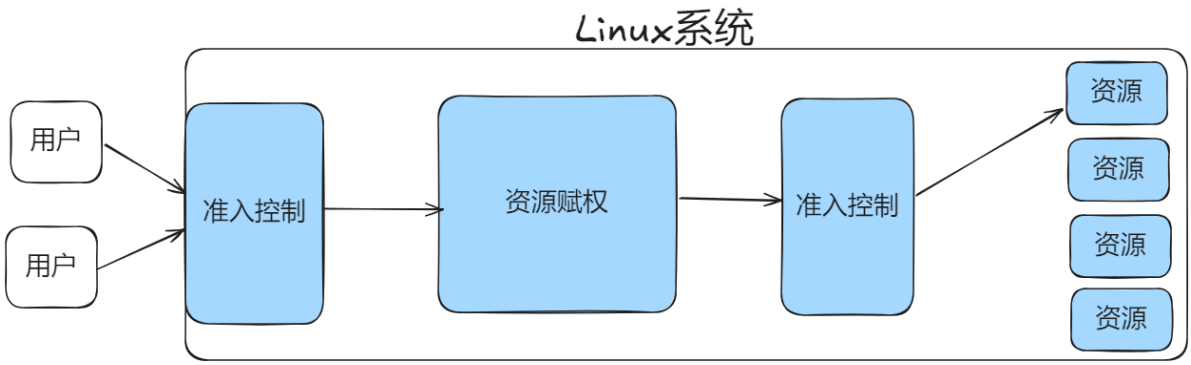
1 Linux用户组和权限管理

1.1 用户基础

1.1.1 用户&用户组

基础知识

用户操作linux的基本流程



在互联网中，几乎所有的软件级别的解决方案，只要涉及到安全操作，基本上都是按照如下的流程进行处理的：

Authentication：认证，验证用户身份
Authorization：授权，不同的用户设置不同权限
Accouting|Audition：审计(准入控制)

用户管理现状

Linux是多用户多任务操作系统，具有很好的稳定性和安全性。既然是多用户，那就意味着多个用户可以同时使用同一个Linux操作系统，因此就会涉及用户的添加、修改、删除等管理工作以及权限分配问题。

在日常工作中，一般都是领导分配一个拥有一定权限的账号，然后开展各项工作，不会开放很高的权限。但初学阶段甚至长期的工作阶段，很多人都是直接用root账户进行操作(包括我本人)，这样的目的是减少权限带来的干扰，让我们更专注于相应知识点的学习。

但是在生产环境中建议慎用root，因为权限太大，控制不当会有安全隐患。

安全攻击-系统攻击

```
May 21 23:57:59 [redacted]4 sshd[2562737]: Invalid user yangbj from 218.207.218.249 port 2714
May 21 23:58:25 [redacted]4 sshd[2562739]: Invalid user mysql from 114.206.23.151 port 59316
May 21 23:58:33 [redacted]4 sshd[2562741]: Invalid user username from 218.207.218.249 port 2715
May 21 23:59:03 [redacted]4 sshd[2562743]: Invalid user yy from 182.253.42.250 port 44495
May 21 23:59:32 [redacted]4 sshd[2562750]: Invalid user user1 from 114.206.23.151 port 48754
May 22 00:00:41 [redacted]4 sshd[2562800]: Invalid user notebook from 114.206.23.151 port 38178
May 22 00:00:53 [redacted]4 sshd[2562802]: Invalid user user123 from 218.207.218.249 port 2719
May 22 00:01:28 [redacted]4 sshd[2562804]: Invalid user minecraft from 218.207.218.249 port 2720
May 22 00:02:38 [redacted]4 sshd[2562821]: Invalid user ubuntu from 218.207.218.249 port 2722
May 22 00:03:12 [redacted]4 sshd[2562825]: Invalid user user from 218.207.218.249 port 2723
May 22 00:04:26 [redacted]4 sshd[2562831]: Invalid user drupal from 218.207.218.249 port 2725
May 22 00:05:00 [redacted]4 sshd[2562833]: Invalid user mysql from 114.206.23.151 port 52350
May 22 00:05:02 [redacted]4 sshd[2562838]: Invalid user peng from 218.207.218.249 port 2726
May 22 00:06:12 [redacted]4 sshd[2562844]: Invalid user dasusr1 from 218.207.218.249 port 2728
May 22 00:06:47 [redacted]4 sshd[2562846]: Invalid user backend from 218.207.218.249 port 2729
May 22 00:07:23 [redacted]4 sshd[2562850]: Invalid user shiny from 114.206.23.151 port 59440
May 22 00:07:57 [redacted]4 sshd[2562853]: Invalid user gustavo from 218.207.218.249 port 2731
May 22 00:08:32 [redacted]4 sshd[2562855]: Invalid user huangzengli from 218.207.218.249 port 2732
May 22 00:08:39 [redacted]4 sshd[2562857]: Invalid user nadmin from 114.206.23.151 port 48872
May 22 00:09:07 [redacted]4 sshd[2562930]: Invalid user note from 218.207.218.249 port 2733
May 22 00:09:44 [redacted]4 sshd[2562935]: Invalid user grid from 218.207.218.249 port 2734
May 22 00:10:20 [redacted]4 sshd[2562939]: Invalid user gh from 218.207.218.249 port 2735
May 22 00:10:55 [redacted]4 sshd[2562941]: Invalid user mona from 218.207.218.249 port 2736
May 22 00:11:11 [redacted]4 sshd[2562943]: Invalid user user_1 from 114.206.23.151 port 55962
May 22 00:13:44 [redacted]4 sshd[2562948]: Invalid user ram from 114.206.23.151 port 34820
May 22 00:15:03 [redacted]4 sshd[2562953]: Invalid user baba from 114.206.23.151 port 52486
May 22 00:18:46 [redacted]4 sshd[2562987]: Invalid user usertest from 114.206.23.151 port 48990
May 22 00:20:08 [redacted]4 sshd[2562989]: Invalid user apagar from 114.206.23.151 port 38420
May 22 00:22:44 [redacted]4 sshd[2562993]: Invalid user mxy from 114.206.23.151 port 45514
May 22 00:25:19 [redacted]4 sshd[2563001]: Invalid user pay from 114.206.23.151 port 52608
May 22 00:45:40 [redacted]4 sshd[2563093]: Invalid user  from 64.62.156.92 port 27631
```

安全攻击-web工具

```
root@[redacted]:~# cat /var/log/nginx/access.log.1 | grep POST
84.54.51.13 - - [21/May/2024:05:31:02 +0800] "POST /login HTTP/1.1" 404 564 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36"
173.248.150.118 - - [21/May/2024:23:34:06 +0800] "POST / HTTP/1.1" 405 568 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.129 Safari/537.36"
root@[redacted]:~# cat /var/log/nginx/access.log.1 | grep admin
3.8.115.44 - - [21/May/2024:01:42:08 +0800] "GET /admin/index.html HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
18.170.63.219 - - [21/May/2024:04:41:55 +0800] "GET /admin/index.html HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
52.56.138.101 - - [21/May/2024:11:08:48 +0800] "GET /admin/index.html HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
root@[redacted]:~# cat /var/log/nginx/access.log.1 | grep login
3.8.115.44 - - [21/May/2024:01:36:38 +0800] "GET /manage/account/login HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
3.8.115.44 - - [21/May/2024:01:59:28 +0800] "GET /cgi-bin/login.cgi HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
3.8.115.44 - - [21/May/2024:02:10:57 +0800] "GET /login.jsp HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
18.170.63.219 - - [21/May/2024:04:36:53 +0800] "GET /manage/account/login HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
18.170.63.219 - - [21/May/2024:04:57:03 +0800] "GET /cgi-bin/login.cgi HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
18.170.63.219 - - [21/May/2024:05:06:34 +0800] "GET /login.jsp HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
84.54.51.13 - - [21/May/2024:05:31:02 +0800] "POST /login HTTP/1.1" 404 564 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36"
52.56.138.101 - - [21/May/2024:11:01:49 +0800] "GET /manage/account/login HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
52.56.138.101 - - [21/May/2024:11:28:05 +0800] "GET /cgi-bin/login.cgi HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
52.56.138.101 - - [21/May/2024:11:39:18 +0800] "GET /login.jsp HTTP/1.1" 404 134 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/networks/ip-networks/deepfield/genome/)'"
```

