Conceptos básicos sobre los permisos y propiedades en Linux

https://docs.bluehosting.cl/tutoriales/servidores/conceptos-basicos-sobre-permisos-y-propiedades-en-linux.html

Actualizado el 1 de Agosto, 2016. Por BlueHosting.

¿Alguna vez ha recibido mensajes como: "*Permiso denegado*" o "*La acción solicitada requiere permisos elevados*"? Todos los sistemas Linux están caracterizados por una esencia multitarea, pero también se basan en un entorno multiusuario (varios usuarios pueden usar el mismo equipo a la vez). Una parte crucial de la administración de entornos multiusuario es la asignación y gestión de propiedades y permisos. Este tutorial aborda los conceptos básicos para comprender de forma definitiva el concepto y aplicación de las propiedades y permisos en Linux.

Introducción

Las propiedades y permisos en Linux son conceptos algo complejos, pero su comprensión es esencial para poder operar y mantener su sistema. Ya sea que utilice su servidor para la gestión de su sitio web o para la administración de grandes plataformas, este tutorial le ayudará a comprender la mayoría de los conceptos de los usuarios, grupos y sus permisos.

Tipos de cuenta

Por defecto, Linux distingue entre varios tipos de cuenta con el fin de aislar los procesos y cargas de trabajo. Linux trabaja con cuatro tipos de cuenta:

- root
- System
- Normal
- Network
-

Una de las recomendaciones básicas de seguridad en Linux (visite nuestra Guía básica de seguridad es limitar el acceso root para proveer un ambiente seguro de trabajo. También se sugiere otorgar el mínimo de privilegios posibles a una cuenta de usuario. La importancia de las buenas prácticas en sistemas multiusuario es crucial para una operación óptima en sus servidores.

Cuenta *root*

root es la cuenta con mayores privilegios en un sistema Linux/UNIX. Esta cuenta tiene la capacidad de llevar a cabo cualquier función de administración, incluyendo: añadir cuentas, cambiar contraseñas de usuarios, examinar archivos *log* e instalar software. Esta cuenta no tiene restricciones de seguridad, y debe tenerse extremo cuidado al usarla.

Si un usuario es el administrador del sistema, y requiere permisos *root* se recomienda que dicho administrador tenga su propio usuario y que ejecute los comandos que requieran privilegios de superusuario usando el prefijo sudo.

Permisos

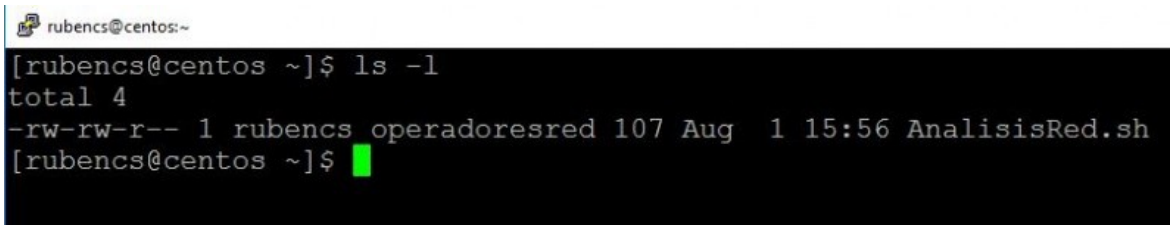
La lectura, escritura y ejecución son los tres parámetros principales de los permisos. Debido a que los usuarios se disponen en grupos cuando sus cuentas son creadas, también puede especificar si ciertos grupos pueden leer, escribir o ejecutar un archivo.

Algunas de las operaciones que requieren privilegios *root* incluyen: crear, remover y administrar cuentas de usuario; remover o modificar archivos del sistema; reiniciar servicios del sistema. En contraste, un usuario regular puede ejecutar operaciones como: ejecutar un cliente de red; operar archivos para los cuales tiene los permisos apropiados; instalar algunos paquetes de software; entre otros.

