Permisos Especiales: setuid, setgid, sticky (o pegajoso).

-> Linux ofrece 3 tipos de permisos 'bit especiales', y que: aplicados en ficheros y directorios permitirá que estos respondan de forma diferente.
-> Deben utilizarse con cuidado para evitar potenciales riesgos de seguridad.

==> setuid → 4xxx

→ El usuario que adquiere el fichero obtiene la personalidad del propietario del mismo.
→ **SIN** permisos de ejecución ('**x**') el 'setuid' será una **S** (mayúscula); **CON** permisos de ejecución ('**x**') será una **s** (minúscula).
→ Un ejemplo clásico es el comando '**su**' del que es propietario el usuario '**root**'.

```
[root@CentOS-7 carlos]# ll /usr/bin/su
-rwsr-xr-x. 1 root root 32064 ago  6 02:24 /usr/bin/su → 4755
```

→ Si eliminamos el '**setuid**' como 'root':

```
[root@CentOS-7 carlos]# chmod u-s /usr/bin/su
[root@CentOS-7 carlos]# ls -al /usr/bin/su
-rwxr-xr-x. 1 root root 32064 ago  6 02:24 /usr/bin/su
```

→ Si ahora intentamos ser '**su**' desde un usuario sin privilegios:

```
[carlos@CentOS-7 ~]$ su
Contraseña:
su: Fallo de autenticación
```

→ Nos impide adoptar la personalidad de '**root**'
→ Ahora debemos dejar las cosas como estaban inicialmente, …

```
[carlos@CentOS-7 ~]$ chmod 4755 /usr/bin/su
chmod: cambiando los permisos de «/usr/bin/su»: Operación no permitida
```

('Houston' tenemos un problema, … → Debemos restaurar el Stma. con parametros en el arranque del kernel (GRUB/LILO, ...) en modo 'single'.
****** NO PROBAR NUNCA ESTE COMANDO EN PRODUCCIÓN ******

→ Es necesario pasar al arranque del kernel en GRUB y al final → **init=/bin/sh**
→ **Nos mostrará el shell 'sh'. Desde aquí ejecutar lo que sigue:**
→ El Stma. Se montará solo en '**lectura (r)**'.
Para montarlo en modo '**rw**':
→ **sh-4.2# mount -o remount,rw /**
→ Restablecemos permisos Originales:
→ **sh-4.2# chmod 4755 /usr/bin/su**
→ Si estaba Activo '**SeLinux**' en modo '**enforcing**' debemos hacer también:
→ **sh-4.2# touch /.autorelabel**
→ Ya podremos iniciar nuestro Sistema de forma natural.
→ Por último podemos encontrar los ficheros '**setuid**' de la forma:
→ [root@CentOS-7 carlos]# find /usr/bin -perm **4755**

```
/usr/bin/Xorg
/usr/bin/at
/usr/bin/chage
/usr/bin/crontab
/usr/bin/fusermount
/usr/bin/gpasswd
/usr/bin/mount
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/pkexec
```

/usr/bin/su
/usr/bin/umount

==> setgid → 2xxx

→ Similar a 'setuid' pero de aplicación para grupos. También es asignable a directorios.
→ Con 'setgid' activado los usuarios NO PROPIETARIOS del grupo adquieren los mismos privilegios que los AUTENTICOS PROPIETARIOS del grupo.
→ **SIN** permisos de ejecución ('**x**') el 'setuid' será una **S** (mayúscula); **CON** permisos de ejecución ('**x**') será una **s** (minúscula).
→ Ejemplo para directorios:
→ Como usuario 'carlos':

```
[carlos@CentOS-7 ~]$ mkdir directorio_x
[carlos@CentOS-7 ~]$ chmod g+s directorio_x/
[carlos@CentOS-7 ~]$ ls -la directorio_x/
total 8
drwxrwsr-x.  2 carlos carlos    6 nov 27 10:22 . → 2775
drwx--x---+ 34 carlos carlos 4096 nov 27 10:22 ..
```

→ Como usuario 'root':

```
[root@CentOS-7 directorio_x]# touch fichero_x
[root@CentOS-7 directorio_x]# ls -al
total 8
drwxrwsr-x.  2 carlos carlos   22 nov 27 10:25 .
drwx--x---+ 34 carlos carlos 4096 nov 27 10:22 ..
-rw-r--r--.  1 root   carlos    0 nov 27 10:25 fichero_x
```

(Al tener 'setgid' activado el propietario de 'fichero_x' NO es 'root', sino 'carlos').
→ Para desasignarlo podemos hacerlo tambien en octal:

```
[carlos@CentOS-7 ~]$ chmod 0775 directorio_x/
[carlos@CentOS-7 ~]$ ls -al directorio_x/
total 8
drwxrwsr-x.  2 carlos carlos   22 nov 27 10:25 . → 0775
drwx--x---+ 34 carlos carlos 4096 nov 27 10:22 ..
-rw-r--r--.  1 root   carlos    0 nov 27 10:25 fichero_x
```

→ Podemos los ficheros '**setgid**' de la forma:

```
[root@CentOS-7 carlos]# find / -perm 2555
/usr/bin/wall
```

==> sticky (pegajoso) → 1xxx

→ Este bit suele asignarse en directorios a los que todos los usuarios tienen acceso, y **permite evitar que un usuario pueda borrar ficheros/directorios de otro usuario dentro de ese directorio, ya que todos tienen permiso de escritura**.
→ Históricamente, el sticky bit se utilizaba en ficheros ejecutables. Cuando se asignaba, le indicaba al sistema operativo (SO) que mantuviera el programa en swap para ejecuciones posteriores (incluso de otros usuarios). Desde entonces el rendimiento de las tecnologías de almacenamiento persistente han mejorado mucho y este uso ha quedado obsoleto.
→ Este bit se asigna siempre como ejemplo clásico en **/tmp** y **/var/tmp**.
→ **SIN** permisos de ejecución ('**x**') el 'sticky' será una **T** (mayúscula); **CON** permisos de ejecución ('**x**') será una **t** (minúscula).
→ Ejemplo para el directorio **/tmp**.

```
[root@CentOS-7 carlos]# ls -al /tmp
total 8
```

drwxrwxrwt. 13 root root 300 nov 27 10:44 . → **1**777
dr-xr-xr-x. 18 root root 4096 nov 22 14:12 ..

→ Para asignar: → **chmod 1777 /tmp**, o tambien: → **chmod o+t /tmp**.
→ Para desasignar: → **chmod 0777 /tmp**, o tambien: → **chmod o-t /tmp**.