用户和用户组

linux用户

当用户登录成功时，系统会自动分配令牌 token，包括：用户标识和组成员等信息。在Linux中每个用户是通过 User Id （UID）来唯一标识的。原则上来说，Linux的用户，主要有两类：

```
管理员：root, 0
普通用户：1-60000 自动分配
    系统用户：对守护进程获取资源进行权限分配
        1-499 （CentOS 6以前）
        1-999 （CentOS 7以后）
    登录用户：给用户进行交互式登录使用
        500+ （CentOS6以前）
        1000+（CentOS7以后）
```

linux用户组

在Linux系统中，为了方便批量对一个或多个用户进行快捷的操作，我们一般会将用户加入用户组中，通过对用户组的操作，从而实现对普通用户的属性的批量操作能力。用户组是通过Group ID（GID） 来唯一标识的。

```
管理员组：root, 0
普通用户组：
    系统用户：对守护进程获取资源进行权限分配。
        1-499（CentOS 6以前）
        1-999（CentOS7以后）
    普通用户：给用户使用
        500+（CentOS 6以前）
        1000+（CentOS7以后）
```

用户和用户组

```
用户的主要组(primary group)：
    用户必须属于一个且只有一个主组，默认创建用户时会自动创建和用户名同名的组，做为用户的主要组，由于此组中只有一个用户，又称为私有组

用户的附加组(supplementary group)：
    一个用户可以属于零个或多个辅助组，附属组
```

```
查看root用户信息
[root@rocky9 ~]# id root
用户id=0(root) 组id=0(root) 组=0(root)

查看sswang普通用户信息
[root@rocky9 ~]# id sswang
用户id=1000(sswang) 组id=1000(sswang) 组=1000(sswang),10(wheel)
```

安全上下文

```
Linux安全上下文Context：
    运行中的程序，即进程 (process)，以进程发起者的身份运行，进程所能够访问资源的权限取决于进程的运行者的身份
    分别以root 和 sswang 的身份运行 /bin/cat  /etc/shadow ，得到的结果是不同的，资源能否能被访问，是由运行者的身份决定，非程序本身
```

```
使用root用户
[root@rocky9 ~]# cat /etc/shadow
root:$6$VftC9gu7HNAEIc79$VdxTdcGlQFjiP.RdyZG6js7fHnzj3WDIDb3v6JGqyD1zhh6Rw/J1ik7
MO6UFNhZ9ihxjIiKpBwz2UEYc8pOsa.::0:99999:7:::
...

使用普通用户
[root@rocky9 ~]# su sswang
[sswang@rocky9 root]$ cat /etc/shadow
cat: /etc/shadow: 权限不够
```

用户登录和退出

```
关于用户登录和退出，在linux系统中，会有对应的日志文件进行相关的记录动作：
    Rocky9的日志：
        /var/log/secure
        /var/log/audit/audit.log
    ubuntu的日志：
        /var/log/auth.log
```

1.1.2 配置文件

基础知识

用户和组的主要配置文件

```
/etc/passwd：用户及其属性信息(名称、UID、主组ID等）
/etc/shadow：用户密码及其相关属性
/etc/group：组及其属性信息
/etc/gshadow：组密码及其相关属性
```

```
[root@rocky9 ~]# ll /etc/passwd
-rw-r--r--. 1 root root 2105  9月 23 03:39 /etc/passwd
[root@rocky9 ~]# ll /etc/shadow
----------. 1 root root 1127  9月 23 03:39 /etc/shadow
[root@rocky9 ~]# ll /etc/group
-rw-r--r--. 1 root root 826  9月 23 09:22 /etc/group
[root@rocky9 ~]# ll /etc/gshadow
----------. 1 root root 664  9月 23 09:22 /etc/gshadow
结果显示：
    shadow 只有 root 才有读和写权限，这样就保证了密码文件的安全性
    passwd和group 文件每个用户都有读权限
```

passwd文件格式

查看用户及其属性信息

```
[root@rocky9 ~]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

从passwd文件中获取某一个用户信息

```
[root@rocky9 ~]# getent passwd sswang
sswang:x:1000:1000:sswang:/home/sswang:/bin/bash
```

注意：

每一列都是以 ： 为分隔符

项	名称	含义	备注
root	用户名	用户登录系统的用户名	建议不要超过8位，不要使用特殊符号或数字
x	密码	密码位（并没有真实的存放密码，因为密码存放在该文件中并不安全）	建议8位以上，大小字母+数字的组合
0	UID	用户标识号	参考用户类型
0	GID	用户缺省组标识号	参考用户组
root	描述信息	例如存放用户全称等信息	建议添加用户描述
/root	宿主目录	用户登录系统后的缺省目录	每个用户必须有个宿主目录，创建完用户后系统会在 /home/ 目录下创建一个与用户名相同的目录，用户登录系统后缺省的就是该目录。
/bin/bash	命令解释器	用户使用的 Shell，默认为 bash	bash 用的较多

shadow文件格式

```
[root@rocky9 ~]# cat /etc/shadow
root:$6$VftC9gu7HNAEIc79$VdxTdcGlQFjiP.RdyZG6js7fHnzj3WDIDb3v6JGqyD1zhh6Rw/J1ik7
MO6UFNhZ9ihxjIiKpBwz2UEYc8pOsa.::0:99999:7:::
```

项	名称	含义	备注
root	用户名	用户登录系统的用户名	
6qHd…	密码	加密的密码	用户的真实密码，如果将该信息**为空**，表示用户登录时可以不需要输入密码；如果**为!!**，表示禁止登录（或未设置密码）。
空	最后修改时间	用户最后一次修改密码的天数	man 5 shadow
0	最小时间间隔	两次修改密码之间的最小天数	man 5 shadow
99999	最大时间间隔	密码保持有效的最多天数	多少天后强制用于修改密码
7	警告时间	从系统刚开始警告到密码失效的天数	用户密码到期前多少天提示用户
空	账号闲置时间	账号闲置时间	用户多久没有登录了，空表是没有闲置
空	失效时间	密码失效的绝对天数	密码失效多长时间了
空	标志	一般不使用（保留字段）	

范例：生成随机密码

```
[root@rocky9 ~]# tr  -dc '[:alnum:]' < /dev/urandom | head -c 12
Eb9pU8hpis2W
[root@rocky9 ~]# tr -dc 'A-Za-z0-9_!@#$%^&*()-+=' < /dev/urandom | head -c 12
hi$DH7j$4+u=
```

注意：

-dc 删除所有不满足条件的字符

/dev/urandom是一个特殊的文件，提供了伪随机数流。

```
openssl生成9个字节的随机数，然后通过base64进行编码【3个字节转换成base64的4个字符】
[root@rocky9 ~]# openssl rand -base64 9
VUsHXvHejGOl
[root@rocky9 ~]# openssl rand -base64 12
Ml2ruTni6XDrmygh

mkpasswd 接收信息后，生成一个密码串
[root@rocky9 ~]# mkpasswd 123456
$y$j9T$g4A3lDW88Ue7tSJihSUm2.$fpJ83ev./R7MswhpT0uQ/moOvcUx7tYsIF8idDOkEZ4

生成随机密码惨网站：
    https://suijimimashengcheng.51240.com/
    https://www.lddgo.net/string/randompassword
```

group文件格式

```
查看group文件的信息
[root@rocky9 ~]# cat /etc/group
root:x:0:
```

项	名称	含义	备注
root	组名	用户登录时所在的组	建议不要超过8位，有意义的组名称
x	组密码	组密码位	一般不使用。用于不是组中的成员时，如果知道组密码的情况下，可以登录改组享有该组的权限。
0	GID	组标识号	
空	组内用户列表	所属组该组的所有用户列表（不包含缺省组为该组的用户）	用户之间以(,)逗号分隔

gshdow文件格式

