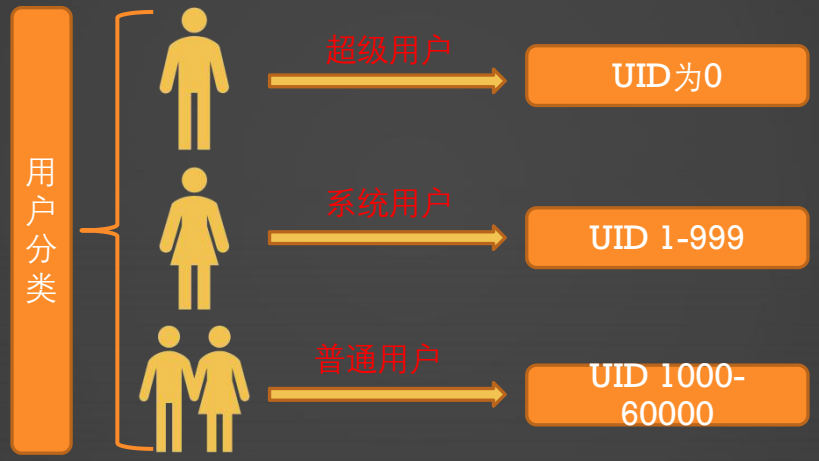
用户管理及权限

《linux操作系统与安全》

LINUX用户分类



/etc/passwd格式


```
[root@localhost zhangsan]# cat /etc/passwd | more
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
alsr:x:1000:1000:lsr:/home/lsr:/bin/bash
zhangsan:x:1001:1001::/home/zhangsan:/bin/bash
user1:x:1002:1004::/home/user1:/bin/bash
```

在Linux系统中，用户分为三类：
超级用户root，系统用户和普通用户。

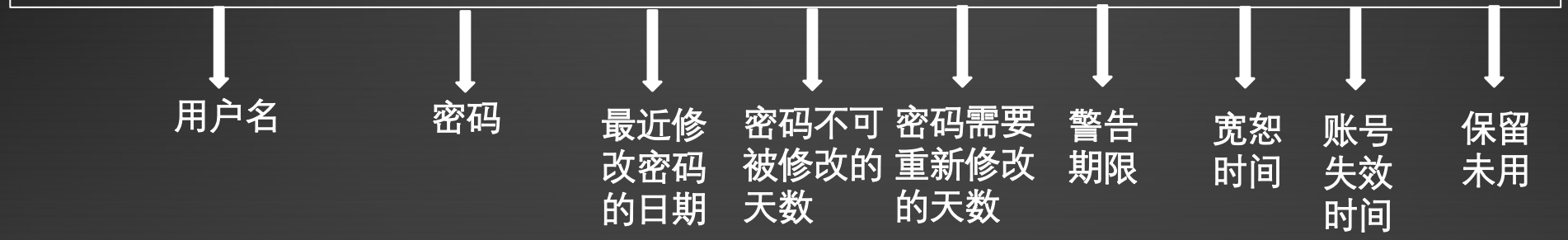
```
[root@localhost zhangsan]# cat /etc/login.defs |grep UID*
UID_MIN                  1000
UID_MAX                  60000
```

/etc/shadow格式

[root@localhost zhangsan]# ll /etc/passwd /etc/shadow
-rw-r--r-- 1 root root 2602 3月 20 18:15 /etc/passwd
---------- 1 root root 1523 3月 20 18:15 /etc/shadow

:6sGoudfWXvN8lbFva$bHXlhRdqIu1ZXRJNlan3xE4PQbFmzqzWXiGgtZT61JVviw57BKvYxnVLd9zUZxCNjJ1hx/st9uVba8H1O/vHM/

zhangsan: 加密口令 : 19424 : 0 : 99999 : 7 : : :



密码字段解析

6sGoudfWXvN8lbFva$bHXlhRdqIu1ZXRJNlan3xE4PQbFmzqzWXiGgtZT61JVviw57BKvYxnVLd9zUZxCNjJ1hx/st9uVba8H1O/vHM/

idsalt$encrypted

id的值	加密算法
1	MD5
2	Blowfish
5	SHA-256
6	SHA-512

加盐哈希算法

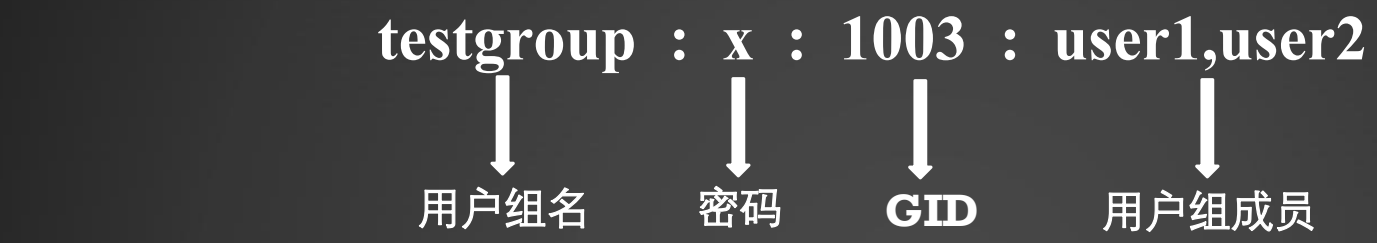
第二个$符号后面的salt表示，加密算法中用到的salt，也就是加盐哈希算法中的盐。salt最长为16个字符。在密码学中，了解到不加盐的哈希加密算法容易遭到字典攻击、暴力攻击和查表法等暴力破解，为了阻止这样的攻击，采用加盐哈希的方法进行加密。

hash(“明文口令” +“盐”) =密文

在给密码hash时加一个随机的字符串，然后再进行hash，这个随机的字符串就称为“盐”。这就是加盐hash。红帽企业版9.1中采用的就是这样的加盐hash加密算法。

第三个$符号后面就是真正的加密后生成字符串，其长度和加密的算法有关，如果采用md5算法加密，其长度为22个字符，如果采用SHA-256算法加密，其长度为43个字符，如果采用SHA-512算法加密，其长度为86个字符。

/etc/group格式



用户组成员，一个用户是可以加入多个用户组的。举例来说，如果我想要让lsr也加入root这个组，那么在第一行的最后面加上,lsr，注意不要有空格，使成为 root:x:0:root,lsr 就可以了。

初始组和有效组

Linux操作系统中，多个用户是用组来划分，一个用户可以同时加入多个组，那么这时候，哪一个用户组起作用呢?