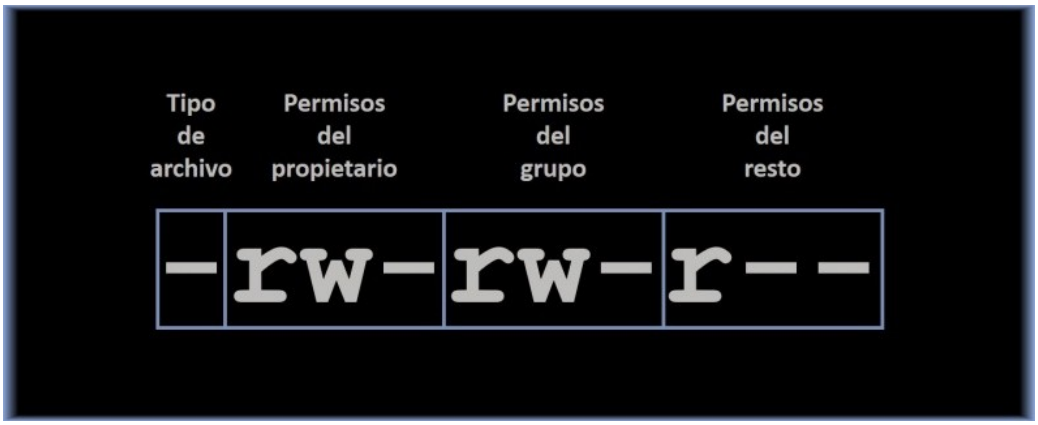
Cómo identificar los permisos y propietarios de los archivos y directorios del sistema

Para aprender a identificar los atributos de un archivo/directorio y determinar cuáles son los permisos y propietarios siga los siguientes pasos:

1. Diríjase a un directorio de su preferencia y ejecute el comando ls -l. Observe el siguiente ejemplo:



   Al ejecutar el comando, veremos una lista de los archivos y subdirectorios que contiene el directorio actual (pwd). En este ejemplo podemos ver el script AnalisisRed.sh acompañado de varios parámetros que iremos entendiendo a lo largo del tutorial.
   El tercer parámetro indica el usuario al cual pertenece el archivo: **rubencs**. El parámetro que le sigue indica el grupo propietario del archivo: **operadoresred**. Este último parámetro podría ser igual al usuario si este no está asociado a ningún grupo.
2. El siguiente parámetro que debemos ver con cuidado es la secuencia -rw-rw-r--, ¿Alguna vez se ha preguntado qué significa cada una de estas letras?, hay que mirarlo paso a paso:
   - Esta secuencia contiene diez (10) caracteres que indican los permisos del archivo o directorio. Para entenderlo debemos dividir la secuencia en el primer carácter y tres grupos de tres caracteres. Miremos más de cerca el archivo del ejemplo:



Tal como se observa en la imagen arriba, el primer carácter indica el tipo de archivo. El primer grupo revela los permisos del usuario propietario del archivo, el siguiente muestra los permisos del grupo propietario y el último grupo indica los permisos que regirán al resto de los usuarios.
Cada una de las letras indica el tipo de permiso, y cada grupo de tres caracteres está ordenado en la forma **rwx**, así:

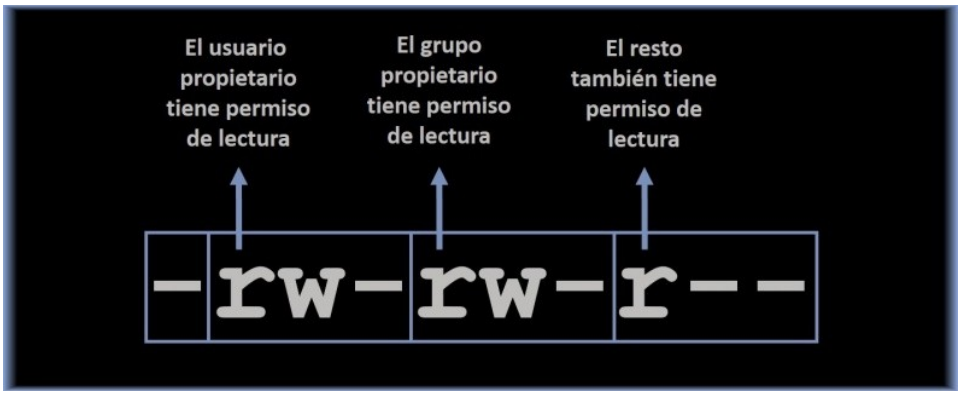
- **r** indica *read* o permiso de lectura;
- **w** indica *write* o permiso de escritura;
- **x** indica *execute* o permiso de ejecución.
- **-** el guion indica que el archivo o directorio carece del permiso correspondiente en dicha posición. Por ejemplo, si en el grupo de permisos del propietario aparece r--, significa que el usuario propietario solo tiene permisos para leer o visualizar el archivo.

