**PROPIEDAD Y PERMISOS**

15.1 Introducción

Este es Lab 15: Propiedad y Permisos. Mediante la realización de esta práctica de laboratorio, los estudiantes aprenderán a ajustar y ver la propiedad y los permisos de los archivos y directorios.

En este laboratorio llevarás a cabo las siguientes tareas:

* Ver y entender los permisos de los archivos y directorios.
* Utilizar el comando chmod para cambiar los permisos.
* Cambiar la propiedad con los comandos chown y chgrp.

## 15.2 Permisos de los Archivos y la Propiedad

En esta tarea vas a crear los archivos y directorios, ver y configurar sus permisos y propiedad.

15.2.1 Paso 1

Crea dos directorios y dos archivos en el directorio /tmp:

cd /tmp mkdir priv-dir pub-dir touch priv-dir/priv-file touch pub-dir/pub-file

Ve el contenido de los directorios:

ls -l priv-dir ls -l pub-dir

Tu salida debe contener lo siguiente:

Observa que para cada archivo que se muestra, el primer carácter de la línea es un guión -. Esto transmite que los artículos son archivos regulares. El primer carácter de la lista indica el tipo de archivo, donde d indica un directorio, - es un archivo regular, les un enlace simbólico, b es un archivo de dispositivo de bloques, c es un archivo de dispositivo de caracteres, p es un archivo canalizador y s es un archivo de ranura.

Los siguientes nueve caracteres están en tres grupos de tres caracteres. El primer grupo de tres caracteres (rw- en el ejemplo anterior) son los permisos del usuario propietario, los siguientes tres caracteres (rw- en el ejemplo anterior) son los permisos del propietario del grupo y los tres últimos caracteres (r-- en el ejemplo anterior) representan todos los demás permisos (denominados «otros»).

Al visualizar los permisos, r indica el permiso leer, w indica el permiso escribir y x indica el permiso de ejecución. El guión - indica que ese permiso no se ha concedido.

Después de los permisos viene un número de enlaces indicando cuántos archivos están vinculados a este archivo. A continuación, ves al usuario propietario, al propietario del grupo, el tamaño del archivo, la fecha/hora de la última modificación del archivo y el nombre del archivo.

Si quieres que un directorio sea más privado, a continuación, puedes utilizar el comando chmod para quitar los permisos que otros tienen en el directorio. Utiliza el comando chmodpara quitar los permisos del otro para leer y ejecutar:

ls -ld priv-dir/ chmod o-rx priv-dir/ ls -ld priv-dir/

Ahora la salida muestra que otros no tienen permiso o acceso al priv-dir:

Utilizaste el comando chmod para modificar los permisos para los demás mediante el uso del carácter o seguido de un carácter + o un carácter - para sumar o restar los permisos. El carácter = se puede utilizar para fijar un permiso exacto.

Puedes utilizar una u en lugar de una o para modificar los permisos del usuario propietario. Utiliza una g si quieres cambiar los permisos para el propietario del grupo.

Para modificar los permisos de todo el mundo utiliza una a r en lugar de o, u o g.

|  |  |
| --- | --- |
| chmod a+x *file* | #proporciona todos los permisos de ejecución |
| chmod g-w *file* | #elimina el permiso de escritura para los propietarios del grupo |
| chmod go+r *file* | #agrega permiso de lectura para el propietario del grupo y otros |
| chmod o=rwx | #configura permisos de otros para leer, escribir y ejecutar |

## 15.2.4 Paso 4

Si quieres que un directorio sea más público, a continuación, puedes utilizar el comando chmod para agregar el permiso de escritura para otros:

ls -ld pub-dir/ chmod o+w pub-dir/ ls -ld pub-dir/

Ahora, tu salida muestra que los demás tienen permiso de escritura en el directorio (pueden añadir o eliminar archivos dentro del directorio):

