Linux权限

1.Linux权限的概念

Linux下有两种用户：超级用户（root）、普通用户。

超级用户：可以再linux系统下做任何事情，不受限制

普通用户：在linux下做有限的事情。

超级用户的命令**提示符**是“#”，普通用户的命令**提示符**是“$”。

命令：su [用户名]

功能：切换用户。

例如，要从root用户切换到普通用户user，则使用 su user。 要从普通用户user切换到root用户则使用 su

root（root可以省略），此时系统会提示输入root用户的口令

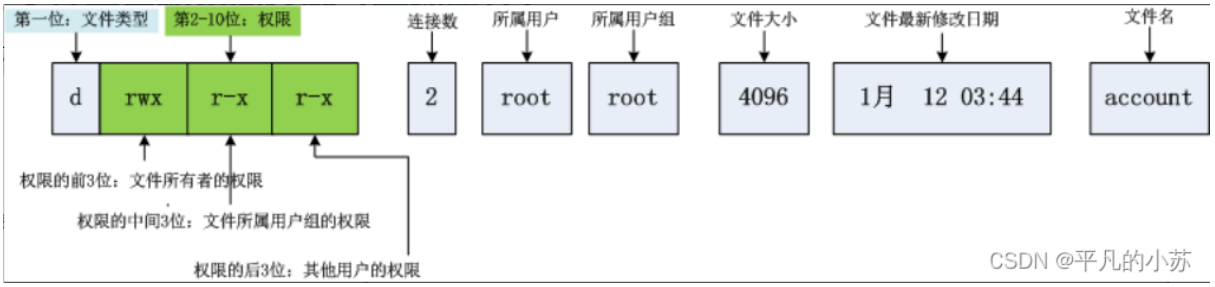
2.Linux权限的管理

2.1文件访问者的分类（人）

- 文件和文件目录的所有者：u—User
- 文件和文件目录的所有者所在的组的用户：g—Group
- 其它用户：o—Others

> 如果我们想要对一条命令进行短暂的提权，sudo command。
>
> 如果是adduser新添加的用户，没有添加到root的信任用户名单里面
>
> 那么是没有办法执行sudo的，系统不信任你。

2.2文件类型和访问权限（事物属性）



> **注：**普通用户可以是拥有者、所属组、和其他人，超级用户也可以是拥有者、所属组、其他人。
>
> 但是超级用户不受权限的约束。为什么有了拥有者和其他人，还要有所属组。
>
> **例子：**
>
> 假设我们在公司做一个项目，然后分成两个小组，需要两个小组竞争，看谁做的好就给哪个组
>
> 提薪，那么我们两个组就属于竞争关系，那么我只想把我的文件给我的小组成员观看，不想
>
> 被其他组的成员看到，就需要有所属组，只有在所属组的成员才能看我的文件，不在所属组的
>
> 都属于其他人，那么别的小组就看不到我们创建的文件了。

文件类型：

> d：文件夹（**主要**）
>
> -：普通文件（**主要**）
>
> l：软链接（类似Windows的快捷方式）
>
> b：块设备文件（例如硬盘、光驱等）
>
> p：管道文件（用来进行通信）
>
> c：字符设备文件（例如屏幕等串口设备）
>
> s：套接口文件

基本权限

i.读（r/4）：Read对文件而言，具有读取文件内容的权限；对目录来说，具有浏览该目录信息的权限

ii.写（w/2）：Write对文件而言，具有修改文件内容的权限；对目录来说具有删除移动目录内文件的权限

iii.执行（x/1）：execute对文件而言，具有执行文件的权限；对目录来说，具有进入目录的权限

iv.“—”表示不具有该项权限

2.3文件权限值得表示方法

字符表示法：

Linux表示	说明	Linux表示	说明
r--	只读	-w-	仅可写
--x	仅可执行	rw-	可读可写
-wx	可写和可执行	r-x	可读可执行
rwx	可读可写可执行	---	无权限

8进制数字表示方法：

权限符号（读写执行）	八进制	二进制
r	4	100
w	2	010
x	1	001
rw	6	110
rx	5	101
wx	3	011
rwx	7	111
---	0	000