当新建一个用户zhangsan时，在/etc/passwd和/etc/group两个文件中会自动生成一行，如下所示:

```
[root@localhost zhangsan]# cat /etc/passwd /etc/group |grep zhangsan
zhangsan:x:1001:1001::/home/zhangsan:/bin/bash
zhangsan:x:1001:
```

第一行中可以看出zhangsan用户属于1001用户组，从第二行可以看到1001组也就是zhangsan组。也就是说zhangsan用户属于zhangsan组。这里的/etc/passwd文件中查看到的组就称为zhangsan用户的初始组。

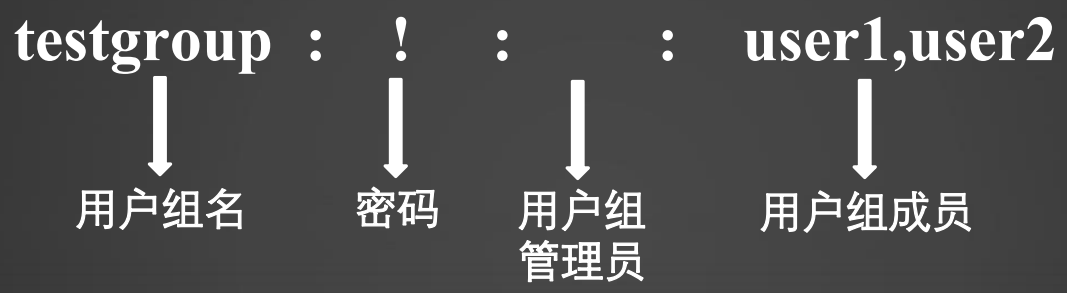
改变有效组

第一组为有效组

```
[root@localhost zhangsan]# usermod -G root zhangsan
[root@localhost zhangsan]# su zhangsan
[zhangsan@localhost ~]$ groups
zhangsan root
[zhangsan@localhost ~]$ touch testfile1
[zhangsan@localhost ~]$ ll testfile1
-rw-rw-r-- 1 zhangsan zhangsan 0  4月   3 12:26 testfile1
[zhangsan@localhost ~]$ newgrp root
[zhangsan@localhost ~]$ touch testfile2
[zhangsan@localhost ~]$ ll testfile2
-rw-r--r-- 1 zhangsan root 0  4月   3 12:26 testfile2
```

Newgrp
改变有效组

/etc/gshadow格式



相关管理命令

用户及用户组相关命令

[root@localhost ~]# useradd user3

[root@localhost ~]# grep user3 /etc/passwd /etc/shadow /etc/group /etc/gshadow
/etc/passwd:user3:x:1004:1007::/home/user3:/bin/bash
/etc/shadow:user3:!!:19426:0:99999:7:::
/etc/group:user3:x:1007:
/etc/gshadow:user3:!::

```
[root@localhost zhangsan]# ls -al /home/user1
总用量 16
drwx------  4 user1 user1 113  3月   8 17:16 .
drwxr-xr-x. 8 root  root   85  3月  11 21:36 ..
-rw-------  1 user1 user1 322  3月   8 18:17 .bash_history
-rw-r--r--  1 user1 user1  18  8月   8  2022 .bash_logout
-rw-r--r--  1 user1 user1 141  8月   8  2022 .bash_profile
-rw-r--r--  1 user1 user1 492  8月   8  2022 .bashrc
drwx------  2 user1 user1   6  3月   8 17:07 .cache
drwxr-xr-x  4 user1 user1  39  2月  18 10:37 .mozilla
```

useradd

- 举例，新建一个用户ID为601，用户名为tom，描述信息为Tom，属于root组，shell类型为/bin/sh，用户主目录为/home/tom。
- [root@localhost zhangsan]# useradd -u 601 -r tom -c "Tom" -g root -s /bin/sh -d /home/tom
- [root@localhost zhangsan]# grep tom /etc/passwd
- tom:x:601:0:Tom:/home/tom:/bin/sh

passwd

[zhangsan@localhost ~]$ passwd
更改用户 zhangsan 的密码 。
当前的密码:
新的密码:
重新输入新的密码:
passwd: 所有的身份验证令牌已经成功更新。

[root@localhost ~]# passwd zhangsan
更改用户 zhangsan 的密码 。
新的密码:
重新输入新的密码:
passwd: 所有的身份验证令牌已经成功更新。

usermod

【例题】使用useradd命令修改用户的信息。帐号的有效期改为2025年12月30日。

[root@bogon ~]# usermod -e "2025-12-25" zhangsan

[root@bogon ~]# passwd -l zhangsan
锁定用户 zhangsan 的密码 。
passwd: 操作成功
[root@bogon ~]# grep zhangsan /etc/shadow
zhangsan:!!6WPE8f6sXKVNiYS6u$6qyh8sdrseLXLdq/rr7I0ifKi.bhiz6j0vdJDxDP.nUSO5H142yCcfEZxAaZgQHICEQY./pi8Xu.Etg34mEE7/:19424:0:99999:7::20447:
[root@bogon ~]# passwd -u zhangsan
解锁用户 zhangsan 的密码。
passwd: 操作成功
[root@bogon ~]# grep zhangsan /etc/shadow
zhangsan:6WPE8f6sXKVNiYS6u$6qyh8sdrseLXLdq/rr7I0ifKi.bhiz6j0vdJDxDP.nUSO5H142yCcfEZxAaZgQHICEQY./pi8Xu.Etg34mEE7/:19424:0:99999:7::20447:

userdel

[root@bogon ~]# userdel user1

//删除用户user2，包括用户user2的主目录一起删除
[root@bogon ~]# userdel -r user2

groupadd

【例题】创建一个新的GID为578的组，组名为group2。

```
[root@localhost ~]# groupadd -g 578 group2
[root@localhost ~]# grep group2 /etc/group /etc/gshadow
/etc/group:group2:x:578:
/etc/gshadow:group2:!::
```

groupmod groupdel

//修改组group2为group3

[root@localhost ~]# groupmod group2 -n group3

[root@localhost ~]# grep group3 /etc/group /etc/gshadow

/etc/group:**group3**:x:578:

/etc/gshadow:**group3**:!::

如果任何一个组的用户在系统中使用，并且要删除的组为该用户的主分组的话，则不能删除该分组，必须首先删除该用户后才能删除该组。

[root@localhost ~]# groupdel user3

groupdel：不能移除用户“user3”的主组

[root@localhost ~]# groupdel group3

id

id命令的主要作用是输出指定用户的用户和用户组信息，语法格式为“id用户名”。

[root@localhost ~]# id zhangsan
用户id=1001(zhangsan) 组id=1001(zhangsan) 组=1001(zhangsan)

[root@localhost ~]# id 0
用户id=0(root) 组id=0(root) 组=0(root)

su

```
[root@localhost ~]# su - zhangsan
上一次登录:　六 3月 11 17:22:23 CST 2023 pts/1 上
[zhangsan@localhost ~]$ pwd
/home/zhangsan
[zhangsan@localhost ~]$ su - root
密码:
上一次登录:　六 3月 11 17:22:43 CST 2023 pts/1 上
```

sudo

- 使用su命令切换用户很简单，但是有一个很严重的问题就是必须要知道超级用户的口令。这样很不安全。为了避免这个问题，可以使用sudo命令。
- 当用户执行 sudo 时，系统会主动的去寻找 /etc/sudoers 文件，判断该用户是否有执行 sudo 的权限；若用户具有可执行 sudo 的权限后，便让用户输入用户自己的密码来确认；若密码输入成功，便开始进行 sudo 后续接的指令。

sudoers

在默认的情况下，普通用户是不能使用sudo权限的，如果想要使用sudo，必须在文件/etc/sudoers中进行权限设置。因为/etc/sudoers文件的语法格式很特殊，所以最好不要用编辑器直接编辑，linux中提供了visudo命令用来编辑/etc/sudoers文件。