## 15.2.5 Paso 5

Utiliza el comando chmod para eliminar cualquier permiso del grupo u otros en el archivo priv-file:

ls -l priv-dir/priv-file chmod g-rw,o-r priv-dir/priv-file ls -l priv-dir/priv-file

El resultado debe ser similar al siguiente:

## 15.2.6 Paso 6

Concede a todos los usuarios los mismos permisos de lectura y escritura para el archivo pub-file:

ls -l pub-dir/pub-file chmod a=rw pub-dir/pub-file ls -l pub-dir/pub-file

El resultado debe ser similar al siguiente:

## 15.2.7 Paso 7

Crea un archivo test.sh en el directorio /tmp con el contenido de «date»:

echo "date" > test.sh

**sysadmin@localhost:/tmp$** echo "date" > test.sh **sysadmin@localhost:/tmp$**

Si un archivo contiene comandos, entonces los comandos pueden correr o se pueden ejecutar, si el archivo tiene permiso de ejecución para el usuario. El proceso de crear un archivo ejecutable requiere dar el permiso de ejecución en el archivo. Sin este permiso, el archivo no puede ser tratado como un programa.

## 15.2.8 Paso 8

Intenta ejecutar el archivo test.sh; debería fallar. Ve los permisos en el archivo para ver por qué:

./test.sh ls -l test.sh

## 15.2.9 Paso 9

Sólo el usuario propietario de un archivo (o el usuario root) puede cambiar los permisos de un archivo. Como usuario propietario, date permisos de ejecución y luego ejecuta test.sh:

chmod u+x test.sh ls -l test.sh ./test.sh

La salida muestra el permiso de ejecución agregado para el usuario propietario y la fecha y la hora actual desde la ejecución del comando date dentro de tu archivo script test.sh:

Hasta ahora, viste cómo utilizar el comando chmod con la *notación simbólica*, donde se utilizan símbolos para representar quién (u, g, o, y a), cómo (+, -, o =), y qué cambiar (r, w, y x). El comando chmod también se puede utilizar con un valor numérico que representa los permisos del usuario propietario, propietario del grupo y otros en lo que se llama *notación octal*.

Para utilizar el comando chmod con la notación octal, primero hay que entender el valor octal de los permisos:

|  |  |
| --- | --- |
| Read (r) (leer) | 4 |
| Write (w) (escribir) | 2 |
| Execute (x) (ejecución) | 1 |

A partir del último listado del archivo test.sh, se mostró que los permisos eran rwx para el usuario propietario, rw para el propietario del grupo, y r para otros. Para expresar estos permisos en notación octal, se calcula un total para cada propiedad.

Como resultado se calcularía el total del permiso del usuario como 4 + 2 + 1, o 7, donde 4 es permiso de lectura, el 2 de escritura, y 1 de ejecución.

El permiso total del grupo sería 4 + 2 o 6, donde 4 es permiso de lectura y 2 de escritura.

El permiso de propiedad para otros sería simplemente 4, permiso de lectura.

Poniendo todo junto, el valor octal para los permisos actuales sería 764.

## 15.2.10 Paso 10

Utiliza el comando stat para verificar el valor octal para los permisos (acceso) al archivo test.sh:

stat test.sh

Si quieree cambiar estos permisos usando la notación octal para dar al grupo y a otros permiso de ejecución, entonces deberías usar los tres números siguientes:

* 7 (lectura, escritura y ejecución) para el usuario propietario
* 7 (lectura, escritura y ejecución) para el propietario del grupo
* 5 (leer y ejecutar) para otros

El nuevo modo, o el número octal para los permisos sería entonces 775.

## 15.2.11 Paso 11

Utilizando la notación octal, modifica los permisos del archivo test.sh de tal modo, que todo el mundo pueda ejecutar el archivo:

chmod 775 test.sh ls -l test.sh

Hay dos comandos que pueden afectar la propiedad de los archivos. El comando chown sólo puede ser ejecutado por el usuario root y puede cambiar el usuario propietario de un archivo o el usuario y grupo propietario de un archivo.