```
[root@rocky9 ~]# cat /etc/gshadow
root:::
```

项	名称	含义	备注
mygroup	组名	必须是系统中已经存在的有效组	
6nhM…	组密码	加密了的密码	
myuser	管理员	管理员可以更改组密码和成员	
myuser	组内用户列表	属于该组的所有用户列表	用户之间以(,)逗号分隔

1.1.3 配置操作

配置修改

工具简介

```
    虽然我们可以借助于nona或vim来对上面的文件进行直接修改，但是我们不推荐。因为这些文件跟用户和权限有关的文件具有敏感性，所以雄通过提供了专用的工具来修改这几个文件
```

vipw和vigr命令

```
vipw|vigr 命令用于编辑/etc/passwd，/etc/group，/etc/shadow，/etc/gshadow
    vipw 默认编辑 /etc/passwd 文件
    vigr 默认编辑 /etc/group 文件

常见选项：
    -g|--group #编辑 group 文件
    -p|--passwd #编辑 passwd 文件
    -s|--shadow #编辑 /etc/shadow 或 /etc/gshadow 文件
```

简单实践

```
默认修改的是 /etc/passwd
[root@rocky9 ~]# vipw
您已经修改了 /etc/passwd。
出于一致性的考虑，您可能需要修改 /etc/shadow。
请使用命令“vipw -s”来进行这个工作。

根据提示来修改/etc/shadow
[root@rocky9 ~]# vipw -s
您已经修改了 /etc/shadow。
出于一致性的考虑，您可能需要修改 /etc/passwd。
请使用命令“vipw”来进行这个工作。
```

```
默认修改/etc/group
[root@rocky9 ~]# vigr
您已经修改了 /etc/group。
出于一致性的考虑，您可能需要修改 /etc/gshadow。
请使用命令“vigr -s”来进行这个工作。

根据提示来修改 /etc/gshadow
[root@rocky9 ~]# vigr -s
您已经修改了 /etc/gshadow。
出于一致性的考虑，您可能需要修改 /etc/group。
请使用命令“vigr”来进行这个工作。
```

配置检查

pwck命令

```
命令使用：
    pwck [options] [passwd_file | shadow_file]
    对用户相关配置文件进行检查，默认检查文件为 /etc/passwd
选项
    -q|--quiet #只报告错误，忽略警告
    -r|--read-only #显示错误和警告，但不改变文件
    -R|--root CHROOT_DIR #chroot 到的目录
    -s|--sort #通过 UID 排序项目
```

grpck命令

```
命令使用：
    grpck [options] [group [gshadow]]
    对用户组相关配置文件进行检查，默认检查文件为 /etc/group，/etc/gshadow

选项：
    -r|--read-only #显示错误和警告，但不改变文件
    -R|--root CHROOT_DIR #chroot 到的目录
    -s|--sort #通过 UID 排序项目
```

简单实践

```
[root@rocky9 ~]# pwck
无效的影子密码文件项
删除“”一行？y
pwck：文件已更新

[root@rocky9 ~]# grpck
无效的组文件条目
删除“”一行？y
grpck：文件已更新
```

1.2 用户管理

1.2.1 用户组管理1

命令简介

groupadd命令可以创建用户组

```
    groupadd 命令是 Linux 和类 Unix 系统中用于创建新用户组的工具。用户组是管理用户账户和文件权限的一种有效方式，它允许你将多个用户归入同一个组中，并对这个组内的所有用户应用相同的权限设置。
```

```
命令格式
    groupadd [options] GROUP

稍微常用选项
    -r|--system #创建一个系统组 CentOS 6之前：ID<500，CentOS 7以后：ID<1000

一般选项
    -f|--force #如果组已经存在则成功退出
    -g|--gid GID #新建组时指定组ID，默认是系统分配，指定值不要超过[GID_MIN，GID_MAX]
    -K|--key KEY=VALUE #不使用 /etc/login.defs 中的默认值，自己指定，比如 -K
GID_MIN=100
    -o|--non-unique #允许创建有重复 GID 的组
    -p|--password PASSWORD #为新组使用此加密过的密码，这里要用加密后的密文，不然就是直接是明文      在/etc/gshadow 里面
```

groupmod命令用于修改group属性

```
命令格式:
    groupmod [options] GROUP

常见选项
    -g|--gid GID #将组 ID 改为 GID
    -n|--new-name NEW_GROUP #改名为 NEW_GROUP
    -o|--non-unique #允许使用重复的 GID
    -p|--password PASSWORD #将密码更改为(加密过的) PASSWORD
```

groupdel命令可以删除用户组

```
命令格式
    groupdel [options] GROUP

常用选项
    -f|--force #强制删除，即使是用户的主组也强制删除组,但会导致无主组的用户不可用无法登录
```

简单实践

创建用户组

```
创建用户组
[root@rocky9 ~]# groupadd group1

查看用户组
[root@rocky9 ~]# getent group group1
group1:x:1026:
```

指定id创建用户组

```
指定id创建用户组
[root@rocky9 ~]# groupadd group2 -g 10086

查看用户组
[root@rocky9 ~]# getent group group2
group2:x:10086:
```

创建系统用户组

```
[root@rocky9 ~]# groupadd -r sgroup1
[root@rocky9 ~]# getent group sgroup1
sgroup1:x:978:
```

重复创建用户组

```
创建已存在用户组
[root@rocky9 ~]# groupadd group1
groupadd: “group1”组已存在

强制创建已存在用户组
[root@rocky9 ~]# groupadd -f group1
[root@rocky9 ~]# groupadd -f group2 -g 10087
[root@rocky9 ~]# groupadd -f group3
```

```
检查效果
[root@rocky9 ~]# getent group group1 group2 group3
group1:x:1026:
group2:x:10086:
group3:x:10087:
结果显示：
    对于已存在的用户组，基本上不再处理
    如果用户组不存在，则正常创建用户组
```

修改用户组属性

```
创建用户组
[root@rocky9 ~]# groupadd group4 -g 20086
[root@rocky9 ~]# getent group group4
group4:x:20086:

修改用户组的属性
[root@rocky9 ~]# groupmod -g 20087 group4
[root@rocky9 ~]# getent group group4
group4:x:20087:

修改用户组名字
[root@rocky9 ~]# groupmod -n group4-1 group4
[root@rocky9 ~]# getent group group4
[root@rocky9 ~]# getent group group4-1
group4-1:x:20087:
```

删除用户组

```
删除指定用户组 - 只能一个一个的删
[root@rocky9 ~]# groupdel -f group1
[root@rocky9 ~]# groupdel -f group2
[root@rocky9 ~]# groupdel -f group3
[root@rocky9 ~]# groupdel -f group4-1
[root@rocky9 ~]# groupdel -f sgroup1

确认用户组效果
[root@rocky9 ~]# getent group group1 group2 group3 group4-1 sgroup1
[root@rocky9 ~]#
```

1.2.2 用户创建

命令简介

id用于查看用户的基本信息

```
命令格式
    id [username]

一般选项
    -a #显示详细信息，默认选项
    -Z|--context #仅显示安全上下文信息，要开启selinux 配置才有
    -g|--group #仅显示GID，就是只显示主组ID
    -G|--groups #显示主组和附加组ID，就是所有组ID
    -n|--name #显示用户名或组名，要组合使用 -nu|-ng|-nG
    -u|--user #仅显示UID
```

useradd 命令可以创建新的Linux用户

```
命令使用
    useradd [options] LOGIN
    useradd -D              # 查看默认的配置属性

常用选项
    -u|--uid UID #指定UID
    -g|--gid GID #指定用户组，-g groupname|--gid GID
    -m|--create-home #创建家目录，一般用于登录用户
    -p|--password PASSWORD #设置密码，这里的密码是以明文的形式存在于/etc/shadow 文件中

