**Comandos de permiso**

Anteriormente, exploraste los permisos de archivo y los comandos que puedes usar para mostrarlos y cambiarlos. En esta lectura, revisarás estos conceptos y también te centrarás en un ejemplo de cómo estos comandos funcionan juntos al poner en práctica el principio de mínimo privilegio.

**Permisos de lectura**

En Linux, los permisos se representan con una cadena de 10 caracteres. Estos son algunos permisos:

* **read** (lectura): para archivos, es la capacidad de leer el contenido de los archivos; para directorios, es la capacidad de leer todo el contenido del directorio, incluidos los archivos y subdirectorios
* **write** (escritura): para archivos, es la capacidad de realizar modificaciones en el contenido de los archivos; para directorios, es la capacidad de crear archivos nuevos en el directorio
* **execute** (ejecución): para archivos, es la capacidad de ejecutar el archivo si se trata de un programa; para directorios, es la capacidad de ingresar al directorio y acceder a sus archivos.

Estos permisos se otorgan a los siguientes tipos de propietarios:

* **user** (usuario): propietario del archivo
* **group** (grupo): un grupo más grande del que el propietario forma parte
* **other** (otros usuarios): todos los demás usuarios del sistema

Cada carácter en la cadena de 10 caracteres transmite información diferente sobre estos permisos. La siguiente tabla describe el propósito de cada carácter:

| **Carácter** | **Ejemplo** | **Significado** |
| --- | --- | --- |
| 1.º | **d**rwxrwxrwx | tipo de archivo   * **d** para el directorio * **-** para un archivo normal |
| 2.º | d**r**wxrwxrwx | permisos de lectura para el usuario   * **r** si el usuario tiene permisos de lectura * **-** si el usuario no tiene permisos de lectura |
| 3.º | dr**w**xrwxrwx | permisos de escritura para el usuario   * **w** si el usuario tiene permisos de escritura * - si el usuario no tiene permisos de escritura |
| 4.º | drw**x**rwxrwx | permisos de ejecución para el usuario   * **x** si el usuario tiene permisos de ejecución * **-** si el usuario no tiene permisos de ejecución |
| 5.º | drwx**r**wxrwx | permisos de lectura para el grupo   * **r** si el grupo tiene permisos de lectura * **-** si el grupo no tiene permisos de lectura |
| 6.º | drwxr**w**xrwx | permisos de escritura para el grupo   * **w** si el grupo tiene permisos de escritura * **-** si el grupo no tiene permisos de escritura |
| 7.º | drwxrw**x**rwx | permisos de ejecución para el grupo   * **x** si el grupo tiene permisos de ejecución * **-** si el grupo no tiene permisos de ejecución |
| 8.º | drwxrwx**r**wx | permisos de lectura para otros usuarios   * **r** si el otro tipo de propietario tiene permisos de lectura * **-** si el otro tipo de propietario no tiene permisos de lectura |
| 9.º | drwxrwxr**w**x | permisos de escritura para otros usuarios   * **w** si el otro tipo de propietario tiene permisos de escritura * **-** si el otro tipo de propietario no tiene permisos de escritura |
| 10.º | drwxrwxrw**x** | permisos de ejecución para otros usuarios   * **x** si el otro tipo de propietario tiene permisos de ejecución * **-** si el otro tipo de propietario no tiene permisos de ejecución |

**Exploración de los permisos existentes**

Puedes usar el comando **ls** para investigar quién tiene permisos en archivos y directorios. Antes, aprendiste que **ls** muestra los nombres de los archivos y directorios en el directorio de trabajo actual.

Existen otras opciones que puedes agregar al comando **ls** para que sea más específico. Algunas de estas opciones proporcionan detalles sobre los permisos. Estas son algunas que utilizan el comando **ls** y resultan importantes para las/los analistas de seguridad:

* **ls -a**: muestra archivos ocultos. Los archivos ocultos comienzan con un punto (**.**) al principio.
* **ls -l**: muestra permisos para archivos y directorios. También muestra otra información adicional, incluido el nombre del propietario, el grupo, el tamaño del archivo y la hora en que fueron modificados por última vez.
* **ls -la**: muestra los permisos de archivo y directorios, incluidos los archivos ocultos. Se trata de una combinación de las otras dos opciones.