El comando chgrp se puede utilizar ya sea por el usuario propietario de un archivo o por el usuario root.

El comando chgrp sólo cambia el grupo propietario de un archivo.

Cuando un usuario que sea un root utiliza el comando chgrp, sólo puede cambiar la propiedad del grupo a un grupo del sea miembro. El usuario root puede utilizar chgrp para cambiar la propiedad de grupo de cualquier archivo a cualquier grupo.

## 15.2.12 Paso 12

Cambia al usuario root para que puedas ejecutar tanto el comando chown como el chgrppara las pertenencias a grupos a cualquier grupo:

su - (proporciona la contraseña root: netlab123)

## 15.2.13 Paso 13

Cambia el directorio /tmp y lista los detalles del archivo pub-dir, y luego su contenido:

cd /tmp ls -ld pub-dir ls -l pub-dir/pub-file

Observa que la salida muestra el directorio y el archivo propiedad del usuario sysadmin, y del grupo sysadmin:

## 15.2.14 Paso 14

Utiliza el comando chown para cambiar al usuario y al propietario de grupo del archivo pub-dir al usuario root y al grupo root. A continuación, ve los detalles del directorio:

chown root:root pub-dir ls -ld pub-dir

La salida debe mostrar que el usuario y los propietarios de grupos han cambiado:

## 15.2.15 Paso 15

Utiliza el comando chown para cambiar al usuario propietario del archivo lpub-file al usuario bin:

chown bin pub-dir/pub-file ls -l pub-dir/pub-file

Ahora la salida muestra que el usuario propietario ha sido actualizado a bin:

## 15.2.16 Paso 16

Visualiza los detalles del archivo priv-dir y su contenido:

ls -ld priv-dir ls -l priv-dir/priv-file

La salida debe demostrar que priv-dir es propiedad del usuario sysadmin y del grupo sysadmin:

## 15.2.17 Paso 17

Cambia el propietario del grupo del archivo priv-dir y priv-file al grupo users de forma recursiva con el comando chgrp y visualiza los archivos actualizados:

ls -ld priv-dir ls -l priv-dir/priv-file chgrp -R users priv-dir ls -ld priv-dir ls -l priv-dir/priv-file

Tu salida refleja que al aplicar los cambios de forma recursiva a un directorio, los cambios se aplican al directorio y a todo lo que contiene. Esto significaría que este cambio se aplica a cada subdirectorio bajo priv-dir, cada archivo en priv-dir y a todos sus subdirectorios.

**PERMISOS ESPECIALES**

16.1 Introducción

Este es Lab 16: Los Permisos Especiales. Mediante la realización de esta práctica de laboratorio, los estudiantes aprenderán acerca de los permisos especiales y diferentes tipos de archivos de enlace.

En este laboratorio llevarás a cabo las siguientes tareas:

* Visualizar los archivos con permisos especiales
* Crear vínculos físicos y simbólicos

## 16.2 Visualización de Permisos Especiales

En esta tarea encontrarás y entenderás el propósito de los permisos especiales más allá de lectura, escritura y ejecución.

## 16.2.1 Paso 1

Lista los detalles de los directorios /tmp y /var/tmp:

ls -ld /tmp ls -ld /var/tmp

La salida muestra que los permisos en estos directorios son iguales:

Los directorios /tmp y /var/tmp tienen permisos de leer, escribir y ejecución para todos. Además de los directorios personales de los usuarios, estos dos directorios «temporales» son las ubicaciones en el sistema de archivos donde los usuarios normales pueden crear nuevos archivos o directorios.

Esto plantea un problema: si todos los usuarios pueden crear nuevos archivos, también pueden eliminar los archivos existentes. Esto se debe a que el permiso de escritura en un directorio otorga a los usuarios la capacidad de agregar y eliminar los archivos en un directorio.