- **Tipos de archivo**

  Tal como se mencionó, el primer parámetro indica el tipo de fichero, existen tres principales:
  - **-** indica un archivo regular;
  - **d** indica directorios o carpetas;
  - **l** o *link* indica que es un enlace simbólico;

- **Permisos de lectura**

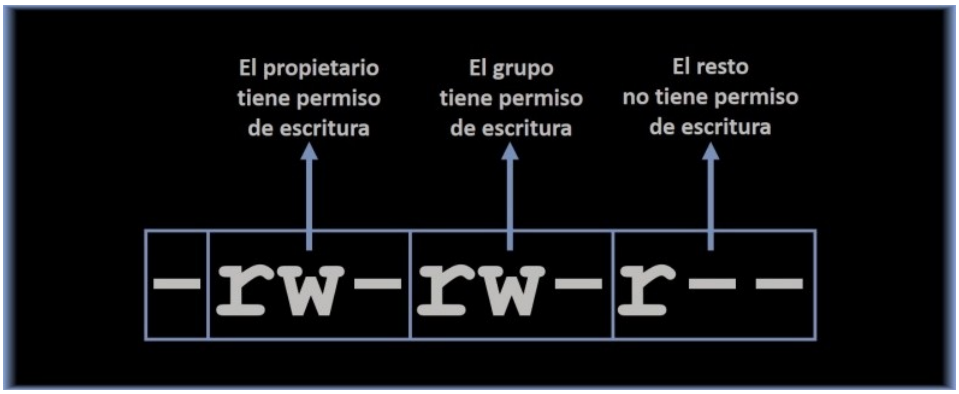
  Este permiso indica que el usuario puede leer o visualizar el contenido de un archivo. Si un usuario tiene permiso de lectura de una carpeta, entonces este podrá ver el contenido de dicho directorio. Para el caso de nuestro ejemplo:



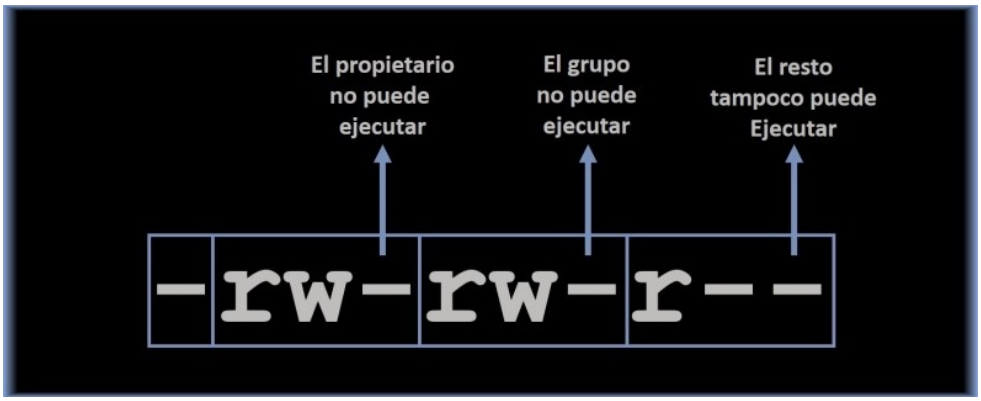
- **Permisos de escritura**

  Los permisos de escritura en un archivo simplemente indican que el usuario puede modificar el contenido del archivo y sus permisos usando chmod o chown. El usuario también podrá eliminar el archivo si tiene permisos de escritura. Cuando el permiso aplica a un directorio, el usuario puede modificar el contenido

dentro de dicho directorio e incluso eliminarlo. Para nuestro ejemplo:



