Linux的权限机制

- 前言：

先前我们用了ls -l命令时，会发现当前文件夹内的所有文件和文件夹都会以列表的形式列举出来。那么，我们在每个文件前面看到的那些英文前缀都是什么呢？这一节，我们探讨的就是这个玩意，以加深对Linux的权限的理解。

- 正文：

例如，我们使用ls -ld (文件夹) 命令时，会出现drwxr-x---. 11 root root 4096类似的结果。

在这里，我们需要把这行莫名其妙的英文前缀分成四部分：①d②rwx③r-x④---，它们分别代表着①是文件则为“-”，否则为文件夹“d”；②属主(u)对该文件夹有Read，Write，eXcute的权利，即读写和执行的权力；③属组(g)对该文件有读和执行的权力，但不能写；④其他用户(o)对该文件没有任何权限，碰不得它

那么，属主和属组又是什么呢？我们看后半部分，有个“root root”，在这里，前面的root即是该文件的属主，而后面的root则是属组。通俗地将，该文件夹承认自己的主人是root，同时也承认一个root组里的其他成员也是自己的主人。

简单了解那一行结果的意思后，我们回来说说读写和执行都是干啥：①对于文件，读指是否可以访问其内容，写是指是否可以保存在VIM里对其的编辑，执行是指若该文件是个脚本文件则是否能运行；②对于文件夹，读指是否可以访问其下内容，写是指是否可以在该文件夹下做创建删除改名等操作，执行则是指能不能进入该文件夹。

那么，我们怎么才能手动更改文件和文件夹的属主属组和权限呢？这里用到了chown和chmod命令：①chown -R 属主:属组 文件夹名(路径)，chown配合-R可以一次改变属主和属组；②chmod (更改对象的英文字母)=(rwx) 文件名(路径)。例如，chmod g=rx /home/linuxuser，表示给linuxuser这个用户文件夹的在属组位上加上读和执行能力，这样它的属组只能读和执行该文件夹，而不能改写。另外，使用a来代替u/g/o可以一次性的改三个权限位，如chmod a=rwx /home/linuxuser表示三个权限位一起拥有读写和执行的能力。其实权限位还可以做加减法，如u+w、u+w-r这种命令形式，不妨自己试试吧。

还有，我们规定r=4，w=2，x=1，-=0，然后用加法来计算，从而更改权限位。例如r+w+x=7，以此类推，chmod 754 /home/linuxuser的意思就是给该文件夹的权限改成drwxr-xr--了，熟悉数字后改权限会更快。

在Linux中，每当创建一个新文件或文件夹时，就算不设定权限，也会自带一个默认权限，这又是为什么呢？不妨输入命令umask，然后Linux就会输出0022，但022才是umask的值。我们需要知道，Linux中有最大权限的概念，touch file——最大权限666，mkdir——最大权限777。然后做一个减法，如777-022=755，所以创建一个文件夹的默认权限就是755，即drwxr-xr-x；不过对于一个文件就不是这么简单的减法了，要分为两种情况：①减法后都是偶数，就是最终结果；②任意一位奇数都要+1，这才是最终结果。我们可以在/etc/profile中更改umask，然后输入命令source /etc/profile使修改生效。