La t en la columna de ejecución para los otros permisos indica que este directorio tiene configurado el permiso sticky bit. Este permiso especial significa que a pesar de que todo el mundo puede añadir archivos en estos directorios, sólo el usuario que crea un archivo puede eliminar ese archivo.

El usuario root no se ve afectado por este permiso, ya que esa cuenta puede eliminar todos los archivos en el directorio, independientemente de la propiedad.

## 16.2.2 Paso 2

Visualiza los permisos en el archivo /etc/shadow:

ls -l /etc/shadow

El resultado muestra que el usuario root tiene permisos de leer y escribir, los miembros del grupo shadow tienen permiso de lectura, mientras otros no tienen ningún permiso para este archivo:

Al igual que los otros archivos ubicados en el directorio /etc, el archivo /etc/shadow contiene información acerca de la configuración host. En concreto, el archivo /etc/shadow contiene las contraseñas cifradas de todas las cuentas de usuario locales e información sobre la caducidad de contraseñas (el tiempo que una contraseña es válida). Dado que esta es una información muy sensible, el acceso a este archivo está limitado al usuario root y comandos ejecutados comandos como root, así como los miembros del grupo shadow.

Cuando un usuario actualiza su contraseña con el comando passwd, el comando passwd se ejecuta con un permiso especial llamado setuid. El permiso setuid hace que un archivo se ejecute como el usuario que posee el archivo, en lugar del usuario que está ejecutando realmente el comando.

Si tu conocimiento proviene de Microsoft Windows, puedes pensar que setuid sería como la característica de «Ejecutar como administrador» en Windows.

## 16.2.3 Paso 3

Visualiza los permisos del archivo /usr/bin/passwd:

ls -l /usr/bin/passwd

El listado de este archivo muestra:

Observa la s en la columna de permiso de ejecución del usuario. Esto indica que el archivo tiene el conjunto de permisos setuid, por lo que se ejecuta como el usuario que lo posee (root) en lugar del usuario que ejecuta el comando.

Por lo tanto, el comando passwd puede actualizar el archivo /etc/shadow, conforme se ejecuta como el usuario root  (recuerda que el usuario root puede editar cualquier archivo, independientemente de los permisos en el archivo).

## 16.2.4 Paso 4

Visualiza los permisos del comando /usr/bin/wall:

ls -l /usr/bin/wall

Ahora la salida muestra la s en la columna de ejecución para el grupo:

La s en la columna de ejecución de grupo indica que este archivo tiene el conjunto de permisos setgid, por lo que se ejecuta como el grupo que lo posee (tty) en lugar del grupo del usuario que ejecuta el comando. Por lo tanto, el comando wall puede escribir en todas las terminales (ttys) conforme se ejecuta como el grupo de tty.

Nota: Esto es muy similar al permiso setuid, pero en lugar de ejecutar el comando como el usuario propietario del programa, el comando se ejecuta como el propietario del grupo del programa.

Hasta ahora has visto tres tipos de permisos especiales: sticky bit en un directorio, setuid en un archivo ejecutable y setgid en un archivo ejecutable. Como has visto, estos tres tipos están presentes en un sistema típico.

Puedes utilizar un tipo de permiso especial; El permiso setgid también se puede aplicar a un directorio.

Si ves la s en la columna de ejecutar para el grupo propietario del directorio, el directorio tiene el permiso setgid. Cuando un directorio tiene el permiso setgid, entonces cualquier nuevo archivo o directorio creado en ese directorio será automáticamente propiedad del grupo que posee el directorio, no del grupo del usuario que creó el archivo.

## 16.3 Los Vínculos Físicos y Simbólicos

### 16.3 Los Vínculos Físicos y Simbólicos

En esta tarea vas a crear y utilizar los vínculos físicos y simbólicos.

