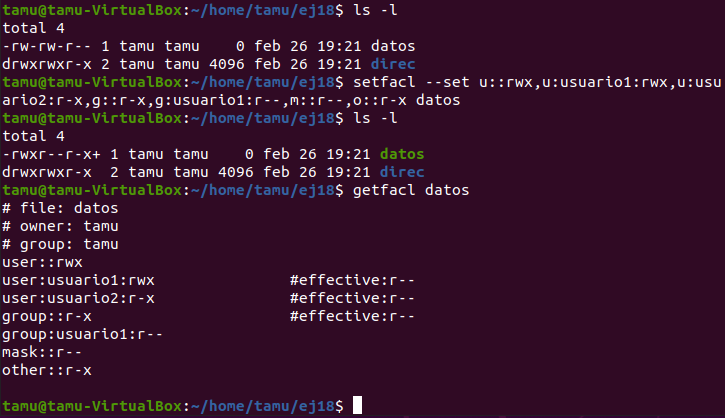


Como podemos observar, tiene activado el SGID.

**17)Revisar las ayudas de las dos órdenes getfacl y setfacl e identificar las diferentes opciones en los ejemplos de modificación de permisos mediante setfacl: setfacl –b –k –R \*** **setfacl –R –m g:sistemas:rw**

* getfacl: Obtiene las listas ACL de control de acceso del fichero.
* setfacl: Actualiza las listas ACL de control de acceso del fichero.
* Setfacl -b -k -R \*: Con el parámetro “-b”, eliminamos todas las entradas opcionales, pero dejamos las entradas base. Con el parámetro “-k” eliminamos las entradas ACL por defecto. Con el parámetro “-R”, hacemos esta operación de manera recursiva, y el \* quiere decir que o hacemos a todos los archivos del directorio. En resumen, eliminamos TODAS las entradas de las ACL de todos los archivos de nuestro directorio, y recursivamente de todos los archivos descendientes.
* Setfacl -R -m g:sistemas:rw: Con el parámetro “-R” hacemos que la operación se realice de manera recursiva. El -m indica que a continuación viene una especificación de ACL, e indicamos que queremos que los usuarios del grupo sistemas tengan permisos de lectura y escritura.

**18)Escribir los ACLs del resto de ficheros y directorios listados en el ejemplo anterior y verificar los valores efectivos debidos a mask.**

****

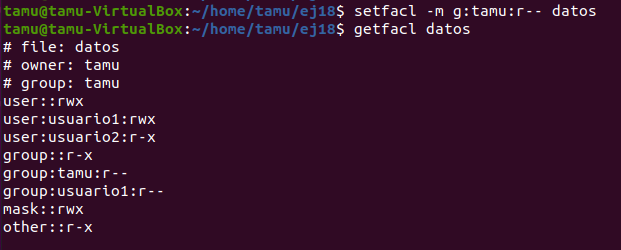
Podemos observar que hemos creado un directorio que contiene el fichero datos y el directorio direc.

Con la orden setfacl hemos hecho lo siguiente:

* Dar permisos de lectura, escritura y ejecución al usuario dueño del fichero.
* Dar permisos de lectura, escritura y ejecución al usuario1
* Dar permisos de lectura y ejecución al usuario2
* Dar permisos de lectura y ejecución a los usuarios del mismo grupo que el dueño
* Dar permisos de lectura a los usuarios que estén en el grupo “usuario1”
* Poner una máscara r—
* Dar permisos de lectura y ejecución al resto de usuarios

Podemos observar que, a diferencia de los apuntes, el hecho de que un fichero tiene valores ACL añadidos se muestra con un + al final de los permisos, y no con un \* al final del nombre.

**19)Escribir la orden para conceder permisos de escritura a un usuario de tu grupo sobre el fichero datos.**

****

Con este comando, todos los usuarios de mi propio grupo (tamu) tienen permisos de lectura. Este grupo coincide con el grupo del dueño del archivo, por lo que los permisos se solaparían.

**20)¿ Si el usuario pedro perteneciese al grupo bin, podría ejecutar el fichero del ejemplo anterior? Si la ACL del fichero fuese la siguiente**

# file: datos

# owner: pedro

# group: inf

user::rw-

user:pedro:rw-

group::r-x

other::---

Por la línea de “other”, pedro no podría ejecutar el fichero. Sin embargo, sobre esta prima la línea “user::rw-”, que nos diría que el usuario dueño del archivo (pedro) no puede ejecutarlo. Aún así, por encima en la jerarquía esta la línea “user:pedro:rw-”, que nos dice que no puede ejecutarlo. En resumen, pedro no puede ejecutar el fichero.