- **Permisos de ejecución**
  El permiso de ejecución se explica por sí solo: un usuario puede ejecutar un archivo solo si posee este tipo de permisos (los archivos ejecutables son las aplicaciones y *scripts*). En nuestro ejemplo, el archivo AnalisisRed.sh es un script, veamos quiénes poseen permisos para ejecutarlo:



En resumen, nuestro archivo puede ser leído por cualquier usuario, modificado por el propietario y el grupo, y no puede ser ejecutado por nadie. En la próxima sección aprenderá cómo cambiar estos permisos.

Usar referencias binarias para configurar permisos

Otra manera muy común de denominar y configurar permisos es usando referencias binarias. Para entender cómo funcionan estas referencias, recuerde que los permisos están dados por tres caracteres **rwx** y que existen tres grupos establecidos para otorgar permisos (propietario, grupo y resto de los usuarios).

Tenga en cuenta que un permiso en forma binaria está representado por **tres números enteros**: el primer número representa los permisos del propietario, el segundo representa los permisos de grupo; y el último número representa los permisos del resto de los usuarios. Los números son una representación binaria de los caracteres **rwx** asignando a cada letra un valor:

- r = 4
- w = 2
- x = 1

Para obtener el número entero que representa cada conjunto de permisos, debe sumar los permisos que desea habilitar para cada conjunto. Si algún permiso no será habilitado no sumará el entero correspondiente a dicho permiso. Por ejemplo, digamos que un archivo posee los siguientes permisos:

-rwxrw-r--

El usuario tiene todos los permisos habilitados (**rwx**), por lo que debe sumar 4+2+1 = **7**. Solo están habilitados los permisos de lectura (r) y escritura (w) en los grupos, por lo tanto, el número entero que identificará los permisos de grupos será **6** (4+2+0). Finalmente, el resto de los usuarios solo puede leer (r) el archivo, este conjunto de permisos se representará entonces con **4** (4+0+0). Así, los permisos de este archivo estarán representados con el número **764**.

Cómo configurar/cambiar los permisos de archivos y directorios

Hay distintos métodos que puede utilizar para cambiar los permisos y propietarios de un archivo o directorio. La explicación y comprensión de las referencias binarias facilitará este proceso:

1. Para cambiar los permisos a través de la línea de comandos se utiliza chmod con la siguiente sintaxis:

   chmod [permisos] nombre_del_archivo

   Puede especificar los permisos a través de una representación binaria, o utilizando la letra inicial a quién va dirigido el permiso (usuario=u; grupo=g; otros=o; todos=a del inglés *all*); seguido del signo + (agregar) - (quitar); y finalmente el permiso correspondiente **r**/**w**/**x**. Ejemplos:

   chmod u+x AnalisisRed.sh

   Otorga el permiso de ejecución al usuario.

   chmod g-r AnalisisRed.sh

   Revoca el permiso de lectura al grupo.

   chmod o-w AnalisisRed.sh

   Revoca el permiso de escritura al resto de los usuarios.

   chmod a+rw AnalisisRed.sh

   Otorga el permiso de lectura y escritura a todos los usuarios.

   chmod 740 AnalisisRed.sh

   Otorga al usuario todos los permisos (rwx); a los grupos solo el permiso de lectura (r); y ningún permiso al resto de los usuarios. Este último método es el más sencillo y recomendable.

2. Para cambiar la asignación de propiedad del usuario o grupo de un archivo (o directorio) se utiliza:

   chown usuario:grupo nombre_del_archivo

   Simplemente se indica cuál usuario y cuál grupos serán propietarios de dicho archivo.