用户名 登录的主机=（可以变换的身份） 可以执行的命令

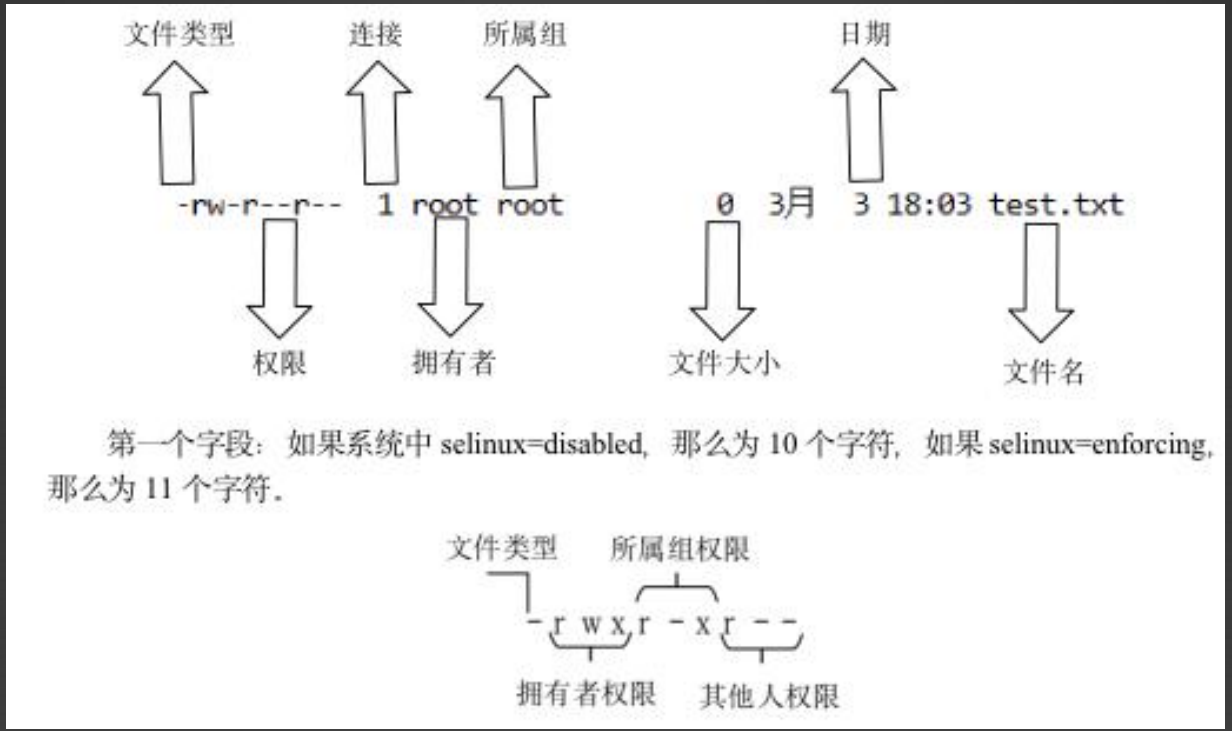
使用**sudo**的用户　从哪台主机登录　可以变换成哪种身份　可以执行系统中哪些命令

zhangsan ALL=(ALL)ALL

zhangsan 192.168.14.136=(root) /sbin/useradd

第二部分 权限管理

文件的基本权限



第一个字段：如果系统中 selinux=disabled，那么为 10 个字符，如果 selinux=enforcing，那么为 11 个字符。

文件权限的作用

权限对于文件的含义

（1）对文件有读（r）权限，代表可以读取文件中的数据。

如果把权限对应到命令上，那么一旦对文件有读（r）权限，就可以对文件执行cat、more、less、head、tail等文件查看命令。

（2）对文件有写（w）权限，代表可以修改文件中的数据。

如果把权限对应到命令上，那么一旦对文件有写（w）权限，就可以对文件执行vim、echo等修改文件数据的命令。

注意：对文件有写权限，是不能删除文件本身的，只能修改文件中的数据。如果要想删除文件，则需要对文件的上级目录拥有写权限。

（3）对文件有执行（x）权限，代表文件拥有了执行权限，可以运行。

在Linux中，只要文件有执行（x）权限，这个文件就是执行文件了。但是这个文件到底能不能正确执行，不仅需要执行（x）权限，还要看文件中的代码是不是正确的语言代码。可执行权限对于文件来说是最高权限。

权限对于目录的含义

（1）对目录有读（r）权限，代表可以查看目录下的内容，也就是可以查看目录下有哪些子文件和子目录。

如果把权限对应到命令上，那么一旦对目录拥有了读（r）权限，就可以在目录下执行ls命令，查看目录下的内容了。

（2）对目录有写（w）权限，代表可以修改目录下的数据，也就是可以在目录中新建、删除、复制、剪切子文件或子目录。

如果把权限对应到命令上，那么一旦对目录拥有了写（w）权限，就可以在目录下执行touch、rm、cp、mv命令。对目录来说，写（w）权限是最高权限。

（3）目录是不能运行的，那么对目录拥有执行（x）权限，代表可以进入目录。

如果把权限对应到命令上，那么一旦对目录拥有了执行（x）权限，就可以对目录执行cd命令，进入目录。

举例说明

【例题】如下所示，在Linux系统中，用户zhang是否能进入teuser1目录内呢?

[lsr@bogon ~]$ ls -l

dr-xr--r-- 2 lsr lsr 6 3月 6 16:10 teuser1

目录teuser1的拥有者是lsr，用户zhang属于其他用户，所以其权限为r--，对于目录来说，只拥有r权限，是不能进入的，如果想要进入该目录，zhang用户必须拥有r-x的权限。

【例题】在Linux系统中，有lsr用户，其主目录为/home/lsr，目录lsr的权限如下所示，

[lsr@bogon teuser1]$ ls -l /home

drwx------. 17 lsr lsr 4096 3月 6 16:10 lsr

在/home/lsr目录内有一个文件，其权限如下所示：

[root@bogon lsr]# ls -l

-rwx------ 1 root root 0 3月 6 16:25 teuser.txt

那么用户lsr能够删除该文件么?

一个用户想要进入一个目录，必须拥有r-x权限

Lsr用户拥有W权限

该文件属于root用户

但是lsr用户拥有其上层目录的W权限

chmod

- chmod命令改变文件权限的方法有两种，一种是字符法，一种是数字法。
- 字符法

在字符法中，使用u表示拥有者，g表示同组用户，o来表示其他用户，a表示所有用户。如下表所示：

【例题】在Linux系统，用户想要设置teuser.txt文件的权限为rwxrw-r--，应该如何设置呢？

```
[root@bogon lsr]# chmod u=rwx,g=rw-,o=r-- teuser.txt
[root@bogon lsr]# ls -l teuser.txt
-rwxrw-r-- 1 root root 0  3月  6 16:25 teuser.txt
```

设置对象			
u g o a	+ - =	r w x	

数字法改变权限

第二种设置权限的方法是用数字法进行设置，在前面学习到，Linux的基本全权限包括9位，每三个表示一组，在数字法中，需要记住如下的对应关系，其中，每个数字表示对应位置的权限。

r	4
w	2
x	1
-	0

举例

【例题】在Linux系统中，新建一个文件hello.c，并修改其权限为rwxrwxr--，

```
[root@bogon lsr]# touch hello.c
[root@bogon lsr]# ls -l hello.c
-rw-r--r-- 1 root root 0  3月  6 20:01 hello.c
[root@bogon lsr]# chmod 774 hello.c
[root@bogon lsr]# ls -l hello.c
-rwxrwxr-- 1 root root 0  3月  6 20:01 hello.c
```

umask默认权限

在Linux系统中，新建一个文件或者目录的时候，如果不指定权限，默认的权限是什么呢？这个默认的权限和umask的值有关系。如下所示：

以超级用户root登录系统时，查看umask的值如下：

[root@bogon ~]# umask
0022

设置umask默认权限的目的是为了保证文件的安全性，如果没有特别的需要，一般情况下，不要改变该值。

举例

如果新建一个文件，这个新建的文件的默认权限的计算方法是用默认的最大权限666减去umask的值的后三位022，也就是新建的这个文件的权限等于644，也就是rw-r--r--。