**a) ¿Podría el usuario pedro en el grupo inf ejecutar el fichero?**

A primera vista podría parecer que si, ya que tenemos la línea “group::r-x”, indicándonos que todos los usuarios del grupo dueño (inf), pueden ejecutar el fichero. Sin embargo, volvemos a lo mismo; pedro tiene una línea más restrictiva (“user:pedro:rw-”), que nos dice que no tiene permisos de ejecución.

**b) ¿Podría el usuario pedro en el grupo bin ejecutar el fichero?**

La respuesta es la misma que en el primer apartado de esta pregunta 20

**c) ¿Qué permisos tendría el usuario pedro en el grupo inf?**

Por la línea de permisos “user:pedro:rw-”, que es la más restrictiva, tendríamos que pedro tiene permisos de lectura y escritura, pero no de ejecución.

**d) ¿Qué permisos tendría el usuario juan en el grupo bin?**

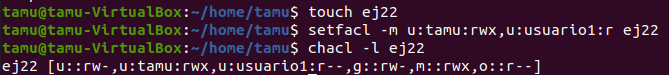
A juan no le afectan las líneas del grupo dueño ni del usuario dueño. Además, no hay líneas especificas para el usuario juan, ni para el grupo bin, por lo que caería en el apartado “other”, donde vemos que no tiene ningún tipo de permisos.

**21)Probar la orden chacl en distintos directorios del sistema. Utiliza la opción -l para comparar los dos formatos de salida.**

****

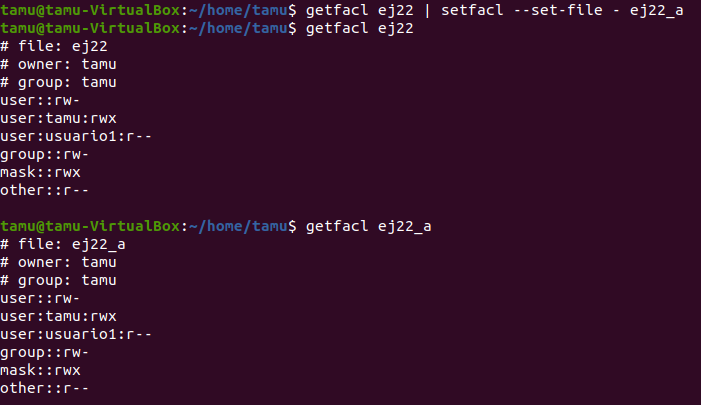
Podemos comprobar que la orden chacl tiene la misma sintaxis que setfacl, y si usamos la opción -l nos imprime los permisos de un fichero

**22)Crea un fichero cualquiera y concede los permisos que quieras a un usuario de tu grupo y a uno de un grupo distinto.**

****

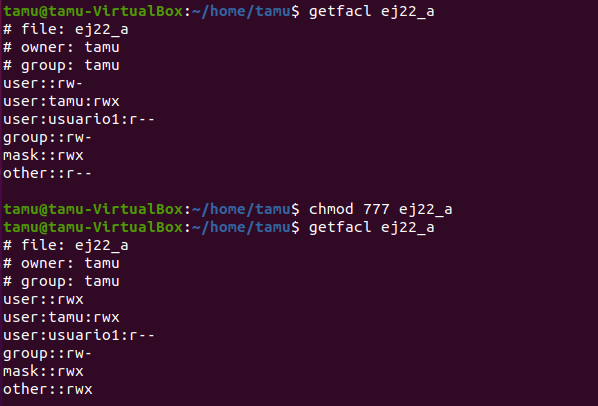
Como podemos observar, a mi usuario (tamu), del grupo tamu, le he otorgado todos los permisos; mientras que al usuario usuario1, del grupo usuario1, solo le he otorgado permisos de lectura.

**a)** **Salvaguarda la ACL del fichero copiándola sobre otro**



Como podemos ver, con el parámetro –set-file de setfacl podemos poner como ACL los parámetros de un fichero. Si en vez de poner el nombre de un fichero, ponemos un -, tomara estos parámetros de la entrada estándar, que en nuestro caso es la salida del comando getfacl del otro fichero.

**b) Cambia los permisos del fichero usando la orden chmod. ¿Qué ocurre con la ACL asociada?**



Los permisos que se referían al usuario, grupo y otros han cambiado, pero los permisos “especiales”, como el del usuario1, se mantiene igual.