🔐 Taller Práctico: Permisos y Propietarios en Linux

📝 Ejercicio 1: Permisos básicos en modo simbólico

- **Objetivo:** Modificar permisos usando letras (u, g, o) y operadores (+, -, =).
- **Paso a paso:**
    1. Crea un archivo: `touch informe.txt`
    2. Quita todos los permisos al grupo y a otros: `chmod go-rwx informe.txt`
    3. Dale permiso de lectura y ejecución al grupo: `chmod g+rx informe.txt`
- **Explicación:** El modo simbólico es intuitivo: `u` (usuario/dueño), `g` (grupo), `o` (otros). Los operadores `+` y `-` añaden o quitan permisos específicos sin alterar el resto. 👤

📝 Ejercicio 2: Permisos básicos en modo octal (numérico)

- **Objetivo:** Configurar permisos absolutos usando la suma de bits (4-Lectura, 2-Escritura, 1-Ejecución).
- **Paso a paso:**
    1. Configura el archivo para que el dueño haga todo, el grupo solo lea y otros nada: `chmod 740 informe.txt`
    2. Verifica el resultado: `ls -l informe.txt`
- **Explicación:** `7` (4+2+1) es rwx, `4` (4+0+0) es r--, y `0` es ---. Es la forma más rápida y común de establecer permisos en servidores. 🔢

📝 Ejercicio 3: Cambio de propietario (chown) y uso de sudo

- **Objetivo:** Entender que solo el superusuario puede cambiar el dueño de un archivo.
- **Paso a paso:**
    1. Intenta cambiar el dueño a root sin privilegios: `chown root informe.txt` (Dará error).
    2. Hazlo usando privilegios de administrador: `sudo chown root informe.txt`
    3. Verifica el cambio: `ls -l informe.txt`
- **Explicación:** Por seguridad, un usuario normal no puede "regalar" la propiedad de sus archivos. Solo `root` (vía `sudo`) tiene permiso para ejecutar `chown`. 🔑

📝 Ejercicio 4: Cambio de grupo (chgrp)

- **Objetivo:** Cambiar la asociación de grupo de un archivo.
- **Paso a paso:**
    1. Cambia el grupo del archivo al grupo "adm": `sudo chgrp adm informe.txt`
    2. Comprueba que ahora el dueño es `root` y el grupo es `adm`.

- **Explicación:** El comando `chgrp` funciona de forma similar a `chown`, pero se limita a la gestión de grupos. También puedes usar `sudo chown root:adm informe.txt` para cambiar ambos a la vez. 👥

📝 Ejercicio 5: El bit SUID (Set User ID)

- **Objetivo:** Permitir que un archivo se ejecute con los permisos del dueño (root) en lugar de los del usuario que lo lanza.
- **Paso a paso:**
  1. Crea un script falso: `touch ejecutable_seguro`
  2. Asigna el bit SUID: `chmod 4755 ejecutable_seguro`
  3. Observa el permiso en el listado: `ls -l ejecutable_seguro`
- **Explicación:** Verás una `s` minúscula en el lugar de la `x` del dueño (`rwsr-xr-x`). Esto indica que el programa se ejecutará con los privilegios del propietario del archivo. 🛡️

📝 Ejercicio 6: El bit SGID (Set Group ID) en directorios

- **Objetivo:** Hacer que todos los archivos creados dentro de una carpeta hereden el grupo de la carpeta madre.
- **Paso a paso:**
  1. Crea una carpeta: `mkdir compartido_grupo`
  2. Cambia su grupo: `sudo chgrp adm compartido_grupo`
  3. Activa el bit SGID: `chmod 2775 compartido_grupo`
  4. Crea un archivo dentro: `touch compartido_grupo/prueba.txt` y mira su grupo.
- **Explicación:** Verás una `s` en la sección del grupo. Es fundamental en entornos colaborativos para que los miembros del grupo siempre puedan leer lo que crean otros. 📂

📝 Ejercicio 7: El Sticky Bit (Bit de permanencia)

- **Objetivo:** Proteger archivos en carpetas públicas para que solo su dueño pueda borrarlos.
- **Paso a paso:**
  1. Crea una carpeta pública: `mkdir carpeta_publica`
  2. Dale permisos totales: `chmod 777 carpeta_publica`
  3. Activa el Sticky Bit: `chmod +t carpeta_publica`
  4. Verifica con `ls -ld carpeta_publica`
- **Explicación:** Verás una `t` al final (`rwxrwxrwt`). Esto impide que un usuario borre los archivos de otro, aunque la carpeta tenga permisos totales (como ocurre en `/tmp`). 📍

📝 Ejercicio 8: Permisos recursivos

- **Objetivo:** Aplicar cambios a toda una estructura de subcarpetas de una vez.
- **Paso a paso:**
    1. Crea una carpeta con contenido: `mkdir -p web/html ; touch web/index.html`
    2. Cambia dueño y permisos a todo el árbol: `sudo chown -R root:root web ; chmod -R 755 web`
- **Explicación:** El parámetro `-R` (Recursive) es extremadamente potente y peligroso; aplica la orden a la carpeta, subcarpetas y todos los archivos internos. 🌳

📝 Ejercicio 9: Máscara de usuario (umask)

- **Objetivo:** Determinar los permisos por defecto que tendrán los archivos nuevos.
- **Paso a paso:**
    1. Escribe `umask` para ver tu valor actual.
    2. Cambia el umask temporalmente: `umask 0077`
    3. Crea un archivo: `touch secreto.txt` y mira sus permisos con `ls -l`.
- **Explicación:** El `umask` resta permisos. Un `umask 077` hará que los archivos nuevos solo tengan permisos para el dueño (600), ocultándolos del resto del sistema automáticamente. 😷

📝 Ejercicio 10: Verificación final con nombre de usuario

- **Objetivo:** Confirmar la identidad y los grupos antes de una auditoría de permisos.
- **Paso a paso:**
    1. Ejecuta `id` para ver tu UID, GID y grupos.
    2. Ejecuta `whoami` para confirmar tu nombre de usuario.
- **Explicación:** Antes de asignar permisos, es vital saber en qué grupos estás. Si intentas acceder a un archivo con permisos de grupo `adm`, pero el comando `id` no muestra que perteneces a ese grupo, el acceso será denegado. 🆔