1 Permisos

Linux permite configurar os privilexios e restricións que os usuarios teñen para acceder ao sistema. Esta configuración baséase nunha serie de permisos sobre os ficheiros que se outorgarán aos usuarios e grupos.

1.1 Sumario

2 Usuarios e grupos propietarios

Antes de falar dos permisos debemos saber que en Linux todos os ficheiros pertencen a un usuario e a un grupo. Cando se crea un novo ficheiro, o seu propietario será o usuario que o creou e o grupo do ficheiro será o grupo principal de devandito usuario. Co comando `ls` engadindo a opción `-l` (formato longo) podemos visualizar o usuario propietario e o grupo propietario do ficheiro.

Se un usuario chamado 'alumno01' con grupo principal 'alumnos' crea un novo ficheiro chamado apuntes.txt, o propietario do ficheiro será 'alumno01' e o grupo propietario do ficheiro será 'alumnos', ou o que é o mesmo, o ficheiro pertencerá ao usuario 'alumno01' e ao grupo 'alumnos'.

Ao executar o comando aparecen todos os ficheiros, un por liña. O bloque de 10 carácteres do principio simboliza o tipo de ficheiro e os permisos.

Na saída do comando anterior:

- O primeiro caracter indica de que tipo de ficheiro se trata:

  · Se é un guión '-' significa que se trata dun ficheiro normal.
  · Se é unha letra 'd' significa que se trata dun directorio (*directory*).
  · Se é unha letra 'l' significa que se trata dun enlace (*link*).
  · Se é unha letra 'p' significa unha *pipe* con nome. Esta permite comunicarse entre si a dous programas en execución. Un abre o ficheiro para lectura e o outro o abre para escritura, permitindo transferir os datos entre os programas.
  · Se é unha letra 's' significa un *socket*, este é similar a un *pipe* con nome, pero permite enlaces de rede e bidireccionais.
  · Se é unha letra 'b' significa un dispositivo de bloque (como un disco duro, disquete, CD-ROM, etc).
  · Se é unha letra 'c' significa un dispositivo de caracteres (como un porto paralelo, serie, teclado, etc).

- Os 9 caracteres seguintes simbolizan os permisos do usuario propietario (3 caracteres), os permisos do grupo propietario (3 caracteres) e os permisos do resto de usuarios (3 caracteres). Veñen codificados coas letras r, w e x que se refiren aos permisos de lectura, escritura e

execución. Se en lugar de aparecer devanditas letras aparecen guións significa que se carece de devandito permiso. Por exemplo, se os dez primeiros carácteres son:

```
-rw-r-----
```

Significa que é un ficheiro normal, que o usuario propietario dispón de permisos de lectura e escritura pero non de execución, que o grupo propietario dispón tan só de permiso de lectura e o resto de usuarios non dispón de ningún permiso.

A mesma información podemos vela desde o administrador de ficheiros en modo gráfico se imos ao directorio /home/alumno01 e mostramos as columnas correspondentes.

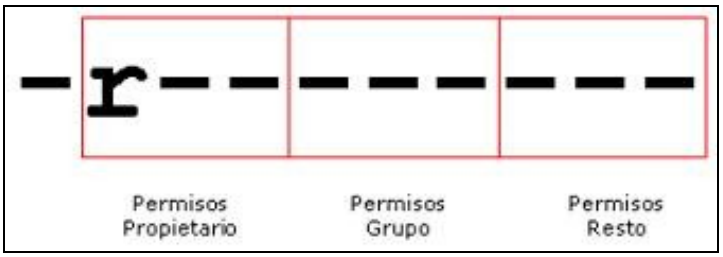
3 Tipos de permisos

A xestión dos permisos que os usuarios e os grupos de usuarios teñen sobre os ficherios e os directorios realízase mediante un sinxelo esquema de tres tipos de permisos que son:

- Permiso de lectura
- Permiso de escritura
- Permiso de execución

O significado destes permisos é distinto se se teñen sobre ficheiros ou sobre directorios, como se verá a continuación.

3.1 Permiso de lectura

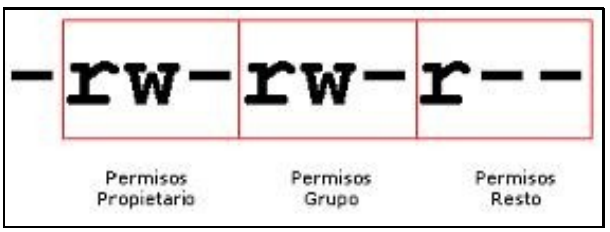


O permiso de lectura simbolízase coa letra 'r' do inglés 'read'.