最后，我们要讲一个特殊权限位，也是比较麻烦的东西。

在Linux中，还有rwSr-Sr-T这样的文件，这三个SST又是什么玩意呢？它们分别对应SUID、SGID、SBIT。我们做例子看区别吧。

```
[root@localhost ~]# ls
[root@localhost ~]# su - linuxuser
Last login: Mon Sep  6 09:30:56 EDT 2021 on tty1
-bash-4.2$ ls -ld /etc/shadow
----------. 1 root root 852 Sep  2 04:23 /etc/shadow
-bash-4.2$ logout
[root@localhost ~]# su - linuxuser
Last login: Mon Sep  6 09:34:07 EDT 2021 on tty1
-bash-4.2$ whereis passwd
passwd: /usr/bin/passwd /etc/passwd /usr/share/man/man1/passwd.1.gz /usr/share
-bash-4.2$ ls -ld /usr/bin/passwd
-rwsr-xr-x. 1 root root 27856 Mar 31  2020 /usr/bin/passwd
-bash-4.2$ logout
[root@localhost ~]# cd /
bin/   dev/   home/  lib64/ mnt/   proc/  run/   srv/   tmp/   var/
boot/  etc/   lib/   media/ opt/   root/  sbin/  sys/   usr/
[root@localhost ~]# cd /tmp/
[root@localhost tmp]# mkdir test_sgid
[root@localhost tmp]# su - linuxuser
Last login: Mon Sep  6 09:34:42 EDT 2021 on tty1
-bash-4.2$ cd /tmp/test_sgid/
-bash-4.2$ mkdir test
mkdir: cannot create directory ‘test’: Permission denied
-bash-4.2$ ls -ld /tmp/test_sgid/
drwxr-xr-x. 2 root root 6 Sep  6 09:36 /tmp/test_sgid/
-bash-4.2$ logout
[root@localhost tmp]# chmod  g+s test_sgid/
[root@localhost tmp]# ls -ld
drwxrwxrwt. 10 root root 4096 Sep  6 09:42 .
[root@localhost tmp]# ls -ld /tmp/test_sgid/
```

如图，我们先前知道密码存放在/etc/shadow中，查看文件属性发现全是横杠，我超，那为什么用户用passwd指令可以改自己密码？我们whereis passwd查看passwd的路径，在查看其权限，发现在属主位有个s。其实，普通用户自然不能访问shadow，但它执行passwd命令时，由于passwd的SUID标识，使得passwd的属主(root)的权力继承到了用户，所以用户才能访问，但这当然不代表该普通用户就可以为所欲为了哈。顺便我还给test_sgid的g权限位加上了s。

对于SGID的实验，我们在/tmp/里建了一个test_sgid文件夹来玩玩。我想在test-sgid里建一个文件夹，结果不行。

```
[root@localhost tmp]# su - linuxuser
Last login: Mon Sep  6 09:36:08 EDT 2021 on tty1
-bash-4.2$ cd /tmp/test_sgid/
-bash-4.2$ mkdir test2
mkdir: cannot create directory ‘test2’: Permission denied
-bash-4.2$ logout
[root@localhost tmp]# chmod o+w /tmp/test_sgid/
[root@localhost tmp]# su - linuxuser
Last login: Mon Sep  6 09:43:13 EDT 2021 on tty1
-bash-4.2$ cd /tmp/test_sgid/
-bash-4.2$ mkdir test2
-bash-4.2$ ls
test2
-bash-4.2$ ls -ld /tmp/test_sgid/
drwxr-srwx. 3 root root 19 Sep  6 09:50 /tmp/test_sgid/
-bash-4.2$ ls -ld /tmp/test_sgid/test2/
drwxrwsr-x. 2 linuxuser root 6 Sep  6 09:50 /tmp/test_sgid/test2/
```

这是自然的，我在o权限位加上了w后，总算可以了，我就新建了一个test2文件夹。欸，我们会看到test2的属组居然是root而不是linuxuser(不像之前创建文件后都是属主属组名字一样的)？这就是sgid的魅力，它会让其下的文件(test2)全部继承它(test_sgid)的属组(root)。当然咯，属主还是linuxuser，因为这是linuxuser用户创造的嘛。

```
-bash-4.2$ ls -ld /tmp/test_sgid/
drwxr-srwx. 3 root root 19 Sep  6 09:50 /tmp/test_sgid/
-bash-4.2$ chmod g-s+x /tmp/test_sgid/
chmod: changing permissions of '/tmp/test_sgid/': Operation not permitted
-bash-4.2$ logout
[root@localhost tmp]# chmod g-s+x /tmp/test_sgid/
[root@localhost tmp]# ls -ld /tmp/test_sgid/
drwxr-xrwx. 3 root root 19 Sep  6 09:50 /tmp/test_sgid/
[root@localhost tmp]# su - linuxuser
Last login: Mon Sep  6 09:50:17 EDT 2021 on tty1
-bash-4.2$ cd /tmp/test_sgid/
-bash-4.2$ mkdir test
-bash-4.2$ ls
test  test2
-bash-4.2$ ls -ld test
drwxrwxr-x. 2 linuxuser linuxuser 6 Sep  6 09:55 test
-bash-4.2$ ls -l
total 0
drwxrwxr-x. 2 linuxuser linuxuser 6 Sep  6 09:55 test
drwxrwsr-x. 2 linuxuser root      6 Sep  6 09:50 test2
```