一般选项
    -c|--comment COMMENT #新账户的 GECOS 字段
    -d|--home-dir HOME_DIR #指定家目录，可以是不存在的，指定家目录，并不代表创建家目录
    -s|--shell SHELL #指定 shell，可用shell在/etc/shells 中可以查看
    -r|--system # 创建系统用户
        # CentOS 6之前 ID<500
        # CentOS7 以后 ID<1000不会创建登录用户相关信息
    -M|--no-create-home #不创建家目录，一般用于不用登录的用户
    -o|--non-unique #允许使用重复的 UID 创建用户
    -G|--groups GROUP1[,GROUP2,...] #为用户指明附加组，组须事先存在
    -N|--no-user-group #不创建同名的组,使用users组做主组
    -D|--defaults #显示或更改默认的 useradd 配置，默认配置文件是 /etc/default/useradd
    -e|--expiredate EXPIRE_DATE #指定账户的过期日期 YYYY-MM-DD 格式
    -f|--inactive INACTIVE #密码过期之后，账户被彻底禁用之前的天数，0 表示密码过期立即禁用，-1表示不使用此功能
    -k|--skel SKEL_DIR #指定家目录模板，创建家目录，会生成一些默认文件，如果指定，就从该目录复制文件，默认是/etc/skel/，要配合-m
    -K|--key KEY=VALUE #不使用 /etc/login.defs 中的默认值，自己指定，比如 -K UID_MIN=100
    -l|--no-log-init #不将用户添加到最近登录和登录失败记录，前面讲到的3a认证审计，就在此处 lastlog|lastb|cat /var/log/secure
```

newusers 批量创建用户

```
命令使用
    newusers files

文件格式
    用户名:密码:uid:gid:描述信息:家目录:登录shell
    u1:123456:1024:1024::/home/u1:/bin/bash
```

简单实践

查看用户信息

```
查看当前登录用户
[root@rocky9 ~]# id
用户id=0(root) 组id=0(root) 组=0(root)

查看指定用户信息
[root@rocky9 ~]# id sswang
用户id=1000(sswang) 组id=1000(sswang) 组=1000(sswang),10(wheel)

通过选项，仅显示部分内容
[root@rocky9 ~]# id -ng sswang
sswang
```

用户创建

```
创建用户默认会添加一个组
[root@rocky9 ~]# useradd nihao
[root@rocky9 ~]# id nihao
用户id=1001(nihao) 组id=1001(nihao) 组=1001(nihao)
[root@rocky9 ~]# getent gshadow nihao
nihao:!::

查看密码信息
[root@rocky9 ~]# getent passwd nihao
nihao:x:1001:1001::/home/nihao:/bin/bash
结果显示：
    它是一个 可以登录的用户
```

```
自动创建了用户家目录
[root@rocky9 ~]# ls /home/
nihao  sswang

用户家目录下，自动添加了一些配置文件
[root@rocky9 ~]# ls -a /home/nihao/
.  ..  .bash_logout  .bash_profile  .bashrc  .mozilla  .zshrc

这些文件来自于 /etc/skel 目录下的文件
[root@rocky9 ~]# ls /etc/skel/ -a
.  ..  .bash_logout  .bash_profile  .bashrc  .mozilla  .zshrc
```

指定用户组创建用户

```
创建用户组
[root@rocky9 ~]# groupadd -g 10086 group1

创建用户
[root@rocky9 ~]# useradd -g 10086 user1

查看用户创建效果
[root@rocky9 ~]# id user1
用户id=1026(user1) 组id=10086(group1) 组=10086(group1)
```

```
在用户存在的前提下，删除用户组
[root@rocky9 ~]# groupdel -f group1
[root@rocky9 ~]# id user1
用户id=1026(user1) 组id=10086 组=10086
结果显示：
    强制删除用户组之后，显示该组名的地方，只能显示该组ID，如果此时新建与组ID相同的新组，则被删除的组的数
据会被新组关联

再次创建相同id的用户组
[root@rocky9 ~]# groupadd group2 -g 10086
[root@rocky9 ~]# id user1
用户id=1026(user1) 组id=10086(group2) 组=10086(group2)
```

ubuntu系统下的用户创建

```
ubuntu系统创建用户默认是不创建用户组的
root@sswang:~# useradd ubuntu-user1
root@sswang:~# ls /home/
sswang
root@sswang:~# getent passwd ubuntu-user1
ubuntu-user1:x:1001:1001::/home/ubuntu-user1:/bin/sh

ubuntu系统创建用户时提供家目录
root@sswang:~# userdel ubuntu-user2
root@sswang:~# useradd -m ubuntu-user2
root@sswang:~# ls /home/
sswang  ubuntu-user2
```

其他实践-基本没用

```
查看用户创建的基本属性
[root@rocky9 ~]# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes

更改用户创建的属性信息
[root@rocky9 ~]# useradd -D -s /bin/sh

检查配置
[root@rocky9 ~]# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/sh
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes

创建用户后再进行检测
[root@rocky9 ~]# useradd test1
```

```
[root@rocky9 ~]# getent passwd test1
test1:x:1002:1002::/home/test1:/bin/sh

恢复配置
[root@rocky9 ~]# useradd -D -s /bin/bash
```

批量创建用户

```
准备用户文件
[root@rocky9 ~]# cat user.txt
sswang1:123456:1024:1024::/home/sswang1:/bin/bash
sswang2:123456:1025:1025::/home/sswang2:/bin/bash

批量创建用户
[root@rocky9 ~]# newusers user.txt

检查效果
[root@rocky9 ~]# id sswang1
用户id=1024(sswang1) 组id=1024(sswang1) 组=1024(sswang1)
[root@rocky9 ~]# id sswang2
用户id=1025(sswang2) 组id=1025(sswang2) 组=1025(sswang2)
```

查看别人的程序是如何使用用户创建的

```
rocky 9 系统
    dnf install postfix -y
    rpm -q --scripts postfix

ubuntu系统
    安装软件并下载软件然后拆解软件包
    apt install postfix -y
    apt download postfix
    dpkg-deb -R postfix_xxx.deb postfix

    查看安装脚本
    # ls postfix/etc/postfix/
    dynamicmaps.cf.d  postfix-files  postfix-files.d  postfix-script  post-
install  sasl
```

1.2.3 密码拆解

基础知识

用户密码拆解

```
查看用户密码信息
[root@rocky9 ~]# getent shadow nihao
nihao:$6$cKdUJegv4ahduOCD$inEcX.EbKACalTOjjvmgWn.XrUw76OUFNU3x9KScwFIRdrHbA.7zy3
Lto46N32Ctq8uVckpdLeU7HLo7SAWQFO:19991:0:99999:7:::
```

```
密码格式：
$6$cKdUJegv4ahduOCD$inEcX.EbKACalTOjjvmgWn.XrUw76OUFNU3x9KScwFIRdrHbA.7zy3Lto46N
32Ctq8uVckpdLeU7HLo7SAWQF0

第一部分：指定加密算法
    $6
第二部分：指定二次校验选项 - salt 盐值
    $cKdUJegv4ahduOCD
第三部分：通过 加密算法(salt值 + 密码) 加密后的一段字符串
    $inEcX.EbKACalTOjjvmgWn.XrUwxxx.7zy3Lto46N32Ctq8uVckpdLeU7HLo7SAWQF0
```

crypt

```
主要用于加密和解密文件或密码。
```

简单实践

查看生成密码的算法解读

```
[root@rocky9 ~]# whatis crypt
crypt (5)             - storage format for hashed passphrases and available
hashing methods
crypt (3p)            - string encoding function (CRYPT)

[root@rocky9 ~]# man 5 crypt
...