Cando un usuario ten permiso de lectura **sobre un ficheiro** significa que pode lelo ou visualizalo. Así, se temos permiso de lectura sobre o ficheiro apuntes.txt, significa que podemos ver o seu contido. Se o usuario non ten permiso de lectura, non poderá ver o contido do ficheiro.

Cando un usuario ten permiso de lectura **sobre un directorio**, significa que pode visualizar o contido do directorio, é dicir, pode ver os ficherios e directorios que contén. Se o usuario non ten permiso de lectura sobre o directorio, non poderá ver o que contén.

3.2 Permisos de escritura

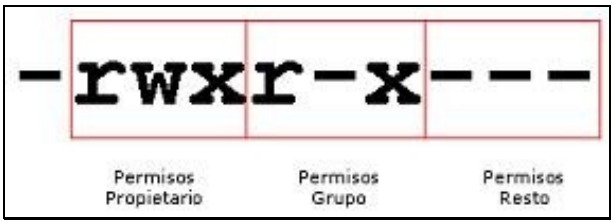


O permiso de escritura simbolízase coa letra 'w' do inglés 'write'.

Cando un usuario ten permiso de escritura **sobre un ficheiro** significa que pode modificar o seu contido, e ata borralo. Tamén lle dá dereito a cambiar os permisos do ficheiro mediante o comando **`chmod`** se é o propietario. Se o usuario non ten permiso de escritura, non poderá modificar o contido do ficheiro.

Cando un usuario ten permiso de escritura **sobre un directorio**, significa que pode modificar o contido do directorio, é dicir, pode crear e eliminar ficheiros e outros directorios dentro del (tamén pode modificar os permisos). Se o usuario non ten permiso de escritura sobre o directorio, non poderá crear nin eliminar ficheiros nin directorios dentro del.

3.3 Permisos de execución



O permiso de execución simbolízase coa letra 'x' do inglés 'eXecute'.

Cando un usuario ten permiso de execución **sobre un ficheiro** significa que pode executalo. Se o usuario non dispón de permiso de execución, non poderá executalo aínda que sexa unha aplicación. Os únicos ficheiros executables son as aplicacións e os ficheiros de comandos (scripts). Se tratamos de executar un ficheiro non executable, dará un erro.

Cando un usuario ten permiso de execución **sobre un directorio**, significa que pode entrar nel, ben sexa co comando 'cd' ou cun explorador de ficheiros como Nautilus ou Konqueror. Se non dispón do permiso de execución significa que non pode ir a devandito directorio.

3.4 Bits de permisos especiais

Existen unhas cantas opcións mais para permisos, que se poden indicar mediante cambios na cadea de permisos. Estes cambios pasaranse "nun primeiro número" con tres bits:

- **O 1º bit define a ID de usuario (SUID):** A opción *set user* ID (SUID) utilízase xunto aos ficheiros executables e lle indica a linux que execute o programa cós permisos do usuario propietario do ficheiro, en lugar de cós permisos do usuario que executa o programa. Por exemplo, se un ficheiro é propiedade de *root* e ten definido o seu bit SUID, o programa executarase con privilexios de *root* e, polo tanto, o poderá ler calquera usuario do ordenador.

  IMPORTANTE: Estes permisos (SUID e SGID) só funcionan con binarios e non con scripts (excepto cos de PERL)
  Os programas SUID están indicados por unha *s* na posición do bit de execución do propietario na cadea de permisos. Vexamos un exemplo:

```
$ ls -l
-rw-r--r-- 2 root    root       0 sep 21 20:32 arquivo
$ chmod 4644 arquivo
$ ls -l
-rwSr--r-- 2 root    root       0 sep 21 20:32 arquivo
```

Outro exemplo interesante neste caso é a aplicación de cambio de contrasinal ***passwd***:

```
$ which passwd
/usr/bin/passwd
$ ls -l /usr/bin/passwd
-rwsr-xr-x  1  root  root  59680  may  17  13:58  /usr/bin/passwd
```

- **O 2º bit define a ID de grupo (SGID):** A opción *set group* ID (SGID) é similar á opción SUID, pero fai que o programa se execute tomando o GID do grupo propietario. Indícase mediante unha *s* na posición do bit de execución do grupo na cadea de permisos. Vexamos un exemplo:

```
$ ls -l
drwxrwxr-x 2 usuario usuario 4096 sep 23 14:26 directorio
$ chmod 2775 directorio
drwxrwsr-x 2 usuario usuario 4096 sep 23 14:26 directorio
# Logo, para eliminalo:
$ chmod -s directorio
$ ls -l
drwxrwxr-x 2 usuario usuario 4096 sep 23 14:26 directorio
```

  Cando se define o bit SGID nun directorio, os novos ficheiros que se creen terán como grupo propietario o mesmo grupo que o directorio, en lugar de tomar o grupo por defecto do usuario actual.