[root@bogon ~]# touch teuser1.txt

[root@bogon ~]# ls -l teuser1.txt

-rw-r--r-- 1 root root 0 3月 7 19:40 teuser1.txt

如果新建一个目录，其默认权限的计算方法是用默认的最大权限777减去umask的值的后三位022，也就是这个目录的权限等于755。

[root@bogon ~]# mkdir temp

[root@bogon ~]# ls -l

drwxr-xr-x 2 root root 6 3月 7 19:39 temp

umask计算

对于新建文件来说，默认最大的权限是666，也就是默认的最大权限是不分配可执行权限的。

对于新建目录来说，默认最大的权限是777，也就所有的权限都开放。

文件和目录的隐藏权限

Linux系统中，除了基本的r、w、x权限之外，还有一些隐藏的权限，这些隐藏的权限只有root用户可以设置，普通用户是不能设置的。

隐藏属性	含义
i	如果对文件设置i属性，那么不允许对文件进行删除、改名，也不能添加和修改数据； 如果对目录设置i属性，那么只能修改目录下文件中的数据，但不允许建立和删除文件；
a	如果对目录设置a属性，那么只允许在目录中建立和修改文件，但是不允许删除文件；
u	设置此属性的文件或目录，在删除时，其内容会被保存，以保证后期能够恢复，常用来防止意外删除文件或目录。
s	和u相反，删除文件或目录时，会被彻底删除（直接从硬盘上删除，然后用0填充所占用的区域），不可恢复。

chattr

【例题】使用chattr命令给文件添加i隐藏权限，并使用lsattr命令来查看。

```
[root@bogon ~]# touch file1
[root@bogon ~]# chattr +i file1
[root@bogon ~]# rm file1
rm：是否删除普通空文件 'file1'？y
rm: 无法删除 'file1': 不允许的操作
```

可以使用lsattr命令来查看文件或目录的隐藏权限，具体用法如下所示：

```
[root@bogon ~]# lsattr file1
----i---------------- file1
```

举例

```
[root@bogon dteuser]# chattr +a a.txt
[root@bogon dteuser]# lsattr
-----a--------------- ./a.txt
//向该文件添加数据，可以成功执行
[root@bogon dteuser]# echo hello >> a.txt
[root@bogon dteuser]# cat a.txt
hello
//修改该文件数据，可以看出不能执行该操作
[root@bogon dteuser]# echo hello > a.txt
-bash: a.txt: 不允许的操作
```

文件的i、a隐藏权限都非常有用，如果一个文件或者目录很重要，需要加强安全保护，就可以考虑给其设置这样的保护措施，从而实现数据安全。

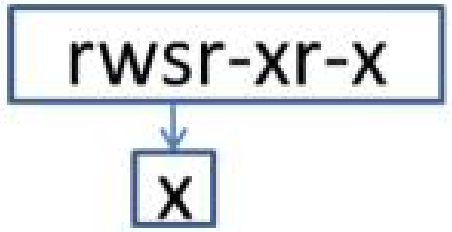
特殊权限SUID SGID SBIT

SUID

在 Linux 系统中，除了基本权限和隐藏权限以外，还有一些很特殊的权限，例如 /usr/bin/passwd 的权限：

```
[root@bogon ~]# ls -l /usr/bin/passwd
-rwsr-xr-x. 1 root root 32648   8 月  10   2021 /usr/bin/passwd
```

SUID



如图所示：本来应该是文件拥有者占据的 x 权限位置，现在被一个字符 s 所替代，那么这个字符表示的含义是什么呢？在 Linux 系统中，这样的权限是一种特殊的权限，称为 SUID。在给一个文件设置 SUID 的时候，需要注意以下几个地方：

(1) 只有二进制可执行程序文件才能设置 SUID；
(2) 运行程序的用户对于该程序要有 x 的可执行权限；
(3) s 权限仅在程序运行的过程中有效；
(4) 运行程序的用户将临时具有该程序拥有者的权限。

用户zhangsan没有/etc/shadow文件的权限，为什么可以修改自己的密码呢？

下面举个实际的例子来进行分析 SUID 的作用，首先使用 ls 命令查看文件/etc/shadow 的权限：

```
[root@bogon ~]# ls -l /etc/shadow
---------- 1 root root 1312  3 月  6 16:12 /etc/shadow
```

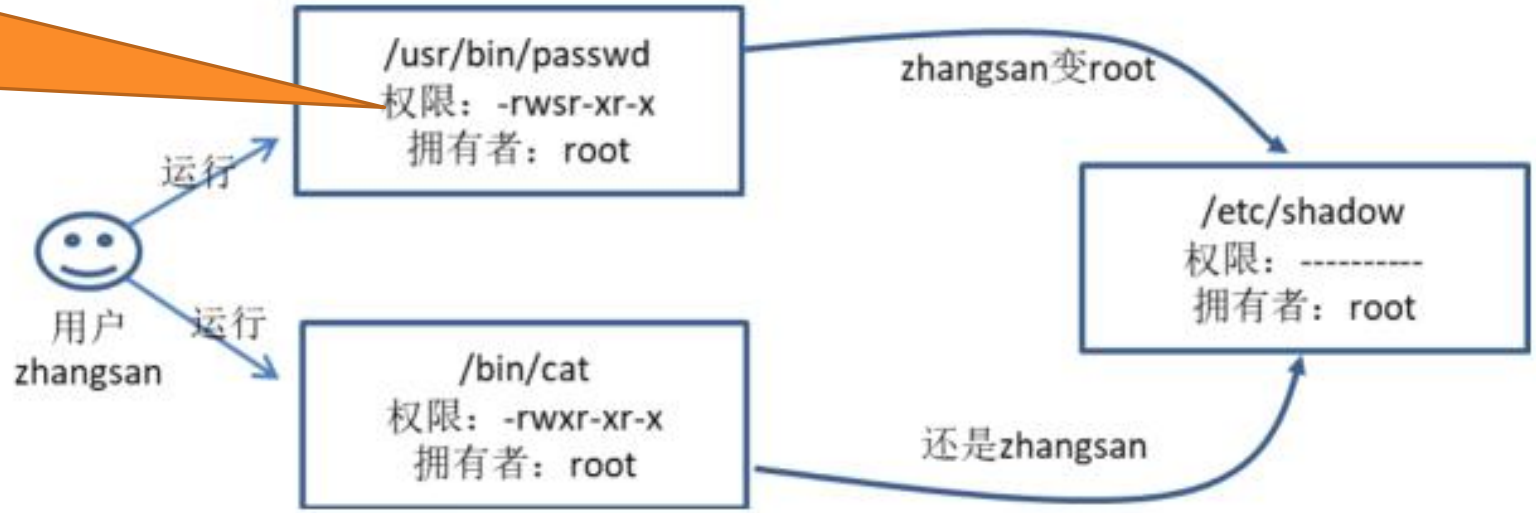
对于文件/etc/shadow 来说，除了 root 用户以外，其他任何用户都不能修改该文件，但是在 Linux 系统内，比如一个普通用户 zhangsan，的确是可以通过 passwd 命令来修改自己的口令的。比如：

```
[zhangsan@bogon root]$ passwd
更改用户 zhangsan 的密码 。
当前的密码:
新的密码:
重新输入新的密码:
passwd: 所有的身份验证令牌已经成功更新。
```

在用户管理的章节中，应该可以知道，修改后的口令其实就是存储在/etc/shadow 文件中了。那么看到这里，很多人应该就很疑惑了，按照道理来说，普通用户 zhangsan 是没有权限对文件/etc/shadow 文件进行写入操作的，但是在这里为什么 zhangsan 这个用户可以通过 passwd 命令来修改口令文件呢?