   yescrypt
     ... # yescrypt 算法生成密码的时候，是随机生成的,每次都不一样
     Hashed passphrase format
         \$y\$[./A-Za-z0-9]+\$[./A-Za-z0-9]{,86}\$[./A-Za-z0-9]{43}
     ...
   scrypt
     ... # scrypt 算法生成密码的时候，前面有$7的标识
     Hashed passphrase format
         \$7\$[./A-Za-z0-9]{11,97}\$[./A-Za-z0-9]{43}
     ...
   sha512crypt
     ... # sha512crypt 算法生成密码的时候，前面有$6的标识
     Hashed passphrase format
         \$6\$(rounds=[1-9][0-9]+\$)?[^$:\n]{1,16}\$[./0-9A-Za-z]{86}
     ...
     Salt size
         6 to 96 bits
     ...
```

生成密码实践

```
使用默认的算法，salt的值是随机的生成随机的密码
openssl passwd -6 123456

指定salt生成 固定内容的密码
openssl passwd -salt abcd -6 123456
```

```
[root@rocky9 ~]# openssl passwd -6 123456
$6$gkHGN9oMgEinZFI6$u/G8.mHCgs4NfbxZObXh8jaI5RhpaonakaHnD.6ft3S2cb/sQ6TRgb7dtsIb5/gXrr0zPr4e73szfjI5i.OSE0
[root@rocky9 ~]# openssl passwd -6 123456
$6$/aVtk2EWoDbkJ2QX$wyTdaUcujf0pH1oxQodbuIsnzNOS/MWq8KBkYoito6JZwVOUaQl4ndDtHOErYCwrdwLBwbxZ7qTuNgMrBUSvv/
[root@rocky9 ~]# openssl passwd -6 123456
$6$jD5dIAest3E6E7fQ$py9DIfLAG89EgyYTgU1uSDCFzhGKq6iCa2bdj7YGYWQhE7RUyWrJQlmQiypRv8l8PYQHU78Do1C/zpiWAzoy30
[root@rocky9 ~]# openssl passwd -6 123456
$6$VFswRTlQKCR5e/P7$6NeL12L7H9U5g6ibdulBeJcox4foB0CDY5fsNiOtAaUWwngAumpbL560S5vEdaLHuKdFhNd/PLcFK.n6ngr3l/
[root@rocky9 ~]# openssl passwd -6 123456
$6$neOfHzjNWbM7Hhhc$MKYXNWRxQ3JY8Ugc8m4/zbghFfcLvE..Os3sC97100pK48fBSUbIUNJrRMfwu3B7zkVHKOvFZktASBuCoJKAr1
[root@rocky9 ~]# openssl passwd -salt abcd -6 123456
$6$abcd$BlbaPOqDDm3eVW.GrfRm4Vcuz4.0WgysdL5FY1CIn/4a5QwuySngbORfHwsHfXQERZejgHIflz1fJkKs82Iix.
[root@rocky9 ~]# openssl passwd -salt abcd -6 123456
$6$abcd$BlbaPOqDDm3eVW.GrfRm4Vcuz4.0WgysdL5FY1CIn/4a5QwuySngbORfHwsHfXQERZejgHIflz1fJkKs82Iix.
[root@rocky9 ~]# openssl passwd -salt abcd -6 123456
$6$abcd$BlbaPOqDDm3eVW.GrfRm4Vcuz4.0WgysdL5FY1CIn/4a5QwuySngbORfHwsHfXQERZejgHIflz1fJkKs82Iix.
[root@rocky9 ~]#
```

验证登录用户密码

```
查看sswang用户密码
[root@rocky9 ~]# getent shadow sswang
sswang:$6$2LXN6/Ci5Msxpg6K$qzkyPrdkYmKd4GTnvuims/AiIbP6Wylo.3B3t4dbihS/WRD6oMh09
8xsqxaZ.kUZ8zYwWh3j6kvCTb.rNhOzY1::0:99999:7:::

根据密码提示，生成对应的passwd密码
[root@rocky9 ~]# openssl passwd -salt 2LXN6/Ci5Msxpg6K -6 123456
$6$2LXN6/Ci5Msxpg6K$qzkyPrdkYmKd4GTnvuims/AiIbP6Wylo.3B3t4dbihS/WRD6oMh098xsqxaZ
.kUZ8zYwWh3j6kvCTb.rNhOzY1

结果显示：
    sswang的密码就是 123456
```

1.2.4 更改密码

命令解析

passwd 更改用户命令

```
    passwd 命令主要用于更新用户密码。当你作为用户登录到系统时，你可以通过运行 passwd 命令来更
改你自己的密码。如果你以 root 用户身份运行 passwd 命令并指定用户名作为选项（例如 passwd 用户
名），则可以更改指定用户的密码。
    场景：passwd 命令主要用于交互式地更改用户密码，适用于单个用户的情况。
```

```
命令使用
    passwd 用户名
```

```
在rocky 和 centos系列的系统中，可以使用如下方式：
    echo '密码' | passwd --stdin 用户名
    passwd --stdin 用户名 <<<密码
```

chpasswd 更改用户命令

```
    chpasswd 命令是一个批量更改用户密码的工具，它允许你从文件或命令行中读取用户名和密码对，并
更新这些用户的密码。这个命令非常适合在系统初始化或批量更新用户密码时使用。
    场景：chpasswd 命令则用于批量更改用户密码，更适合在系统管理或脚本自动化时使用。
```

```
命令使用
    chpasswd < file
    echo "用户名:密码" | chpasswd

文件格式:
    用户名:密码
    u1:1234567
```

简单实践

修改密码

```
[root@rocky9 ~]# passwd nihao
更改用户 nihao 的密码 。
新的密码:                       # jtsghrz123
重新输入新的密码:                # jtsghrz123
passwd:所有的身份验证令牌已经成功更新。

[root@rocky9 ~]# passwd nihao
更改用户 nihao 的密码 。
新的密码:                          # 123456
无效的密码: 密码少于 8 个字符
重新输入新的密码:                  # 123456
passwd:所有的身份验证令牌已经成功更新。
```

修改密码常见提示

```
[root@rocky9 ~]# passwd nihao
更改用户 nihao 的密码 。
新的密码:                               # 输入 123456
无效的密码: 密码少于 8 个字符

[root@rocky9 ~]# passwd nihao
更改用户 nihao 的密码 。
新的密码:                               # 输入 12345678aaa
无效的密码: 密码未通过字典检查 - 太简单或太有规律

[root@rocky9 ~]# passwd nihao
更改用户 nihao 的密码 。
新的密码:
重新输入新的密码:
抱歉,密码不匹配。
passwd: 鉴定令牌操作错误
```

批量修改密码

```
准备修改密码
[root@rocky9 ~]# cat passwd.txt
nihao:woainizg123
sswang1:woainizg123
sswang2:woainizg123

批量修改密码
[root@rocky9 ~]# chpasswd < passwd.txt
```

密码检测

```
查看生成后的密码
[root@rocky9 ~]# getent shadow nihao sswang1 sswang2
nihao:$6$Awaq1Bssyj9.7lFi$8yg/MeuFbokKbf9qzQIKd2HBdzOUcwejKG8H4ncKLerOqsLWH/gVTO
l/4POFdKiUTO3F1M83DAFRex4C7Z7pM.:19991:0:99999:7:::
sswang1:$6$BtLBiljnln2KxdPU$wMowVBGJVnvoacZWtrCpiMl6Xp2G9o1zN7JAiKQF/uGdHVKBPwdy
vQxrlwFtAkb7bYEbOHA8ixmcKPcLf2wZ30:19991:0:99999:7:::
sswang2:$6$KmLCmraSOEGSiuvW$fduCx/TKRx/zVpQzO5nO9a5.rmUvZ2nBbrmlRnwceYXHaB3oHut
VFMYxkYikAa7vmyoM.ywB3jUOecbpShsQ.:19991:0:99999:7:::

测试密码是否是我们改变的，一sswang2为例
[root@rocky9 ~]# openssl passwd -salt KmLCmraSOEGSiuvW -6 woainizg123
$6$KmLCmraSOEGSiuvW$fduCx/TKRx/zVpQzO5nO9a5.rmUvZ2nBbrmlRnwceYXHaB3oHutVFMYxkYi
kAa7vmyoM.ywB3jUOecbpShsQ.
结果显示：
    密码更改成功。
```

便捷修改密码