3. Si solo desea cambiar el grupo propietario puede utilizar chgrp nuevo_grupo nombre_del_archivo.

4. Siguiendo con el ejemplo anterior hagamos un par de cambios y veamos cómo afectan los atributos del archivo:

   ```
   chown rubencs:administradores AnalisisRed.sh
   chmod 700 AnalisisRed.sh
   ```

   El primer comando mantuvo a **rubencs** como el propietario del archivo, pero definió un nuevo grupo propietario: **administradores**. El segundo comando otorga todos los permisos a **rubencs** como propietario del archivo, y revoca todos los permisos a los grupos y otros usuarios.
   Si volvemos a usar ls -l para visualizar los permisos del archivo, podremos ver los cambios:

   ```
   [root@centos rubencs]# ls -l
   total 4
   -rwx------ 1 rubencs administradores 107 Aug  1 15:56 AnalisisRed.sh
   ```

   Ahora el propietario (**rubencs**) será capaz de visualizar, modificar y ejecutar el script. Ningún otro usuario (excepto *root*) tendrá acceso al archivo.

Consejos

Al administrar un sistema Linux debe ser muy cauteloso con respecto a la asignación de permisos y propietarios de sus archivos y directorios. La recomendación general es **conceder la menor cantidad de permisos posibles** a un usuario. ¡Los permisos de escritura y ejecución pueden ser peligrosos si no administran con cuidado!

Recuerde ser **consecuente** en la asignación de permisos, de lo contrario podría ocasionar contradicciones. Por ejemplo: Digamos que otorga permisos de lectura del archivo HolaChile.txt al grupo "Administradores". Pero el directorio que contiene el archivo HolaChile.txt no posee permisos de lectura para ese grupo, o es propiedad de un grupo distinto. Si esto ocurre, el grupo "Administradores" no podrá acceder al directorio donde se encuentra el archivo -y por lo tanto- no será capaz de usarlo.

Además, es importante mantener una organización apropiada de los grupos y usuarios. Siempre que cree un usuario nuevo inclúyalo en el grupo adecuado.

Para ver una lista de los grupos en su sistema, incluyendo los usuarios que pertenecen a cada grupo, puede utilizar:

```
sudo cat /etc/group
```

Recursos adicionales

Este es un tema sumamente extenso y no es necesario ahondar en más detalles; sin embargo, este tutorial da un punto de partida sólido para comprender los conceptos básicos. Puede consultar los siguientes recursos en busca de información adicional. Aunque este material es provisto esperando que sea útil, tome en cuenta que no podemos dar fe de la actualidad o precisión de los contenidos externos.

- Artículo de Linux sobre permisos y propiedades.
- Consulte los manuales de los comandos aquí mencionados usando:
- chmod --help;
- chown –help;
- chgrp –help;

http://mural.uv.es/oshuso/8339_permisos_y_atributos.html

Al igual que en Windows, en Linux, los usuarios disponen de ciertos permisos o privilegios que limitan su control sobre el sistema.

Como ya hemos visto en apartados anteriores, para saber los permisos que un usuario tiene sobre determinados directorios, no tenemos más que observar el primer atributo que aparece en cada caso al ejecutar la orden **ls -l**.

Si, además añadimos -d, y el nombre del directorio que queremos, veremos exclusivamente los permisos que tenemos sobre ese directorio. Así, si ejecutamos **$ ls -ld Fotos**, veremos qué permisos tenemos sobre él.

Los tipos de permisos sobre archivos en Linux son los siguientes:

- **Lectura**: Permite fundamentalmente visualizar el contenido del archivo con órdenes como **ls**, **cat**, etc. También permite el uso de órdenes como **cp**.
- **Escritura**: Permite modificar el contenido del archivo. El archivo se puede editar, por ejemplo, con gedit y modificar su contenido sin ningún problema.
- **Ejecución**: Permite ejecutar el archivo como si de un programa ejecutable se tratase. Estos permisos se suelen asignar a archivos Shell, es decir, archivos que realizan funciones propias del sistema operativo, como copias de seguridad, análisis de la integridad del sistema, etc.