设置了SUID权限的程序，当zhangsan用户执行该程序时，就临时变身成为该程序拥有者root，所以就可以修改自己的密码了。

这就是 SUID 的作用。因为命令文件/usr/bin/passwd 有了 rwsr-xr-x 这样的权限，就可以实现一些特殊的功能，也就是说当一个普通用户 zhangsan 去运行/usr/bin/passwd 这个程序的时候，在程序运行的过程中，zhangsan 用户就临时的具备了程序/usr/bin/passwd 的拥有者的权限，而程序/usr/bin/passwd 的拥有者是 root，也就是说这个时候的 zhangsan 就拥有了 root 的权限，那么自然地也就可以对口令文件/etc/shadow 进行写入操作了。



如图所示，作为 zhangsan 这个用户，本来是没有权限去对文件/etc/shadow 写入的，但是由于程序/usr/bin/passwd 被赋予了 SUID 权限，所以 zhangsan 这个用户去运行该程序时，就临时取得了 root 的身份，从而能够对文件/etc/shadow 进行写入操作。

SUID

赋予VIM编辑器SUID权限，会发现普通用户拥有了超级用户的能力。

【例题】如果赋予了一些不该拥有 SUID 权限的程序拥有了该权限，会有什么后果呢？比如 Linux 系统中经常使用 vim 编辑器来修改文件。如果给 /usr/bin/vim 这个程序赋予了 SUID 权限，那么就带来很多意想不到的问题，如下所示：

```
[zhangsan@bogon root]$ ls -l /usr/bin/vim
-rwxr-xr-x. 1 root root 4025784  6 月  13  2022 /usr/bin/vim
[zhangsan@bogon root]$ vim /etc/shadow
//如果以普通用户登录，使用 vim 编辑器修改/etc/shadow 文件，这里会提示权限不够
//切换 root 身份
[zhangsan@bogon root]$ su root
密码:
//赋予/usr/bin/vim 以 SUID 权限
[root@bogon ~]# chmod u+s /usr/bin/vim
[root@bogon ~]# ls -l /usr/bin/vim
-rwsr-xr-x. 1 root root 4025784  6 月  13  2022 /usr/bin/vim
//切换到普通用户 zhangsan，发现就可以修改/etc/shadow 文件了
[root@bogon ~]# su zhangsan
[zhangsan@bogon root]$ vim /etc/shadow
```

SUID的安全问题

黑客的最爱

该例题表明，赋予/usr/bin/vim 这个程序 SUID 权限以后，任何一个普通用户在运行 vim 命令的时候，都临时的具备了 root 的权限，也就是说这个时候，任何一个普通用户都变成了超级用户，可以对系统当中重要的资料和文件以 root 的身份进行删改，对系统可以造成严重的安全问题。这样的权限配置漏洞也经常会被黑客利用，对系统进行提权攻击。比如普通用户可以使用 vim 修改/etc/paawd、/etc/shadow 文件，把自己的 uid 修改为 0，这样普通用户就变成了超级用户，那么这个普通用户就可以随时关闭系统的服务，随时修改任何数据文件，向网络发送数据，随时添加各种类型的用户，做任何想做的事情，那么后果是难以想象的，对于 Linux 系统来说，就没有安全性可言了。

千万要记得以 root 身份登录，使用如下命令把权限修改回来：

```
[root@bogon ~]# chmod u-s /usr/bin/vim
[root@bogon ~]# ls -l /usr/bin/vim
-rwxr-xr-x. 1 root root 4025784    6 月   13    2022 /usr/bin/vim
```

SGID

用户组中的x位置被s取代，
就称为拥有SGID权限

-rwxrwsr-x

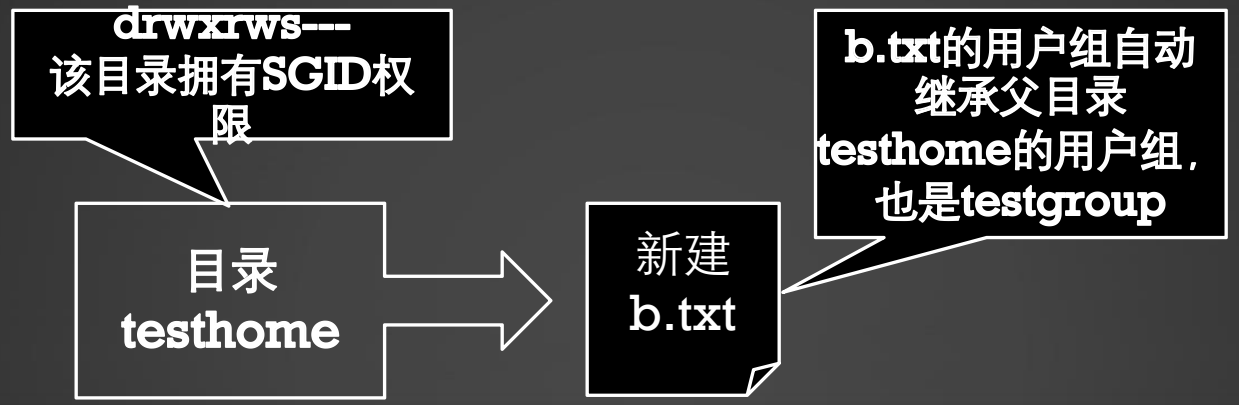
注意：SGID 特殊权限使用说明，一般情况下，SGID 权限是针对目录来说的，当一个目录配置了 SGID 权限后：