```
修改前检查密码
[root@rocky9 ~]# getent shadow sswang1
sswang1:$6$BtLBiljnln2KxdPU$wMowVBGJVnvoacZWtrCpiMl6Xp2G9o1zN7JAiKQF/uGdHVKBPwdy
vQxrlwFtAkb7bYEbOHA8ixmcKPcLf2wZ30:19991:0:99999:7:::

单行便捷修改密码
[root@rocky9 ~]# echo sswang1:123456s | chpasswd

再次检查密码
[root@rocky9 ~]# getent shadow sswang1
sswang1:$6$aMXZJzUyx1UZn4ij$zjloCGOw3hnHBOXMaoX18XMP42qepvIp6LV4dJMqRQQnjme.mzh7
lmn2KasaK1Rk8mHkYCYch.5FaV/2Aa/jK1:19991:0:99999:7:::

验证密码
[root@rocky9 ~]# openssl passwd -salt  aMXZJzUyx1UZn4ij -6 123456s
$6$aMXZJzUyx1UZn4ij$zjloCGOw3hnHBOXMaoX18XMP42qepvIp6LV4dJMqRQQnjme.mzh7lmn2Kasa
K1Rk8mHkYCYch.5FaV/2Aa/jK1
结果显示：
    密码修改成功。
```

centos&rocky专属修改密码

```
rocky系统上更改密码操作
[root@rocky9 ~]# echo taihuaile | passwd --stdin sswang1
更改用户 sswang1 的密码 。
passwd：所有的身份验证令牌已经成功更新。
[root@rocky9 ~]# passwd --stdin sswang1 <<<123456
更改用户 sswang1 的密码 。
passwd：所有的身份验证令牌已经成功更新。
```

1.2.5 用户属性

命令解读

usermod命令用于进行用户属性的修改动作

```
命令格式：
    usermod [options] LOGIN

常见选项
    -g|--gid GROUP #修改组
    -a|--append GROUP #将用户追加至上边 -G 中提到的附加组中，并不从其它组中删除此用户
    -L|--lock #锁定用户帐号，在/etc/shadow 密码栏的增加 !
    -l|--login LOGIN #新的登录名称

一般选项
    -c|--comment COMMENT #修改注释
    -d|--home HOME_DIR #修改家目录
    -e|--expiredate EXPIRE_DATE #修改过期的日期，YYYY-MM-DD 格式
    -f|--inactive INACTIVE #密码过期之后，账户被彻底禁用之前的天数，0 表示密码过期立即禁
用，-1表示不使用此功能
    -G|--groups GROUPS #groupName|GID... 新附加组，原来的附加组将会被覆盖；若保留原有，则
要同时使用-a选项
    -L|--lock #锁定用户帐号，在/etc/shadow 密码栏的增加 !
    -m|--move-home #将家目录内容移至新位置，和 -d 一起使用
    -o|--non-unique #允许使用重复的(非唯一的) UID
    -p|--password PASSWORD #修改密码，这里是明文，如果要在此处修改密码，则要用加密后的字符
串
    -s|--shell SHELL #修改 shell
    -u|--uid UID #修改 UID
    -U|--unlock #解锁用户帐号，将 /etc/shadow 密码栏的!拿掉
```

命令实践

登录用户名改名

```
[root@rocky9 ~]# id nihao
用户id=1001(nihao) 组id=1001(nihao) 组=1001(nihao)

更改登录用户为nihao2
[root@rocky9 ~]# usermod -c "nihao to nihao2" -l nihao2 nihao
[root@rocky9 ~]# id nihao
id: “nihao”：无此用户
[root@rocky9 ~]# id nihao2
用户id=1001(nihao2) 组id=1001(nihao) 组=1001(nihao)
```

```
确认用户家目录
[root@rocky9 ~]# ls /home/
nihao  remote  sswang  sswang1  sswang2  test1  user1
[root@rocky9 ~]# getent passwd nihao2
nihao2:x:1001:1001:nihao to nihao2:/home/nihao:/bin/bash
结果显示：
    nihao用户更改为了nihao2，但是用户家目录没有变动
```

更改用户其他信息

```
更改用户的登录shell 以及 用户家目录
[root@rocky9 ~]# usermod -s /bin/sh -d /home/nihao2 nihao2

确认效果
[root@rocky9 ~]# getent passwd nihao2
nihao2:x:1001:1001:nihao to nihao2:/home/nihao2:/bin/sh
```

```
默认情况下，即使我们使用 -d 修改了用户的家目录，但是名称不会更改
[root@rocky9 ~]# ls /home/
nihao  remote  sswang  sswang1  sswang2  test1  user1

确认效果
[root@rocky9 ~]# su nihao2
sh-5.1$ echo $HOME                  # 查看家目录环境变量
/home/nihao2
sh-5.1$ pwd                         # 查看当前所在位置
/root
sh-5.1$ cd                          # 回到家目录，提示没有家目录
sh: cd: /home/nihao2: 没有那个文件或目录
sh-5.1$ exit
exit
```

禁用用户实践

```
尝试使用nihao2进行账户登录
root@sswang:~# ssh nihao2@10.0.0.13
nihao2@10.0.0.13's password:
Last login: Wed Sep 25 10:32:21 2024 from 10.0.0.12
Could not chdir to home directory /home/nihao2: No such file or directory
[nihao2@rocky9 /]$
结果显示：
    可以通过 nihao2登录到10.0.0.12主机
```

```
修改nihao2用户禁用登录
[root@rocky9 ~]# usermod -L nihao2

确认效果
[root@rocky9 ~]# getent shadow nihao2
nihao2:!$6$ReJ2PO8ZEONu44XG$gRDh7wntpj6.IC7vVprzgrfsILW1Wz1G8KdwfkyHuAXd7gtkOWYv
ukOzXMEOBFAdLwsNOpI.syO1gAGjkSuGm.:19991:0:99999:7:::
结果显示：
    在用户密码前面多了一个 !
```

```
ssh远程登录测试
root@sswang:~# ssh nihao2@10.0.0.13
nihao2@10.0.0.13's password:
Permission denied, please try again.
nihao2@10.0.0.13's password:
Permission denied, please try again.
nihao2@10.0.0.13's password:
结果显示：登录失败
```

撤销用户登录锁定

```
恢复登录测试
[root@rocky9 ~]# usermod -U nihao2
[root@rocky9 ~]# getent shadow nihao2
nihao2:$6$ReJ2PO8ZEONu44XG$gRDh7wntpj6.IC7vVprzgrfsILW1Wz1G8KdwfkyHuAXd7gtkOWYvu
kOzXMEOBFAdLwsNOpI.syO1gAGjkSuGm.:19991:0:99999:7:::

ssh远程登录测试
nihao2@10.0.0.13's password:
Last failed login: Wed Sep 25 10:36:34 CEST 2024 from 10.0.0.12 on ssh:notty
There were 3 failed login attempts since the last successful login.
Last login: Wed Sep 25 10:34:04 2024 from 10.0.0.12
Could not chdir to home directory /home/nihao2: No such file or directory
[nihao2@rocky9 /]$
结果显示：
    用户登录成功了
```

用户空密码设定

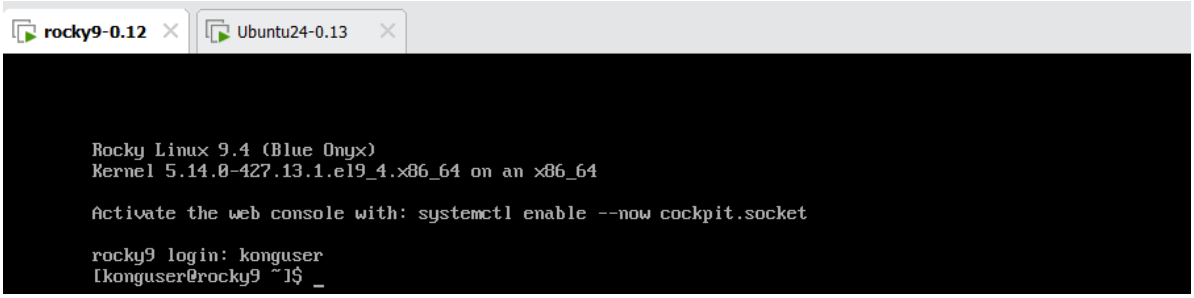
```
创建一个用户
[root@rocky9 ~]# useradd konguser