La asignación, modificación y eliminación de permisos o atributos sobre los archivos en entorno gráfico se realiza de forma análoga a Windows (botón derecho sobre el directorio o archivo, en **Propiedades**, en la pestaña de **Permisos**).

Cuando creamos un archivo o directorio nuevo, se definen unos permisos concretos. Tienen la siguiente apariencia.

rwx rwx rwx -> Donde para dar permiso o no, consideraremos 0 ó 1.

111 111 100 **->** El propietario y su grupo, tienen permisos de lectura, escritura y ejecución. El resto de usuarios, de lectura.

7 7 4

Existen varios comandos en linux que nos permiten modificar los permisos:

- **chmod:** Se utiliza para cambiar los permisos del archivo o directorio. Tenemos distintas formas de hacerlo. Por ejemplo:

- Ejecutando la orden **$ chmod 777 Ejercicios/Ejer1.txt** asignamos permisos de lectura, escritura y ejecución (control total) a todos los usuarios sobre el fichero Ejer1.txt dentro del directorio Ejercicios.

- Ejecutando la orden **$ chmod ugo+rwx Ejercicios/Ejer1.txt**, conseguimos los mismo. Aquí, podemos variar el significado de la orden según utilicemos **+** (añadimos los permisos que se indican a ese usuario), **-** (le quitamos los permisos indicados), **=** (le asignamos los permisos).

- **chown:** Esta orden se utiliza para cambiar el propietario del fichero o directorio. Como parámetro, le pasaremos el nombre del propietario y el nombre del archivo o directorio. Por ejemplo: **$ chown alumno Cosas**, haremos que el propietario del directorio Cosas pase a ser el usuario alumno.

Existe la opción -R, que hace que el propietario de la carpeta cambie de forma recursiva, es decir, que también afectará el cambio al contenido de ésta.

- **chgrp:** Es análoga a la anterior, sólo que aquí, en lugar de cambiar el propietario, cambiamos el grupo.

Comando	Resultado
$ chmod g+x doc1 $ chmod go-wx doc1	Asigna privilegios de lectura, escritura y ejecución a todos los usuarios del archivo doc1
$ chmod ugo+x doc1	Asigna a todos los usuarios el permiso de ejecución sobre doc1
$ chmod 764 doc1	Quita todos los privilegios a todos los usuarios del archivo doc1

https://blog.carreralinux.com.ar/2016/10/comando-stat-propiedades-archivos-linux/

Comando stat: Conociendo propiedades de nuestros archivos

Publicado el
Gabriel Canepa

Como vimos en una oportunidad, el comando ls -l nos permite observar varias propiedades de un archivo mediante la línea de comandos en Linux. Entre ellos podemos mencionar el tipo de objeto (archivo, directorio, enlace simbólico, dispositivo de bloques, etc), el usuario dueño y grupo propietario, la fecha de última modificación, y el tamaño. Si bien todos esos detalles son importantes, en este post y en el siguiente explicaremos cómo utilizar el comando **stat** para ver más propiedades de nuestros archivos.

El comando stat

En su man page, el propósito de **stat** es enunciado de la siguiente forma:

> display file or file system status

Veamos entonces qué datos sobre el estado de nuestros archivos (o sistema de archivos) podemos averiguar con esta herramienta. Con el modificador -c, seguido por una opción, le indicamos el formato que deseamos que tenga la salida. Por otra parte, si utilizamos el modificador -f, estaremos indicando que queremos ver los datos del sistema de archivos en vez de los correspondientes a un archivo en particular.