(1) 如果用户对于该目录具有 r 与 x 的权限，那么该用户可以进入此目录；

(2) 用户在该目录下的用户组将会变成该目录的用户组；

(3) 如果用户对此目录有 w 权限，则用户所创建的新文件的用户组与该目录的用户组相同。

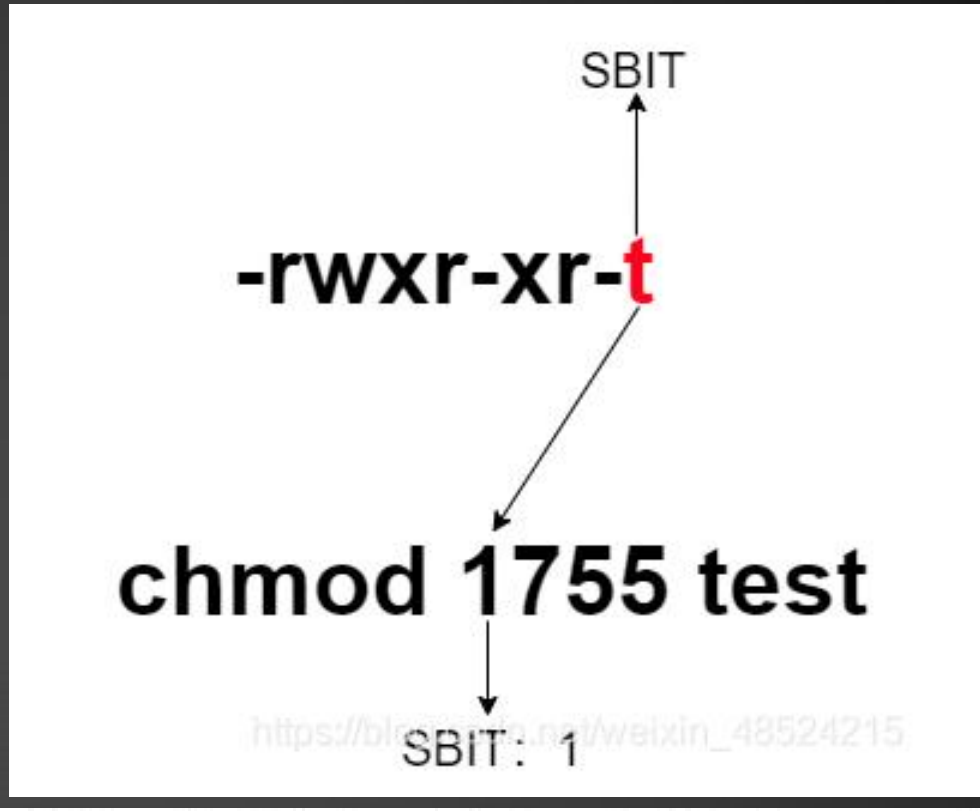
SGID含义



testhome目录所属组为testgroup

原因是：testhome拥有SGID权限

SBIT



为了说明 SBIT 特殊权限的作用，首先来分析一个例题，如下所示，首先查看 Linux 系统中/tmp 目录的权限，发现一个特殊的标志，在第 10 位上的权限 x 位置被一个 t 字符所代替。这里的 t 就是所讲的 SBIT 特殊权限，那么该权限有什么作用呢？

先解释 SBIT 权限的含义：

（1）首先 SBIT 权限只针对目录有效，对文件无效；

（2）一个目录设置 SBIT 权限后，用户对于该目录具有 w、x 权限的情况下，当用户在该目录下创建文件获目录时，只有自己与 root 才有权利删除该文件。

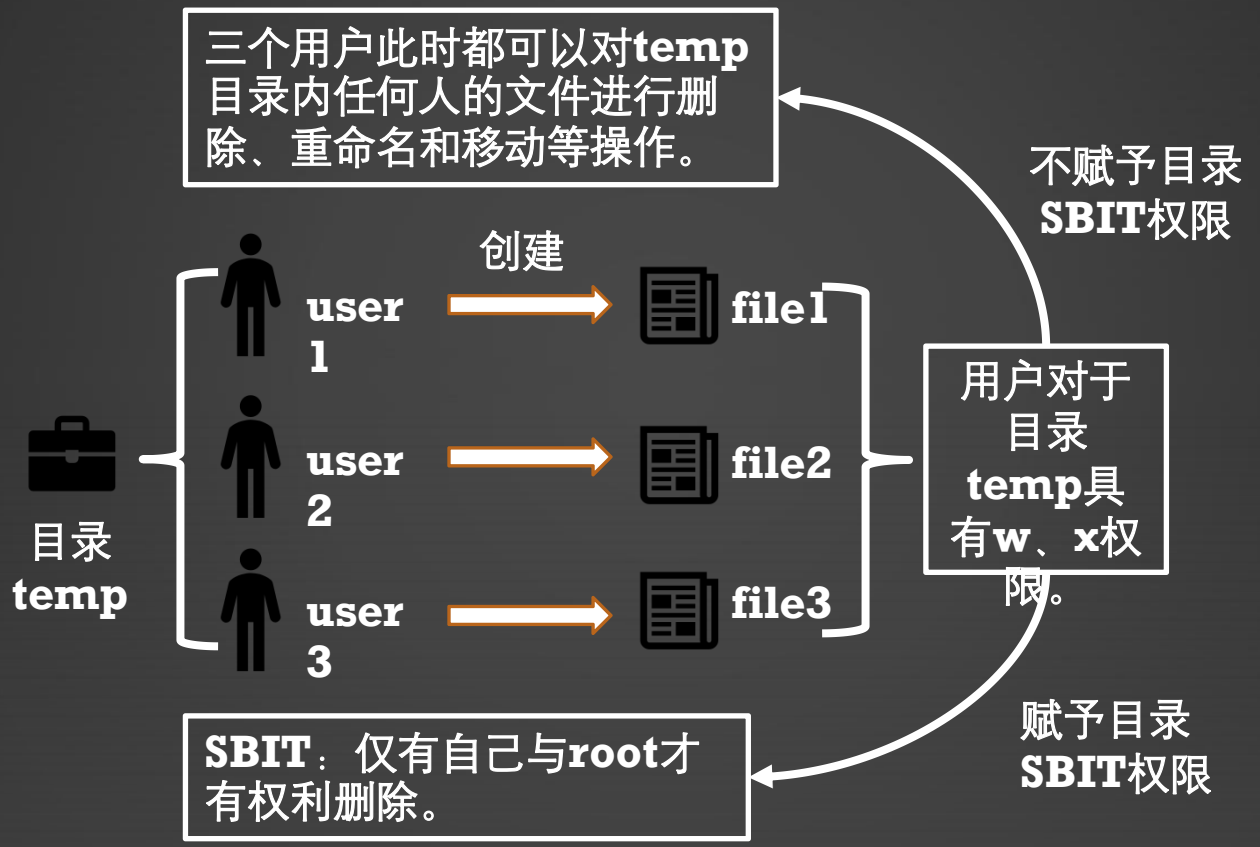
所以SBIT特殊权限又称为粘滞位，防删除位。SBIT 权限仅对目录有效，一旦目录设定了 SBIT 权限，则用户在此目录下创建的文件或目录，就只有自己和 root 才有权利修改或删除该文件。