查看效果
[root@rocky9 ~]# getent shadow konguser
konguser:!!:19991:0:99999:7:::
此处的konguser的密码位置是 !!
```

```
手动修改shadow文件，取消!!
[root@rocky9 ~]# vim /etc/shadow
[root@rocky9 ~]# getent shadow konguser
konguser::19991:0:99999:7:::
注意：
    我们修改的文件，仅仅是将 两个!! 取消掉了
```

```
使用ssh登录失失败的
root@sswang:~# ssh konguser@10.0.0.13
konguser@10.0.0.13's password:

但是我们在终端里面可以直接用konguser登录
```



1.2.6 用户删除

命令解析

userdel 可以对linux系统用户进行删除操作

```
命令格式：
userdel [options] LOGIN

常见选项
    -f|--force #强制删除，哪怕用户正在登录状态
    -r|--remove #删除家目录和邮件目录
```

```
注意：
    正在使用的用户无法删除
```

命令实践

删除正在使用的用户

```
查看正在使用的用户
[root@rocky9 ~]# who
root     pts/0        2024-09-25 08:55 (10.0.0.1)
root     pts/1        2024-09-25 09:42 (10.0.0.1)
konguser tty5         2024-09-25 10:43

删除正在使用的用户
[root@rocky9 ~]# userdel konguser
userdel: user konguser is currently used by process 3313
```

正常删除用户-不清理家目录

```
删除正常用户 user1
[root@rocky9 ~]# userdel user1

结果user1 的家目录仍然存在
[root@rocky9 ~]# ls /home/
konguser  nihao  nihao3  remote  sswang  sswang1  sswang2  test1  user1
```

正常删除用户-清理家目录

```
删除正常用户 test1
[root@rocky9 ~]# userdel -r test1

test1 的家目录已经被删除
[root@rocky9 ~]# ls /home/
konguser  nihao  nihao3  remote  sswang  sswang1  sswang2  user1
```

强制删除用户-即使正在使用

```
查看登录用户
[root@rocky9 ~]# who
root     pts/0        2024-09-25 08:55 (10.0.0.1)
root     pts/1        2024-09-25 09:42 (10.0.0.1)
konguser tty5         2024-09-25 10:43

强制删除用户
[root@rocky9 ~]# userdel -f -r konguser
userdel: user konguser is currently used by process 3313

检查效果
[root@rocky9 ~]# id konguser
id: “konguser”：无此用户
[root@rocky9 ~]# ls /home/
nihao  nihao3  remote  sswang  sswang1  sswang2  user1

被删除的用户，还在使用
[root@rocky9 ~]# who
root     pts/0        2024-09-25 08:55 (10.0.0.1)
root     pts/1        2024-09-25 09:42 (10.0.0.1)
konguser tty5         2024-09-25 10:43
注意：
    只有该用户退出使用后，才会真正的消失
```

1.2.7 用户切换

基础知识

用户切换

```
    在Linux系统中，用户切换是一个常见的操作，它允许当前用户暂时放下自己的权限和工作环境，转而以另一个用户的身份进行操作。
    这种机制是Linux多用户环境的核心特性之一，对于系统管理员来说尤其重要，因为它允许他们在不同用户账户之间切换，以执行需要特定权限的任务。
```

用户身份

```
    如果在当前登录终端中，要执行某条命令，但当前登录用户又没有可执行权限或没有某些资源权限；则在此种情况下，我们可以：
1. 让有权限的用户登录终端，再执行相应的操作；
2. 在当前终端终，临时切换，以有权限的用户的身份去执行命令；
```

```
关于 切换 用户身份的命令就是 su ，即 switch user，以指定用户的身份执行相关命令
```

命令解析

```
    su命令是最常用的用户切换命令之一。它代表“substitute user”或“switch user”的缩写。它主要有两种使用方式：
    样式1：su 用户名
        - 非登录式切换，即会读取目标用户的配置文件，不改变当前工作目录，即不完全切换
        - 如果不加用户名，表示切换到root用户
    样式2：su - 用户名
        - 登录式切换，会读取目标用户的配置文件，切换至自己的家目录，即完全切换
        - 表示启动一个新的shell进行登录，然后使用 指定用户的环境信息
```

```
注意：
    从 root用户 su切换至其他用户无须密码
    从 非root用户 su切换至其他用户时需要密码
    su 切换新用户后，使用 exit 退回至旧的用户身份，而不要再用 su 切换至旧用户。
    如果普通用户是 /bin/false 或者 /sbin/nologin | /usr/sbin/nologin 的时候，无法切换
        但是，如果通过 -s 指定 /bin/bash，可以登录，无论是rocky还是ubuntu
```

命令实践

从 root用户su切换到任意普通用户

```
切换前查看效果
[root@rocky9 ~]# pwd
/root
[root@rocky9 ~]# echo $PATH
/root/.local/bin:/root/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
[root@rocky9 ~]# echo $HOME
/root

切换到sswang1普通用户
[root@rocky9 ~]# su sswang1

查看切换后的效果
[root@rocky9 ~]# su sswang1
bash-5.1$ echo $HOME
/home/sswang1
bash-5.1$ pwd
/root
bash-5.1$ echo $PATH
/root/.local/bin:/root/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
bash-5.1$ exit
exit
[root@rocky9 ~]#
结果显示：
    su 不加 - 切换用户，相当于，把普通用户的身份，直接穿到了root的身上，但是没有做到完全更改
```

从 root用户su - 切换到任意普通用户

```
切换到sswang1普通用户
[root@rocky9 ~]# su - sswang

查看相关的信息
[sswang@rocky9 ~]$ echo $HOME
```

```
/home/sswang
[sswang@rocky9 ~]$ pwd
/home/sswang
[sswang@rocky9 ~]$ echo $PATH
/home/sswang/.local/bin:/home/sswang/bin:/usr/local/bin:/usr/bin:/usr/local/sbin
:/usr/sbin
[sswang@rocky9 ~]$ exit
注销
结果显示：
    所有的信息都是sswang 自己的属性信息，
```

普通用户身份切换

```
从 sswang用户 切换到 sswang1 用户
[sswang@rocky9 ~]$ su - sswang1
密码：                          # 此处需要输入密码

查看效果
[sswang1@rocky9 ~]$ id
用户id=1024(sswang1) 组id=1024(sswang1) 组=1024(sswang1)
[sswang1@rocky9 ~]$ pwd
/home/sswang1
[sswang1@rocky9 ~]$ echo $PATH
/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin

第一次 exit回到上一级用户， 也就是sswang
[sswang1@rocky9 ~]$ exit
注销

第二次 exit回到上一级用户，也就是root
[sswang@rocky9 ~]$ exit
注销
[root@rocky9 ~]#
```

非登录用户切换

```
查找一个不能登录用户
[root@rocky9 ~]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
...

切换测试
[root@rocky9 ~]# su - daemon
This account is currently not available.