Tomaremos como ejemplo un archivo vacío llamado prueba.txt.

```
touch prueba.txt
```

Fecha de último acceso (utilizar %x):

```
stat -c %x prueba.txt
```

Fecha de última modificación (mediante %y):

```
stat -c %y prueba.txt
```

Si ahora «accedemos» al archivo para leerlo con **cat**, y posteriormente consultamos su fecha de último acceso veremos que cambió. Lo mismo sucede si le agregamos contenido con **echo** y luego verificamos su fecha de última modificación.

```
cat prueba.txt
stat -c %x prueba.txt
echo "Nuevo contenido" > prueba.txt
stat -c %y prueba.txt
```

En la Fig. 1 podemos observar los cambios en las fechas de último acceso y modificación con puntos rojos y azules, respectivamente:

```
[root@server ~]# touch prueba.txt
[root@server ~]# stat -c %x prueba.txt
2016-10-14 21:24:01.061303647 +0000
[root@server ~]# stat -c %y prueba.txt
2016-10-14 21:24:01.061303647 +0000
[root@server ~]# cat prueba.txt
[root@server ~]# stat -c %x prueba.txt
2016-10-14 21:24:30.848303647 +0000
[root@server ~]# echo "Nuevo contenido" > prueba.txt
[root@server ~]# stat -c %y prueba.txt
2016-10-14 21:24:52.542303647 +0000
[root@server ~]#
```

Otra opción que nos puede resultar útil es %a, que nos permitirá ver los permisos de prueba.txt en forma octal, tal como apreciamos en la Fig. 2:

stat -c %a prueba.txt

```
[root@server ~]# ls -l prueba.txt
-rw-r--r-- 1 root root 16 Oct 14 21:24 prueba.txt
[root@server ~]# stat -c %a prueba.txt
644
[root@server ~]#
```

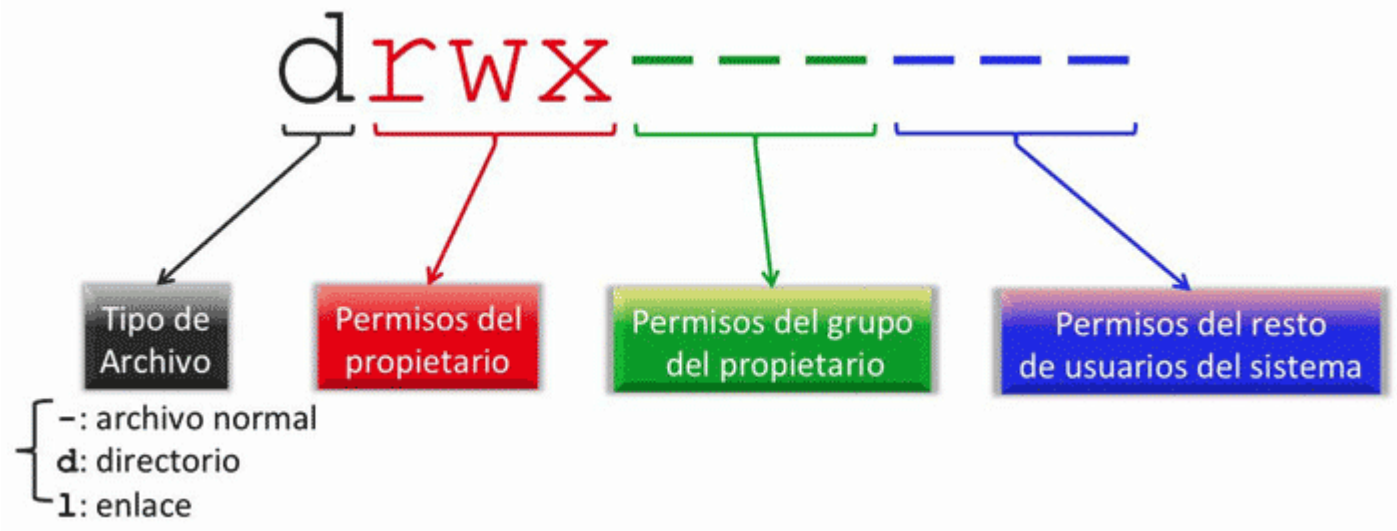
Estas son solamente algunas de las opciones que nos ofrece stat para conocer más a fondo nuestros archivos. En el próximo post continuaremos explicando otras que también son muy útiles y veremos la manera de combinar dos o más en un solo comando. De esa manera la salida de **stat** nos brindará varios datos simultáneamente. ¡No se lo pierdan!

