> Linux除了我们常见的读(r)写(w)执行(x)权限外，还有三个较为特殊的权限SetUID、SetGID与Sticky BIT。

1. SetUID与SetGID

通常对于可执行程序能够设定`SetUID`权限，且命令执行者需要对该程序拥有`x`权限。

设定`SetUID`权限后，命令执行者在执行程序时获得程序拥有者的身份，且身份改变仅在执行过程中有效。

例：修改用户`passwd`

```
$ ll /etc/passwd /etc/shadow /usr/bin/passwd
-rw-r--r-- 1 root root    3360  3月 30 15:40 /etc/passwd
-rw-r----- 1 root shadow  1808  3月 30 15:40 /etc/shadow
-rwsr-xr-x 1 root root   63960  2月  7  2020 /usr/bin/passwd
```

- 可以看到普通用户并没有对`/etc/passwd`与`/etc/shadow`具有执行修改权限。但在实际使用需求上是有修改自己password的需求。
- 所以在用户的`/usr/bin/passwd`程序中，设置了`SetUID`即拥有者权限中的`s`。
- 可以看到`Others`用户具有可执行权限`x`，而程序拥有者设定了`SetUID`，同时程序拥有者为`root`。
- 所以`Others`用户在执行`/usr/bin/passwd`时，会临时获得`root`权限，从而能修改密码。

> SetGID与SetUID原理基本相同。

2. Sticky BIT

通常对于目录能够设定`Sticky BIT`。用于避免非文件拥有者误操作文件。

例：对于`/tmp`目录所有用户都具有操作权限，但任何人创建的文件能被其他人任意修改与删除可能会带来错误。

```
$ ll -d /tmp
drwxrwxrwt 19 root root 4096  4月 15 09:39 /tmp
```

可以通过对`/tmp`目录设定`Sticky BIT`即权限最后的`t`，来保证其中的文件仅能够被文件拥有者修改。

3. 权限修改

在常规的 *777* 类型的模式下设置权限，在最前面增加一位设置 `SetUID` 、 `SetGID` 与 `Sticky BIT` 即可，分别对应4、2、1。或通过增加权限的方式。

```
# 设置具有SetUID但其他用户只读的权限
chmod 4744 filename
# 设置setuid权限
chmod u+s filename
# 设置setgid权限
chmod g+s filename
# 设置stick bit权限，针对目录
chmod o+t dirname
```