- **O 3º bit define o *Sticky bit* (bit pegañento):** Este bit foi cambiando de significado ao longo da historia de Unix. Nas implementacións modernas de Linux (e as versións mais modernas de Unix), utilízase para evitar que os ficheiros poidan ser borrados por usuarios que non son os seus propietarios.

Cando este bit está presente nun directorio, só poderán borrar os ficheiros do directorio os propietarios destes, o propietario do directorio ou *root*. O *sticky bit* ven indicado como unha letra *t* na posición do bit de execución dos permisos xerais. Vexamos un exemplo:

```
$ ls -l
drwxrwxr-x 2 usuario usuario 4096 sep 23 14:26 directorio
# Activamos o sticky bit:
$ chmod 1775 directorio
$ ls -l
drwxrwxr-t 2 usuario usuario 4096 sep 23 14:26 directorio
# Reseteamos o sticky bit:
$ chmod -t directorio
$ ls -l
drwxrwxr-x 2 usuario usuario 4096 sep 23 14:26 directorio
```

3.5  Importante

Activando o bit adherente nun directorio, evítase que un usuario que non é propietario dun arquivo poida eliminalo do mesmo. Para asegurar o contido dun arquivo en directorios onde todo o mundo pode escribir, como en /tmp, non só se debe desactivar o permiso de escritura do arquivo se non tamén activar o **bit adherente do directorio**. Caso contrario, o arquivo pode ser borrado por un usuario que teña permiso de escritura no directorio, ao crear un novo arquivo có mesmo nome.

```
$ ls -l / | grep tmp
drwxrwxrwt  8  root  root  4096  oct  9  20:17  tmp
```

4 A quen se pode outorgar permisos

En Linux non existe a posibilidade de asignar permisos a usuarios concretos nin a grupos concretos (como en Windows), tan só se poden asignar permisos ao:

- **Usuario propietario** do ficheiro.
- **Grupo propietario** do ficheiro.
- **Resto de usuarios** do sistema (todos menos o usuario propietario e os usuarios que estean no grupo propietario).

Polo tanto, pódense dar permisos de lectura, escritura, execución ou combinación de ambos ao usuario propietario do ficheiro, ao grupo propietario do ficheiro ou ao resto de usuarios do sistema.

Supoñamos que temos un ficheiro apuntes.txt cos permisos rw-r----- pertencente ao grupo profesores e ao usuario pepe. O usuario propietario (pepe) poderá ler e escribir no documento. Os pertencentes ao grupo profesores poderán lelo e o resto non poderá facer nada. Se desexo que outros usuarios teñan algún permiso sobre o ficheiro apuntes.txt, non me quedará máis remedio que incluílos no grupo profesores ou outorgar o permiso ao resto de usuarios pero se fago isto último, absolutamente todos os usuarios do sistema gozarán do permiso. Por iso non se recomenda facer isto útlimo, salvo que iso sexa a nosa intención.

Para poder cambiar permisos sobre un ficheiro, é necesario posuír o permiso de escritura sobre o mesmo. O usuario root pode modificar os permisos de calquera ficheiro xa que ten acceso total sen restricións á administración do sistema.

5 Cambio de permisos: chmod

Para cambiar os permisos dun ficheiro ou un directorio é necesario dispor do permiso de escritura (w) sobre devandito ficheiro ou directorio. Para facelo, utilízase o comando chmod. A sintaxe do comando chmod é a seguinte:

```
chmod [opcións] permisos nome_ficheiro_ou_directorio
```

Os permisos pódense representar de dúas formas, tal e como veremos a continuación.