可以看到，即使o权限位上有rwx，我也不能用chmod改权限，别忘了这些权限是给出其他用户对文件夹查看删除重命名打开的能力，而不是改权限的能力哦。随后我把test_sgid/的sgid权限去掉，又新建了一个test文件夹，这下ls -l后，test和test2的差异一眼看出，也印证了前面的内容。

```
-bash-4.2$ whereis chmod
chmod: /usr/bin/chmod /usr/share/man/man1/chmod.1.gz /usr/share/man/man1p/chmo
/man2/chmod.2.gz /usr/share/man/man3p/chmod.3p.gz
-bash-4.2$ ls -ld /usr/bin/chmod
-rwxr-xr-x. 1 root root 58592 Aug 20  2019 /usr/bin/chmod
-bash-4.2$ logout
[root@localhost tmp]# chmod u+s /usr/bin/chmod
[root@localhost tmp]# ls -ld /usr/bin/chmod
-rwsr-xr-x. 1 root root 58592 Aug 20  2019 /usr/bin/chmod
[root@localhost tmp]# su - linuxuser
Last login: Mon Sep  6 09:55:13 EDT 2021 on tty1
-bash-4.2$ cd /tmp/test_sgid/
-bash-4.2$ ls
test  test2
-bash-4.2$ ls -l
total 0
drwxrwxr-x. 2 linuxuser linuxuser 6 Sep  6 09:55 test
drwxrwsr-x. 2 linuxuser root      6 Sep  6 09:50 test2
-bash-4.2$ cd ..
-bash-4.2$ ls -ld
drwxrwxrwt. 10 root root 4096 Sep  6 10:11 .
-bash-4.2$ pwd
/tmp
-bash-4.2$ cd test_sgid/
-bash-4.2$ ls
test  test2
-bash-4.2$ ls -ld
drwxr-xrwx. 4 root root 31 Sep  6 09:55 .
-bash-4.2$ chmod u+s /tmp/test_sgid/
-bash-4.2$ ls -ld /tmp/test_sgid/
drwsr-xrwx. 4 root root 31 Sep  6 09:55 /tmp/test_sgid/
-bash-4.2$ mkdir test3
-bash-4.2$ ls
test  test2  test3
-bash-4.2$ ls -ld test3
drwxrwxr-x. 2 linuxuser linuxuser 6 Sep  6 10:14 test3
-bash-4.2$
```

还没完，我切换到了root，直接给chmod文件的u权限位加了个suid权限，这样一来我就可以随意的更改文件夹的权限，这就是我前面说的，当启用这条指令时，我目前的用户可以说是"狗仗人势"了。如图，我去修改了test_sgid/的权限，给它的u权限位加了个s权限，看看属主能否像属组一样被继承。但很遗憾，新建的test3的属主和属组都是linuxuser，看来属主不能继承的呢。

```
-bash-4.2$ chmod o-w /tmp/test_sgid/
-bash-4.2$ ls -ld /tmp/test_sgid/
drwsr-xr-x. 5 root root 44 Sep  6 10:14 /tmp/test_sgid/
-bash-4.2$ mkdir test4
mkdir: cannot create directory ‘test4’: Permission denied
-bash-4.2$
```

最后，我还玩了一会儿。我把test_sgid/的o权限位的写权力去掉，果不其然不能在里面新建文件了，然后就对chmod进行chmod去掉它的suid的特殊权限，下线了🤣🤣🤣。

- 后记：

这一节还是有点难的，要多动手才行。