2.4文件访问权限得相关方法

2.4.1、chmod

- 功能：设置文件的访问权限
- 格式: chmod [参数] 权限 文件名

常用选项：

`R` → 递归修改目录文件的权限

说明：只有文件的拥有者和root才可以改变文件的权限

chmod 命令权限值得格式

① 用户表示符+/-=权限字符

+:向权限范围增加权限代号所表示的权限

-:向权限范围取消权限代号所表示的权限

=:向权限范围赋予权限代号所表示的权限

用户符号：

`u`：拥有者

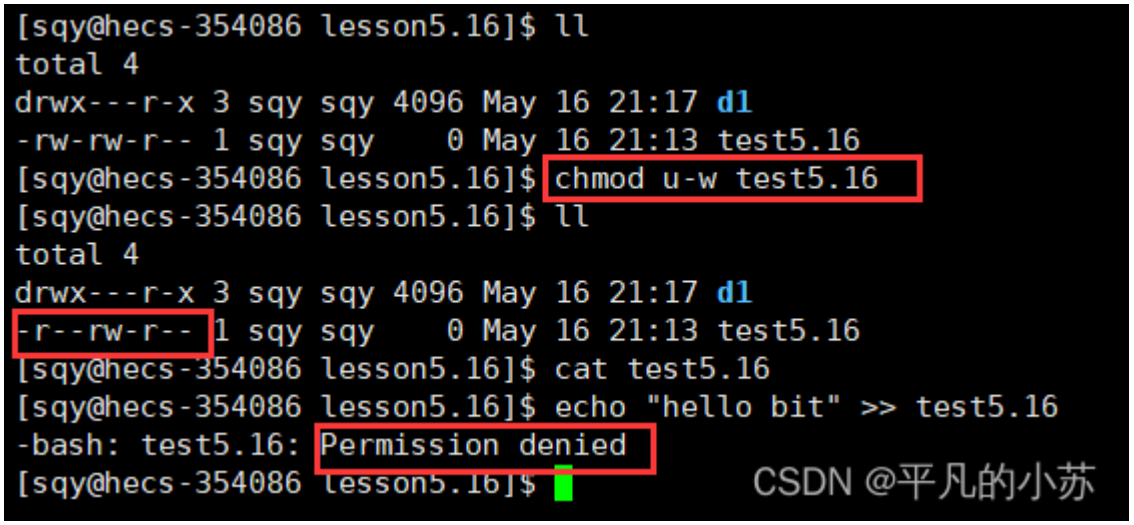
`g`：拥有者同组用

`o`：其它用户

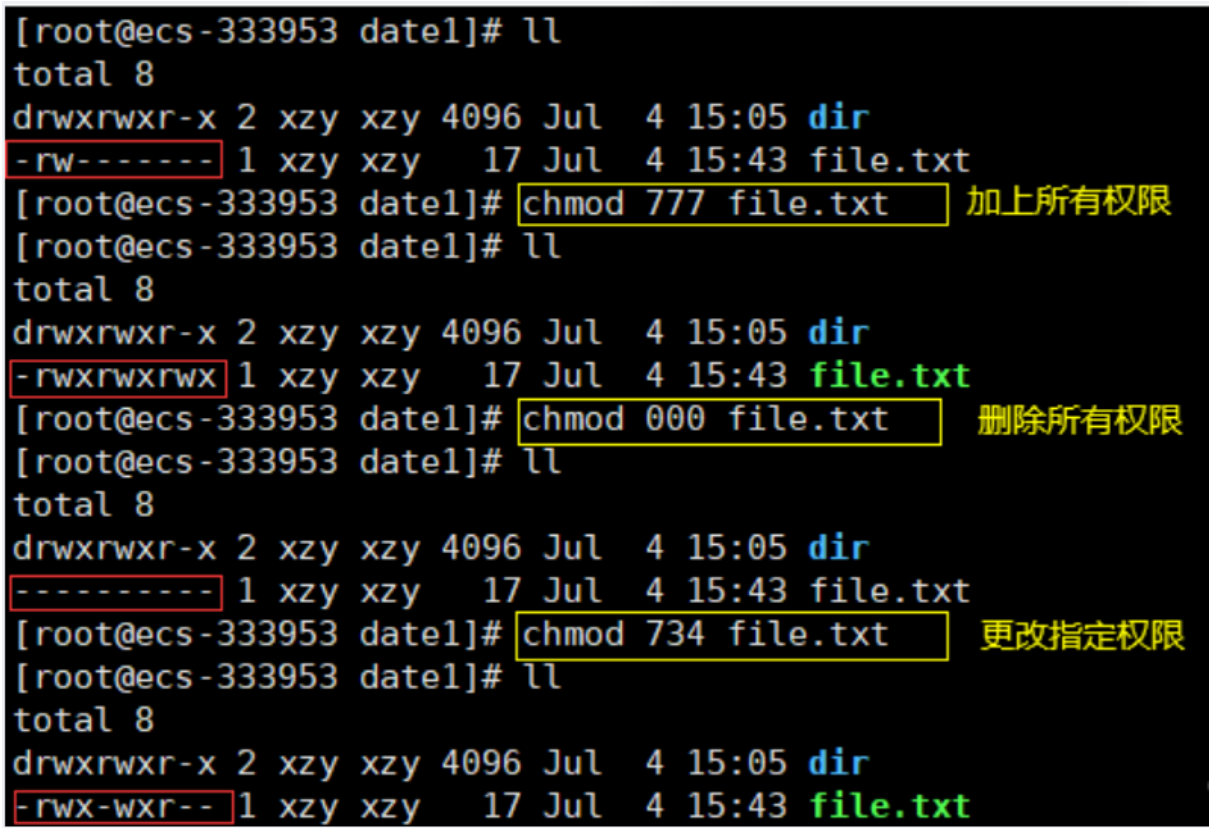
`a`：所有用户

⚠ **注意**：root不受任何权限约束，可以为所欲为

示例1：



示例2：使用三位八进制数字



2.4.2、chown

功能：修改文件的拥有者

格式：chown [参数] 用户名 文件名

示例：

chown user1 f1

chown -R user1 filegroup1

> **注意**：我们在改拥有者的时候，普通用户是给不了的，因为我们给别人文件需要经过别人的同意，所以Linux不允许普通用户给人文件的权限，我们可以使用超级用户强制去修改文件的拥有者

2.4.3、chgrp

功能：修改文件或目录的所属组

格式：chgrp [参数] 用户组名 文件名

常用选项：-R 递归修改文件或目录的所属组

> 与chown一样都需要超级用户修改

2.4.4、umask

功能：

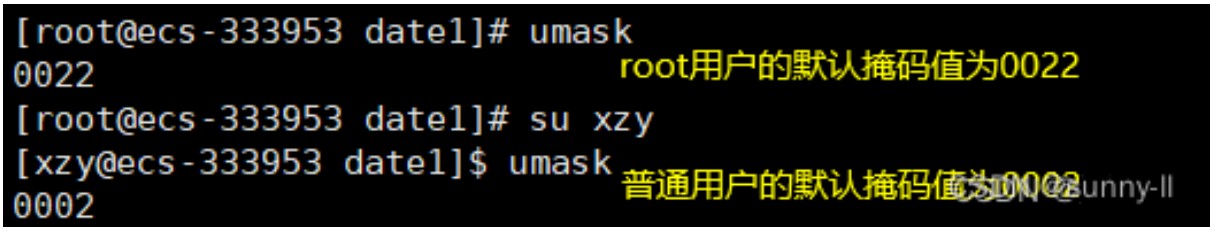
查看或修改文件掩码

新建文件夹**默认权限**=`0666`

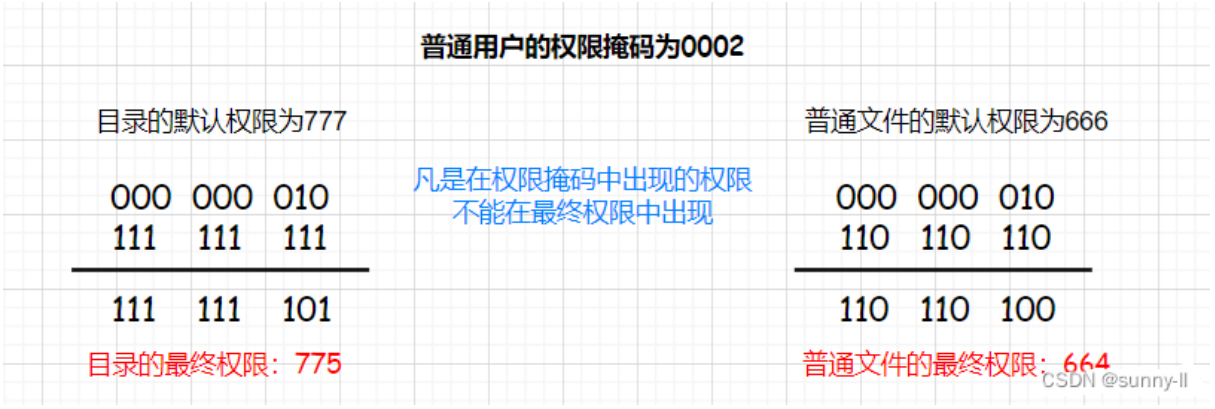
新建目录**默认权限**=`0777`

但实际上你所创建的文件和目录，看到的权限往往不是上面这个值。原因就是创建文件或目录的时候还要受到

umask的影响。假设默认权限是mask，则实际创建的出来的文件权限是: mask & ~umask



注意：凡是在权限掩码中出现的权限都不应该在最终权限出现。实际中我们只关注权限掩码的后三位。

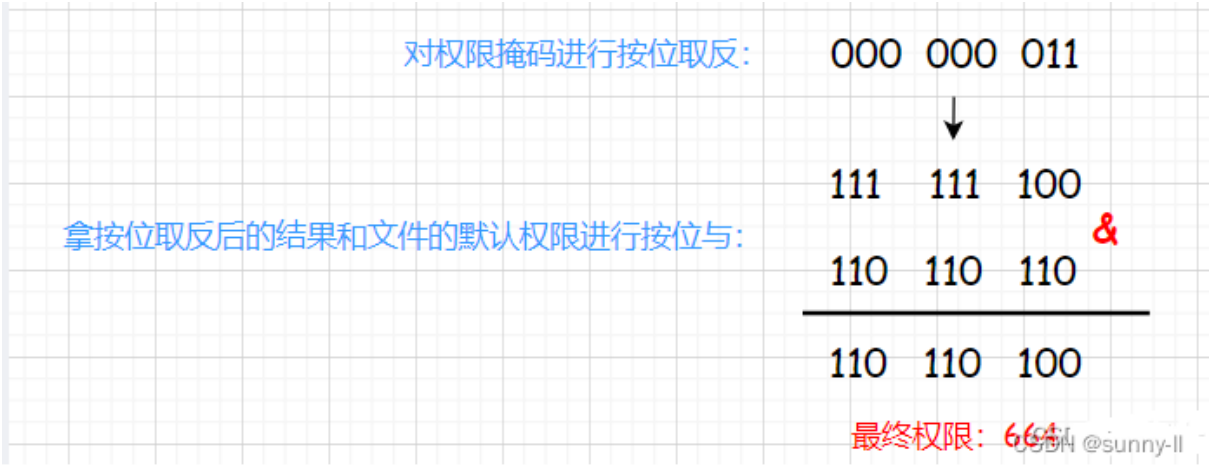


此图很好的解释了为什么先前我们创建的目录权限为775，而普通文件的权限为664。

普通文件的默认权限是从666开始的，目录文件的默认权限是从777开始的，但是最终权限 ！= 默认权限。原因就在于存在权限掩码umask。这也就导致Linux最终权限 = 默认权限“去掉”umask中存在的权限。换句话说就是凡是在权限掩码中出现的权限都不应在最终权限中出现。

- **那么最终权限到底如何计算呢？**

首先，对**默认权限掩码按位取反**，接着拿按位取反后的结果与文件的默认权限进行按位与的操作 ，得到的就是最终权限。



综上，**最终权限 = 默认权限 &（~umask）**。

3.目录的权限

> 了解了普通文件的rwx可读可写可执行，现在来看看目录的rwx的含义：
>
> - **r**可读权限: 如果目录没有可读权限, 则无法用ls等命令查看目录中的文件内容.
> - **w**可写权限: 如果目录没有可写权限, 则无法在目录中创建文件, 也无法在目录中删除文件.
> - **x**可执行权限: 如果目录没有可执行权限, 则无法cd到目录中.

4.粘滞位

> 在Linux中，可以存在一些目录，拥有者和所属组是root，其它人允许以other的身份在该目录下进行文件的创建，读取，删除，修改等(公共信息传递区)。如下的一个名为tmp的目录
> 该目录的拥有者和所属组均属于root，且other其它人的权限都是没有限制的，也就是说任何人都可以在里头读写文件。

> **总结**：当一个目录被设置为"粘滞位"(用**chmod +t**),则该目录下的文件只能由
>
> - 超级管理员root删除
> - 该目录的所有者删除
> - 该文件的所有者删除

Untitled