**Cambio de permisos**

El **principio de mínimo privilegio** es el concepto de otorgar solo el acceso y la autorización mínimos necesarios para ejecutar una tarea o función. En otras palabras, los/las usuarios/as no deben tener privilegios que estén más allá de lo necesario. No seguir el principio del mínimo privilegio puede crear riesgos de seguridad.

El comando **chmod** puede ayudarte a administrar esta autorización. El comando **chmod** cambia los permisos en archivos y directorios.

**Cómo usar chmod**

El comando **chmod** requiere dos argumentos. El primer argumento indica cómo cambiar los permisos, y el segundo indica el archivo o directorio para el que quieres cambiar los permisos. Por ejemplo, el siguiente comando agregaría todos los permisos a **login\_sessions.txt**:

**chmod u+rwx,g+rwx,o+rwx login\_sessions.txt**

Si quisieras quitar todos los permisos, podrías usar

**chmod u-rwx,g-rwx,o-rwx login\_sessions.txt**

Otra forma de asignar estos permisos es usar el signo igual (**=**) en este primer argumento. Al usar **=** con **chmod**, se establecen o asignan los permisos exactamente según se especificó. Por ejemplo, el siguiente comando establecería permisos de lectura para **login\_sessions.txt** para usuario, grupo y otros usuarios:

**chmod u=r,g=r,o=r login\_sessions.txt**

Este comando sobreescribe permisos existentes. Por ejemplo, si antes el usuario tenía permisos de escritura, estos permisos de escritura se eliminan después de especificar permisos de solo lectura con **=**.

La siguiente tabla revisa cómo se usa cada carácter dentro del primer argumento de **chmod**:

| **Carácter** | **Descripción** |
| --- | --- |
| **u** | indica que se realizarán cambios en los permisos de usuario |
| **g** | indica que se realizarán cambios en los permisos de grupo |
| **o** | indica que se realizarán cambios en los permisos de otros usuarios |
| **+** | agrega permisos al usuario, grupo u otros usuarios |
| **-** | elimina permisos del usuario, grupo u otros usuarios |
| **=** | asigna permisos para el usuario, grupo u otros usuarios |

**Nota:** Cuando hay cambios de permiso en más de un tipo de propietario, hay que usar comas para separar los cambios para cada tipo de propietario. No se debe agregar espacios después de esas comas.

**El principio de mínimo privilegio en acción**

Como analista de seguridad, es posible que te encuentres en una situación como esta: hay un archivo llamado **bonuses.txt** dentro de un directorio de remuneración. El propietario de este archivo es un miembro del departamento de Recursos Humanos cuyo nombre de usuario es **hrrep1**. Se decidió que **hrrep1** necesita acceso a este archivo. Pero, dado que este archivo contiene información confidencial, nadie más en el grupo **hr** debería acceder a él.

Ejecutas **ls -l** para verificar los permisos de los archivos en el directorio de remuneración y descubres que los permisos para **bonuses.txt** son **-rw-rw----**. El tipo propietario del grupo tiene permisos de lectura y escritura que no se alinean con el principio de mínimo privilegio.

Para remediar la situación, ingresas **chmod g-rw bonuses.txt**. Ahora, solo puede acceder a este archivo el/la usuario/a que lo necesita para llevar a cabo sus responsabilidades laborales.

**Conclusiones clave**

Administrar permisos de directorios y archivos puede ser parte de tu trabajo como analista de ciberseguridad. El uso de **ls** con las opciones **-l** y **-la** te permite investigar los permisos de directorio y archivo. El uso de **chmod** te permite cambiar los permisos de usuario y asegurarte de que estén alineados con el principio de mínimo privilegio.