【例题】查看/tmp 目录的 SBIT 权限，并进行删除文件测试

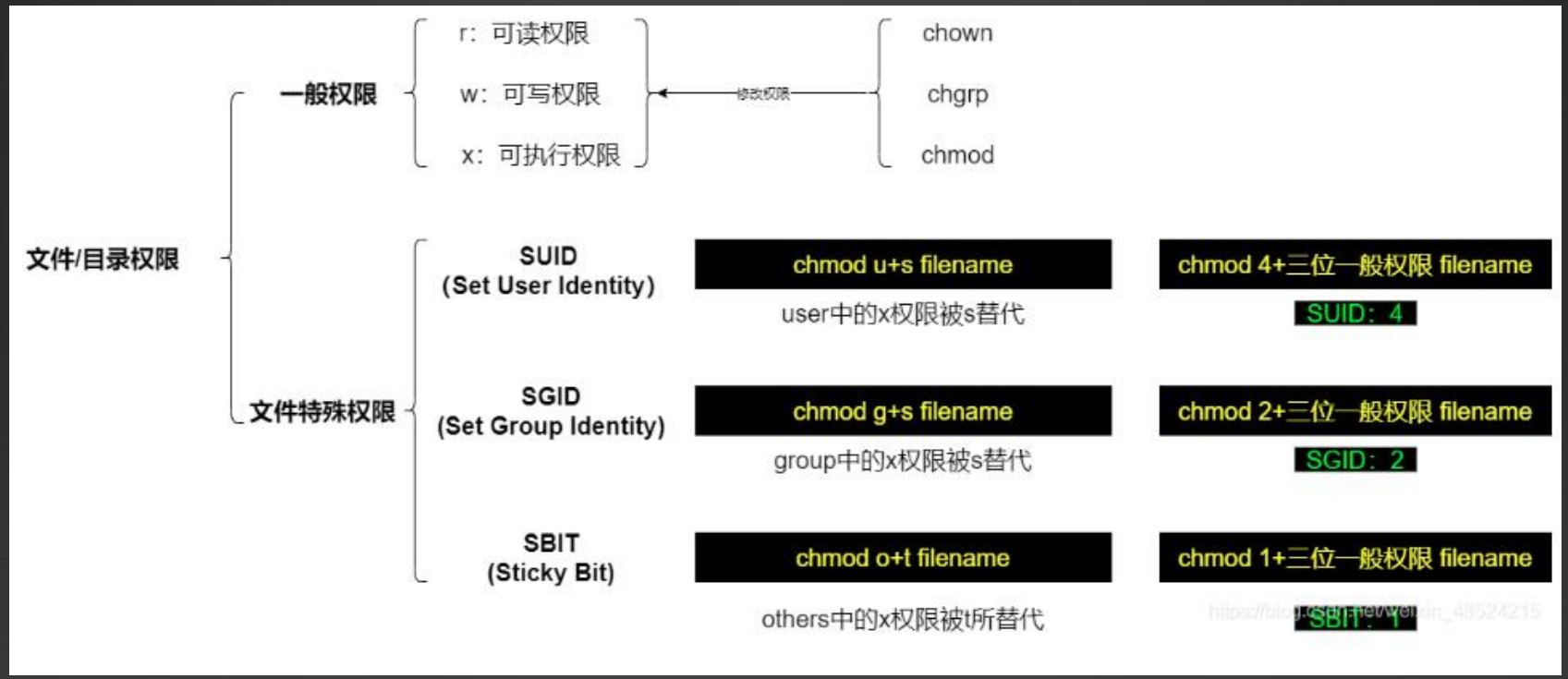
```
//以 root 身份登录系统，查看/tmp 目录的权限
[root@localhouser ~]# ll -d /tmp
drwxrwxrwt. 25 root root 4096  3 月  8 19:32 /tmp
//新建一个测试文件
[root@localhouser tmp]# touch teuser.txt
[root@localhouser tmp]# ls -l teuser.txt
-rw-r--r-- 1 root root 0  3 月  8 18:29 teuser.txt
//切换普通用户 zhangsan
[root@localhouser tmp]# su zhangsan
//删除用户，发现不能删除
[zhangsan@localhouser tmp]$ rm /tmp/teuser.txt
rm：是否删除有写保护的普通空文件 'teuser.txt'？ y
rm: 无法删除 'teuser.txt': 不允许的操作
```

分析：zhangsan 用户为什么不能删除/tmp/teuser.txt，按照正常来理解/tmp 的权限为 rwxrwxrwt，也就意味着，zhangsan 用户是可以删除/tmp/teuser.txt 文件的，但是正是因为/tmp 具有 SBIT 特殊权限，导致 zhangsan 用户的不能删除其他用户创建的文件．这也就是 SBIT 权限的特殊作用．

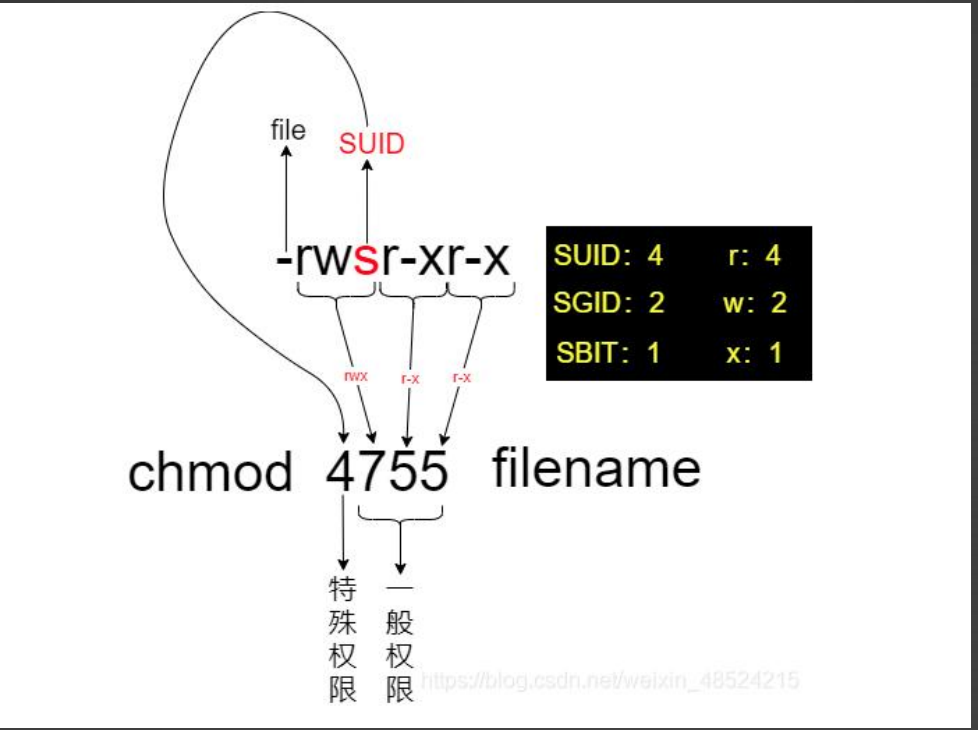
SBIT


三个用户此时都可以对temp目录内任何人的文件进行删除、重命名和移动等操作。
不赋予目录SBIT权限
创建
user 1
file1
user 2
file2
user 3
file3
目录 temp
用户对于目录temp具有w、x权限。
赋予目录SBIT权限
SBIT：仅有自己与root才有权利删除。