## 16.3.1 Paso 1

Cambia a tu directorio home:

cd

## 16.3.2 Paso 2

Crea un archivo llamado source que contiene el texto «data» utilizando la redirección:

echo "data" > source

## 16.3.3 Paso 3

Visualiza los detalles e información del i-nodo del archivo source:

ls -li source

La salida resaltada muestra que el archivo tiene el número i-nodo 2076 (este número puede variar de un sistema a otro) y un conteo de vínculos 1:

El sistema operativo Linux utiliza los i-nodos para realizar un seguimiento de la información acerca de un archivo. Una entrada de directorio asocia un i-nodo con un nombre de archivo.

Al crear un vínculo físico se crea otra entrada de directorio asociado con un i-nodo existente, y aumentará el número de los vínculos.

Los vínculos físicos te permiten tener varios nombres para referirse a un mismo archivo. Si se elimina cualquiera de estos nombres, aún puedes utilizar los otros nombres para hacer referencia al archivo.

De hecho, estos otros nombres, incluso se pueden utilizar para crear vínculos adicionales. Todos estos nombres se consideran equivalentes, ya que todos se refieren a un i-nodo existente.

Nota: No puedes crear vínculos físicos a los directorios. Además, un vínculo físico a un archivo también tiene que existir dentro del mismo sistema de archivos (partición) igual que el archivo al que se vincula.

## 16.3.4 Paso 4

Utiliza el comando In para crear un vínculo físico. Visualiza los detalles e información del i-nodo del archivo de vínculo físico original y el nuevo:

ln source hardlink ls -li source hardlink

Observa que el archivo hardlink y el archivo original source comparten el mismo i-nodo:

## 16.3.5 Paso 5

### 16.3.5 Paso 5

Utiliza el comando In para crear un vínculo físico al archivo source. Visualiza los detalles e información del i-nodo del archivo de vínculo físico original y el nuevo:

ln hardlink hardtoo ls -li hardlink hardtoo source

Observa que el resultado sigue mostrando que los vínculos físicos comparten el mismo i-nodo y el número de vínculos aumenta en todos los vínculos cuando se añade uno nuevo:

## 16.3.6 Paso 6

### 16.3.6 Paso 6

Retira el último vínculo que se creó y lista los archivos source y hardlink:

rm hardtoo ls -li source hardlink

Observa que el número de vínculos se disminuye cuando se elimina uno de los archivos con el vínculo físico:

## 16.3.7 Paso 7

Elimina el archivo hardlink y lista los detalles del archivo source:

rm hardlink ls -li source

**sysadmin@localhost:~$** rm hardlink **sysadmin@localhost:~$** ls -li source 2076 -rw-rw-r-- 1 sysadmin sysadmin 5 Apr 12 13:27 source **sysadmin@localhost:~$**

Otro tipo de vínculo que puedes crear se conoce como un vínculo simbólico o vínculo blando. Los vínculos simbólicos no aumentan el número de vínculos de los archivos con los que están vinculados.

Los archivos de vínculos simbólicos tienen su propio i-nodo y el tipo de archivo. En lugar de vincular y compartir un i-nodo, se vinculan al nombre de archivo. A diferencia de los vínculos físicos, los vínculos simbólicos pueden estar vinculados a los directorios y pueden atravesar los dispositivos y las particiones de sus objetivos.

## 16.3.8 Paso 8

Crea un vínculo simbólico al archivo source y ve los detalles de los dos archivos:

ln -s source softlink ls -li source softlink

La salida resaltada muestra que el archivo softlink y source tienen diferentes i-nodos. Observa que el tipo de vínculo del archivo se crea al realizar un vínculo simbólico. Los permisos del enlace son irrelevantes, ya que son los permisos del archivo de destino que determinan el acceso:

## 16.3.9 Paso 9

Crea un enlace simbólico al directorio /proc y muestra el vínculo:

ln -s /proc crossdir ls -l crossdir

El éxito de estos comandos muestra que los vínculos simbólicos pueden hacer referencia a los directorios, y pueden cruzar de un sistema de archivos a otro, que son dos cosas que los enlaces físicos no pueden hacer: