permisos

cada archivo tiene 3 grupos de permisos, el del autor (owner), el del grupo (group) y el del resto.
en un listado (`ls -la`) los permisos se muestran en el mismo orden (primero owner, después grupo, después resto) representado por 3 letras:

- para ficheros:
  - r = permiso de lectura
  - w = permiso de escritura, borrado, renombrado
  - x = permiso de ejecución.
    - si en lugar de una **x** aparece una **s** significa que tiene el SUID activo, se explica más adelante.
- para directorios:
  - r = lectura del contenido de la carpeta
  - w = escribir, renombrar, borrar
  - x = acceso
    - si en lugar de una **x** aparece una **S** significa que tiene el SETDIG activo, se explica más adelante.

si la letra aparece, el permiso está activo, si aparece un guión en el lugar el permiso no está activo

SUID

este permiso activo indica que el fichero se ejecuta con el usuario propietario y no con el usuario que lo ejecuta

```
$ chmod u+s <file>
```

(solo con sentido en /usr/bin/passwd, no debería haber más en el sistema)

Si al listar el fichero aparece la S en mayúscula, es que no tenía activo el permiso de ejecución (x)

GUID

este permiso activo indica que los ficheros que se graben/creen asumirán automáticamente el grupo de la carpeta que lo contiene y no el original del mismo.

```
$ chmod g+s <file>
```

Si al listar el fichero aparece la S en mayúscula (en el apartado del grupo), es que no tenía activo el permiso de ejecución (x)

sticky

añade un «candado» al fichero o directorio, de manera que solo el OWNER del fichero, aunque existan otros permisos en «grupo» o «otros», puede borrarlo

cambiar los permisos

se cambian con el comnado `chmod`:

- `$ chmod {u,g,o}{+,-}{r,w,x,s,t}`

refinamiento permisos (ACL Extended)

- http://rm-rf.es/acl-access-control-list-en-sistemas-de-ficheros-gnulinux/
- https://juncotic.com/acl-access-control-lists-y-los-permisos-en-gnu-linux/
- `getfacl`: consultar ACLs sobre fichero
- `setfacl`
  - **-m**: establecer permisos
  - **-x**: eliminar permisos
  - **-b**: quitar todos

setfacl

- modificar:
  - `[d:][u:uid][:perm]`
    - si **u:** está vacío, se aplica al propietario del fichero
    - los **perm** pueden estar en octal o simbólico
  - `[d:]g:[uid][:perm]`
    - si **g:** está vacío, se aplica al grupo propietario del fichero
  - `[d:]m[:perm]`
    - máscara
  - `[d:]o[:perm]`
    - other
  - sobre los directorios, para que se apliquen por defecto en el contenido, hay que usar **d:**

exportar/importar

- con **getfacl** se pueden exportar los permisos a fichero (pipe >)
- modificar con editor
- importar los permisos con **setfacl -M**

ln

- crear enlaces, duros o simbólicos
- `$ ln -s /path/to/dest name_soft_link`
- para ver la trayectoria real, no la modificada, después de seguir un link, pwd -p. pwd muestra la «lógica»

umask

actua negando ciertos bits (**r** y **w** concretamente, no tiene efecto sobre **x**)

```
Umask doesn’t work as you might expect: it doesn’t represent the permissions a file
gets, but the bits of the full access value that are turned off (masked). With a
umask value of 000, a file gets 666 permissions, equivalent to rw-rw-rw-. Any other
value of umask will turn bits off (if the bit value is 1, make it 0):

in the case of umask 442, we negate r--r---w- bits, resulting in -w--w-r--.
for umask 553, we negate r-xr-x-wx bits, obtaining -w--w-r-- once again.
As you can see, x bit is turned off by default, so masking it has no effect.
```