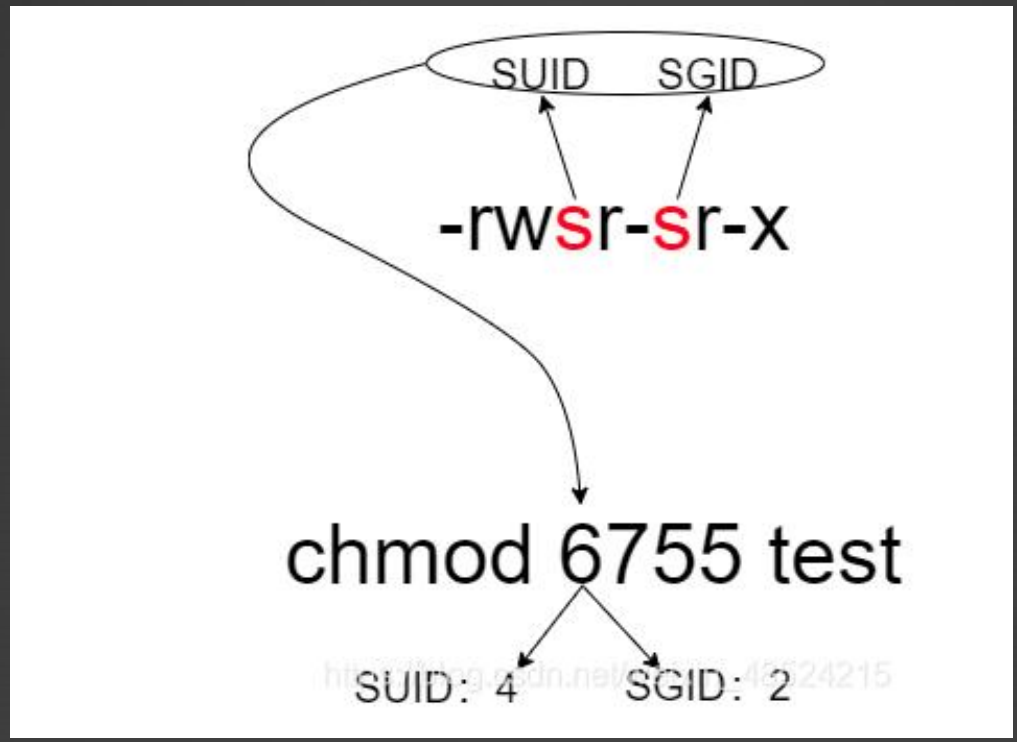
特殊权限设置方法


文件/目录权限
一般权限
r：可读权限
w：可写权限
x：可执行权限
修改权限
chown
chgrp
chmod
文件特殊权限
SUID
(Set User Identity)
chmod u+s filename
user中的x权限被s替代
chmod 4+三位一般权限 filename
SUID：4
SGID
(Set Group Identity)
chmod g+s filename
group中的x权限被s替代
chmod 2+三位一般权限 filename
SGID：2
SBIT
(Sticky Bit)
chmod o+t filename
others中的x权限被t所替代
chmod 1+三位一般权限 filename
SBIT：1

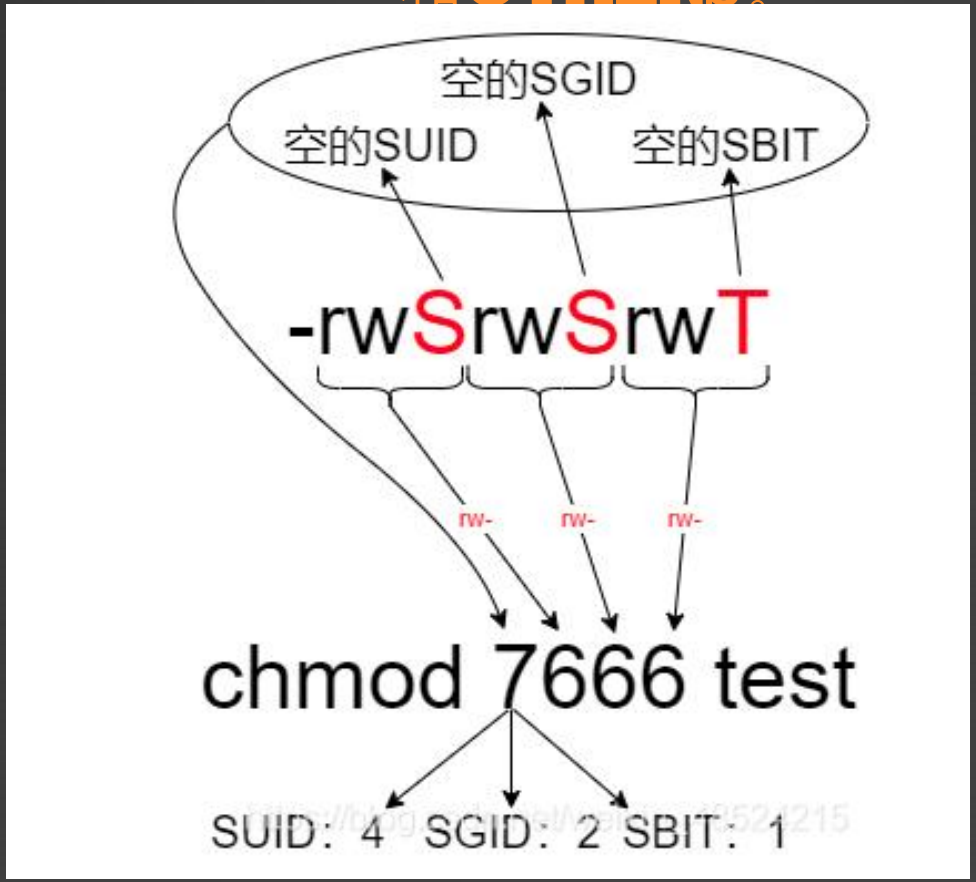
设置举例


file
SUID
-rwsr-xr-x
SUID: 4
SGID: 2
SBIT: 1
r: 4
w: 2
x: 1
rwx
r-x
r-x
chmod 4755 filename
特殊权限
一般权限
https://blog.csdn.net/weixin_48524215

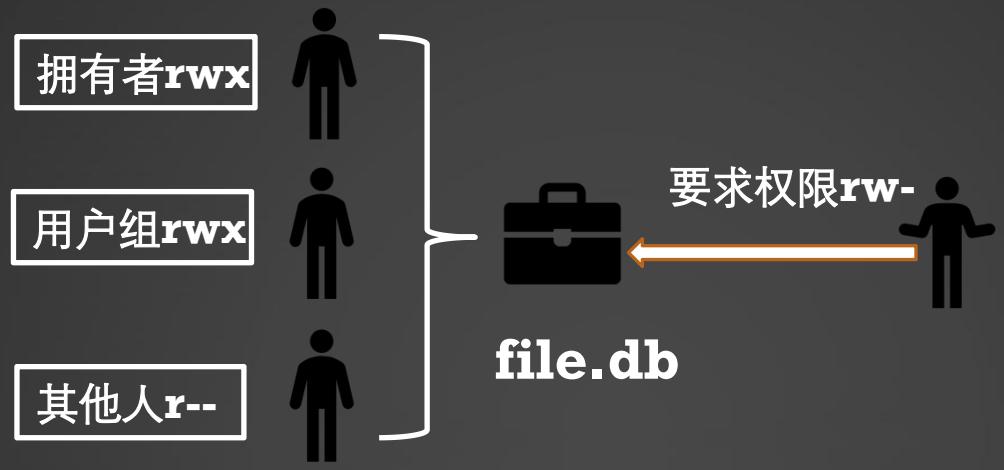
设置举例



因为S与T都是取代X这个权限的位置，这里的所有的用户都没有X权限，所以大写的S与T代表的是空的。这里文件拥有者没有执行权限X，连自己都没有可执行权限，也就无法将权限给GROUP和OTHERS。



ACL



小结

- （1）四个文件/etc/passwd /etc/shadow /etc/group /etc/gshadow
- （2）几个相关命令useradd usermod userdel groupadd groupmod groupdel
- id newgrp su sudo
- （3）文件基本权限 r w x chmod两种用法
- （4）文件隐藏权限 chattr lsattr
- （5）文件特殊权限 SUID SGID SBIT

《LINUX操作系统与安全》

《Linux系统及应用》第二次随堂测试

微信扫码或长按识别，填写内容