https://inspiretic.wordpress.com/2016/04/24/linux-atributos-de-ficheros/

LINUX – ATRIBUTOS DE FICHEROS

LINUX – ATRIBUTOS DE FICHEROS

VISTA EN GLOBAL



En Linux cada fichero y directorio tiene atributos que determinan el acceso al propietario, grupo y al resto de usuarios. Estos atributos se utilizan para determinar si un usuario o proceso pueden leer, escribir y ejecutar un fichero.

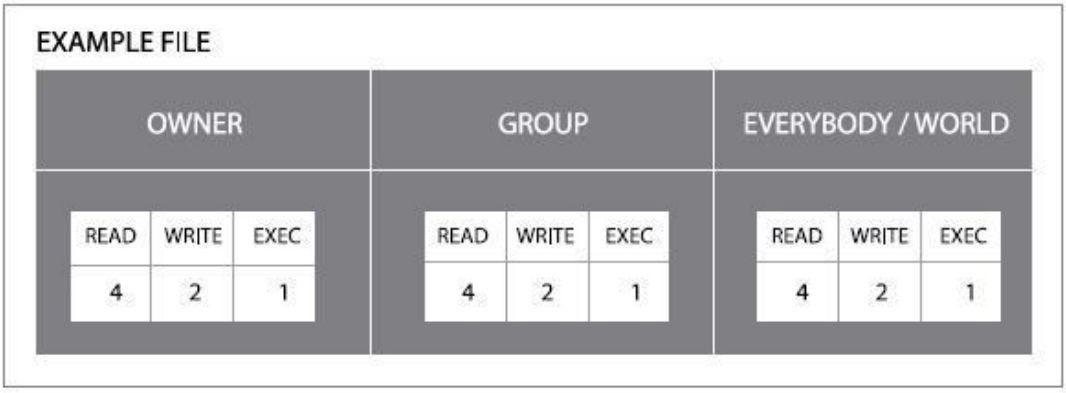
El acceso a los ficheros está garantizado jerárquicamente. El acceso a grupos anula el acceso universal y el acceso del propietario anula al acceso a grupos. Los permisos dados a un fichero o directorio son representados habitualmente en sistema octal numéricamente.

IDEAS CLAVE:

Niveles de permisos: cada fichero y directorio tiene establecidos permisos para los tres niveles: propietario, grupo y resto de usuarios. Los permisos en cada nivel pueden estar representados juntos en octal (un numero de tres dígitos, uno para el propietario, otro para el grupo y otro para el resto).

Tipos de permisos: Los tipos son lectura, escritura y ejecución. El permiso de lectura se representa con un 4, el de escritura con un 2 y el de ejecución con un 1. Si, por ejemplo, un permiso debe dejar leer, escribir y ejecutar, éste se representa como un 7.

r --> read permissions
w --> write permissions
x --> execute permissions



Comando ls -l: al ejecutarlo y devolver un resultado cualquiera, el primer carácter que devuelve nos indica si se trata de un directorio (si hay una d). Luego nos muestra los tres caracteres siguientes (read, write, execute = 7 que se refiere a los permisos activos para el propietario). Luego nos muestra otros 3 caracteres (read, nada y execute = 5) que se refiere a los permisos del grupo. Por último nos muestra otros 3 caracteres (read,nada y execute = 5) que se refiere a los permisos del resto de usuarios activos. La imagen que sigue a continuación muestra gráficamente la explicación que acabamos de hacer:


```
drwxr-xr-x.  3 root root  18 Feb 11 22:49 mnt
```

Comando chmod: este comando se ejecuta para cambiar permisos de un fichero o directorio.

Comando chown: este comando se usa también para cambiar los permisos de un fichero o directorio.

Comando ls

https://www.servidoresadmin.com/comando-linux-ls/