5.1 Representación dos permisos mediante iniciais

Consiste en incluír as iniciais de a quen vai dirixido o permiso (usuario=u, grupo=g, resto=o (other)), seguido dun signo + se se quere engadir permiso ou un signo - se se quere quitar e seguido do tipo de permiso (lectura=r, escritura=w e execución=x). Vexamos algúns exemplos:

```
# Dar permiso de escritura ao usuario propietario sobre o ficheiro 'exame.txt'
$ chmod u+w exame.txt

# Quitar permiso de escritura ao resto de usuarios sobre o ficheiro 'exame.txt'
$ chmod o-w exame.txt

# Dar permiso de execución ao grupo propietario sobre o ficheiro '/usr/bin/writer'
$ chmod g+x /usr/bin/writer

# Dar permiso de lectura ao grupo propietario sobre o ficheiro 'exame.txt'
$ chmod g+r exame.txt

# Agrega permisos de lectura a todos os usuarios
$ chmod +r exame.txt

# Elimina o permiso de execución a todos os usuarios
$ chmod ?x exame.txt

# Pódense pór varios permisos xuntos separados por comas
$ chmod u+w,g-r,u-r exame.txt

# Establece os permisos de lectura e escritura ao dono
# e elimina todos os permisos aos demais usuarios
$ chmod u=rw,go= exame.txt

# Pódense pór varios usuarios xuntos
$ chmod ug+w exame.txt
```

5.2 Representación dos permisos mediante código numérico

Código	Binario	Permisos efectivos
0	0 0 0	- - -
1	0 0 1	- - x
2	0 1 0	- w -
3	0 1 1	- w x
4	1 0 0	r - -
5	1 0 1	r - x
6	1 1 0	r w -
7	1 1 1	r w x

A segunda forma de representar os permisos é **mediante un código numérico** que traducido a binario representaría a activación ou desactivación dos permisos.

O código numérico está composto por tres cifras entre 0 e 7. A primeira delas representa os permisos do usuario propietario, a segunda os do grupo propietario e a terceira os do resto de usuarios. En binario, as combinacións representan o tipo de permiso. O bit máis á dereita (menos significativo) refírese ao permiso de execución (1=activar e 0=desactivar). O bit central refírese ao permiso de escritura e o bit máis á esquerda refírese ao permiso de lectura. A táboa da esquerda mostra as 8 combinacións posibles.

Así, se desexamos outorgar só permiso de lectura, o código a utilizar é o 4. Se desexamos outorgar só permiso de lectura e execución, o código é o 5. Se desexamos outorgar só permiso de lectura e escritura, o código é o 6. Se desexamos outorgar todos os permisos, o código é o 7. Se desexamos quitar todos os permisos, o código é o 0.

Vexamos algúns exemplos:

```
# Dar todos os permisos ao usuario e ningún nin ao grupo nin ao resto
$ chmod 700 exame.txt

# Acceso total ao dono e lectura e escritura aos demais:
$ chmod 766 exame.txt

# Dar ao usuario e ao grupo permisos de lectura e execución e ningún ao resto
$ chmod 550 exame.txt

# Dar todos os permisos ao usuario e lectura e execución ao grupo e ao resto
$ chmod 755 apuntes.txt

# Dar todos os permisos ao usuario e de lectura ao resto, sobre todos os ficheiros
$ chmod 744 *

# Lectura e escritura ao dono, escritura e execución ao grupo e lectura e execución ao resto:
$ chmod 635 file.txt

# Cambiar permisos a todos os ficheiros incluíndo subcarpetas
$ chmod -R 744 *
```

Existe a posibilidade de cambiar os permisos utilizando o explorador de ficheiros. Para iso tan só hai que seleccionar os ficheiros ou directorios e facendo clic sobre a selección co botón dereito do rato -> Propiedades, apareceranos a ventá de propiedades. Facendo clic no separador Permisos poderemos establecer os permisos dunha forma sinxela e facendo clic en Permisos 'avanzados' dunha forma avanzada.

6 Máscaras

Os permisos orixinais por defecto dun ficheiro son 666 e dun directorio son 777. Devanditos permisos por defecto poden modificarse co comando umask (*user mask*), que permite definir a **máscara de permisos**.

O valor a asignar como ***umask*** atopábase antes configurado no arquivo */etc/profile* pero hoxe configúrase en */etc/login.defs*. O que indica é o número a ser "restado" do total de permisos 777 (por exemplo, unha *umask* de 022 implica permisos por defecto de 755).

Así, o permiso por defecto será o resultado de restar do permiso orixinal, o valor da máscara. Se desexamos que os ficheiros se cren con permisos 644 (o máis habitual), poremos máscara 022 xa que 666-022=644. No caso dos directorios, o permiso efectivo será 755 xa que 777-022=755. Pero, ¿que pasaría se pomos máscara 033? pois según o visto para os ficheiros teriamos 666-033=633 e para os directorios 777-033=744, sen embargo se vemos os permisos de ficheiros e directorios creados con esa máscara(033) soamente concordan os directorios. Entón, ¿que pasa? Pois que realmente a resta é factible cos directorios pero non cos ficheiros, realmente non se fai unha operación resta senón 2 operacións na seguinte orde: primeiro cámbiase os valores da máscara(033) a binario(000 011 011) e néganse os valores(111 100 100) -isto é os ceros toman valor un e viceversa- e segundo faise a operación AND cos valores 666 en binario(110 110 110) pertencentes aos ficheiros ou 777(111 111 111)pertencentes aos directorios.

Vexamos un exemplo:

- **Ficheiros:**

```
permisos orixinais ficheiro 666 --> pasar a binario  110 110 110
umask 033 --> binario 000 011 011 -->       negar    111 100 100
operación AND                                       ----------------
                                                      110 100 100

Así, os permisos actuais son: 644
```

- **Directorios:**

```
permisos orixinais directorio 777 -->  pasar a binario 111 111 111
umask 033 --> binario 000 011 011 -->  negar           111 100 100
operación AND                                         ----------------
                                                        111 100 100

Así, os permisos actuais son: 744
```

Outro exemplo:

```
110 110 110 (666 ficheiro) ---> 110 110 110      111111111 (777 directorio) ---> 111 111 111
000010111 (umask = 027)  ---> 111 101 000        000010111 (umask = 027)    ---> 111 101 000
                           --------------- AND                                 --------------- AND
                              110 100 000 (640)                                    111 101 000 (750)
```

Tamén podemos averiguar os permisos efectivos dun ficheiro ou directorio pensando que cada bit de *umask* a 1 "borra" o permiso correspondente. Por exemplo, se umask vale *027* os permisos dos ficheiros serán:

```
110110110 (666)
000010111 (umask = 027)
---------
110100000 (640)
```

É dicir, onde hai un 1 no *umask* o permiso queda a 0.

Para ver o valor de *umask* chega con teclear:

```
$ umask
0002
$ umask -S
u=rwx,g=rwx,o=rx
```

Así obteremos unha saída de 4 díxitos onde o primeiro deles é sempre "0" (ata o momento este díxito non se utiliza, ainda que se plantexa relacionalo cós bits de seguridade **SUID** ou **GUID** nun futuro).

Cada usuario, no ficheiro ***/home/usuario/.profile***, ten a súa máscara. Así, pódese fixar a máscara por defecto para todos os usuarios no ficheiro */etc/login.defs* (anteriormente en */etc/profile*) ou para cada usuario no ficheiro */home/usuario/.profile* (este último valor prevalece sobre o xeral).

Vexamos un exemplo de *umask*:

```
# Miramos o valor por defecto de umask
$ umask
0022
# Cambiamos umask dende "usuario" a 0002
$ umask 0002
# Comprobamos cambio
$ umask
0002
# Creamos un directorio
$ mkdir directorio
$ ls -l
drwxrwxr-x 2 usuario usuario 4096 sep 23 14:15 directorio
# Volvemos a cambiar umask ao valor por defecto
$ umask 0022
# Creamos un novo directorio (chamado "directorio2")
# coa máscara por defecto
$ mkdir directorio2
$ ls -l
drwxrwxr-x 2 usuario usuario 4096 sep 23 14:15 directorio
drwxr-xr-x 2 usuario usuario 4096 sep 23 14:16 directorio2
```

A modificacion con *umask* da máscara por defecto non afecta aos ficheiros e directorios existentes senón só aos novos que cre ese usuario a partir dese momento.

7 Cambio de propietarios: chown

Para poder cambiar o usuario propietario dun ficheiro ou directorio utilízase o comando chown (change owner). Para iso hai que ser root. A sintaxe simplificada do comando é:

```
chown novo_usuario nome_ficheiro_directorio
```

Vexamos uns exemplos:

```
# Poñer a xan como propietario de arquivo1 e arquivo2
```

```
$ chown xan arquivo1 arquivo2

# Poñer a maria como propietaria do directorio /var/datos e de todos
# os seus subdirectorios e arquivos
$ chown -R maria /var/datos
```

8 chgrp

O comando [chgrp] permite cambiar o grupo propietario dun ficheiro.

```
# Poñer a proxecto1 como propietario de arquivo1 e arquivo2
$ chgrp proxecto1 arquivo1 arquivo2

# Poñer a maria como usuaria propietaria e poñer a proxecto1 como grupo propietario
# do directorio /var/datos e de todos os seus subdirectorios e arquivos
$ chown -R maria:proxecto1 /var/datos
```

--Arribi 12:12 6 oct 2009 (BST)