但是指定shell可以实现切换
[root@rocky9 ~]# su - daemon -s /bin/bash
[daemon@rocky9 ~]$ id
用户id=2(daemon) 组id=2(daemon) 组=2(daemon)
[daemon@rocky9 ~]$ exit
注销
```

1.2.8 密码策略

基础知识

chage 可以修改用户密码策略

```
命令格式
    chage [options] LOGIN

一般选项
    -d LAST_DAY #更改密码的时间
    -m|--mindays MIN_DAYS
    -M|--maxdays MAX_DAYS
    -W|--warndays WARN_DAYS
    -I|--inactive INACTIVE #密码过期后的宽限期
    -E|--expiredate EXPIRE_DATE #用户的有效期
    -l #显示密码策略
```

简单实践

查看当前的密码策略

```
[root@rocky9 ~]# chage -l sswang
最近一次密码修改时间              ：从不
密码过期时间                      ：从不
密码失效时间                      ：从不
帐户过期时间                      ：从不
两次改变密码之间相距的最小天数    ：0
两次改变密码之间相距的最大天数    ：99999
在密码过期之前警告的天数          ：7
```

修改默认的密码策略

```
修改密码策略
[root@rocky9 ~]# chage sswang
Changing the aging information for sswang
Enter the new value, or press ENTER for the default

        Minimum Password Age [0]:
        Maximum Password Age [99999]: 10            # 最大使用时间10天
        Last Password Change (YYYY-MM-DD) [2024-09-25]: 2024-09-11   # 故意将修改日
期超时
        Password Expiration Warning [7]:
        Password Inactive [-1]:
        Account Expiration Date (YYYY-MM-DD) [-1]:
```

```
查看修改后的密码策略
[root@rocky9 ~]# chage -l sswang
Last password change                                    : Sep 11, 2024
Password expires                                        : Sep 21, 2024
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 10
Number of days of warning before password expires       : 7
```

```
刚才看到，密码已经过期了，所以再次登录的话，就会出现问题
[root@rocky9 ~]# su - sswang
You are required to change your password immediately (password expired).
Changing password for sswang.
Current password:
New password:
Retype new password:
```

1.2.9 用户组管理2

基础知识

gpasswd更改组成员和密码 - 站在用户的角度

```
命令格式：
gpasswd [option] GROUP

一般选项：
    -a|--add USER #向组中添加用户
    -d|--delete USER #从组中移除用户
    -r|--delete-password #删除组密码
    -R|--restrict #向其成员限制访问组 GROUP
    -M|--members USER,... #批量加组
    -A|--administrators ADMIN,... #批量设组管理员

注意：
    组没有密码的情况下，加组只能由root来操作
```

groupmems 可以管理附加组的成员关系-站在用户组的角度

```
命令格式：
    groupmems [options] [action]

一般选项
    -g|--group groupname #更改为指定组（只有root）
    -a|--add username #指定用户加入组
    -d|--delete username #从组中删除用户
    -p|--purge #从组中清除所有成员
    -l|--list #显示组成员列表
```

groups 可查看用户组关系

```
命令格式：
    groups [OPTION].[USERNAME]...
```

简单实践

将用户加入到一个小组

```
查看当前用户和组信息
[root@rocky9 ~]# id sswang
用户id=1000(sswang) 组id=1000(sswang) 组=1000(sswang),10(wheel)
[root@rocky9 ~]# groups sswang
sswang : sswang wheel

添加一个新的组
[root@rocky9 ~]# groupadd web
[root@rocky9 ~]# getent gshadow web
web:!::

将sswang加入到web组
[root@rocky9 ~]# gpasswd -a sswang web
正在将用户“sswang”加入到“web”组中

确认效果
[root@rocky9 ~]# getent gshadow web
web:!::sswang
[root@rocky9 ~]# id sswang
用户id=1000(sswang) 组id=1000(sswang) 组=1000(sswang),10(wheel),10087(web)
结果显示：
    将sswang加入到了一个小组
```

为组添加密码

```
为组添加密码
[root@rocky9 ~]# gpasswd web
正在修改 web 组的密码
新密码：
请重新输入新密码：

确认效果
[root@rocky9 ~]# getent gshadow web
web:$6$Yn38J2xYaxPJiqsq$GOLKqbtMg9BHqeXcROEzHgOWbsEIle89YTA45MXNzEOcWaZOIbLYxzSv
6RBr75H23Edi/xNizvMl.use7v1Vt1::sswang
```

从组中移出用户

```
从组里面移除用户
[root@rocky9 ~]# gpasswd -d sswang web
正在将用户“sswang”从“web”组中删除

查看效果
[root@rocky9 ~]# groups sswang
sswang : sswang wheel
[root@rocky9 ~]# id sswang
用户id=1000(sswang) 组id=1000(sswang) 组=1000(sswang),10(wheel)
```

维护组关系

```
查看web组有哪些用户
[root@rocky9 ~]# groupmems -g web -l

为web组添加用户
[root@rocky9 ~]# groupmems -g web -a sswang

再次检测
[root@rocky9 ~]# groupmems -g web -l
sswang
[root@rocky9 ~]# id sswang
用户id=1000(sswang) 组id=1000(sswang) 组=1000(sswang),10(wheel),10087(web)
```

```
从组里面删除用户
[root@rocky9 ~]# groupmems -g web -d sswang
[root@rocky9 ~]# groupmems -g web -l
[root@rocky9 ~]# id sswang
用户id=1000(sswang) 组id=1000(sswang) 组=1000(sswang),10(wheel)
```

1.2.10 其他配置

用户登录配置文件 /etc/login.defs

```
[root@rocky9 ~]# grep -Ev '#|^$' /etc/login.defs
MAIL_DIR        /var/spool/mail
UMASK           022
HOME_MODE       0700
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7
UID_MIN                  1000
UID_MAX                 60000
SYS_UID_MIN               201
SYS_UID_MAX               999
SUB_UID_MIN            100000
SUB_UID_MAX         600100000
SUB_UID_COUNT           65536
GID_MIN                  1000
GID_MAX                 60000
SYS_GID_MIN               201
SYS_GID_MAX               999
SUB_GID_MIN            100000
```

```
SUB_GID_MAX             600100000
SUB_GID_COUNT               65536
ENCRYPT_METHOD SHA512
USERGROUPS_ENAB yes
CREATE_HOME     yes
HMAC_CRYPTO_ALGO SHA512
```

查看创建用户的配置文件 /etc/default/useradd

```
[root@rocky9 ~]# cat /etc/default/useradd
# useradd defaults file
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes
```

定制新用户基本配置和属性

```
[root@rocky9 ~]# ls /etc/skel/

注意：
    该目录保存了创建新用户后需要拷贝到 /home 目录下所需的相关文件。
```

定制用户登录的显示页面

```
用于设置用户登录系统后显示给用户的提示信息，该文件默认为空。
[root@rocky9 ~]# cat /etc/motd
```

面试题：我在创建用户的时候，如何为新创建的用户，提供一个通用的提示文件

```
在 /etc/skel/ 目录下提供一个提示文件，这样每个新创建用户，都会在自动包含该文件。
```

1.3 权限管理

1.3.1 权限体系

基础知识

简介

```
    Linux系统下的文件权限体系是控制对文件和目录访问权限的重要机制，它决定了谁可以读取、修改或执行文件。这一体系是Linux系统安全性和数据保护的基础。
    在linux系统中，文件的权限，主要体现在两个方面：文件属性、操作权限
```

文件操作权限

Linux文件权限主要分为三种类型：
- 读权限（Readable）：
  - 允许用户读取文件内容 或 查看目录中的文件列表。
  - 简写：r、4
- 写权限（Writable）：
  - 允许用户修改文件内容 或 在目录中创建、删除或重命名文件。
  - 简写：w、2
- 执行权限（eXcutable）：
  - 对于文件，表示该文件是可执行的程序；对于目录，表示用户可以进入该目录。
  - 简写：x、1

文件用户属性

文件权限被分为三个级别，分别对应不同的用户群体：
- 所有者（Owner）：
  - 文件或目录的创建者，拥有对文件或目录的最高权限。
  - 简写：u
- 所属组（Group）：
  - 文件或目录被分配到的用户组，组内的所有成员将继承该组对该文件或目录的权限。
  - 简写：g
- 其他用户（Others）：
  - 既不是文件所有者也不是所属组成员的所有其他用户。
  - 简写：o

程序访问文件时的权限，取决于此程序的发起者
- 进程的发起者，同文件的属主：则应用文件属主权限
- 进程的发起者，属于文件属组：则应用文件属组权限
- 应用文件“其它”权限

文件权限表示示例

字符表示	二进制表示	八进制表示	备注
---	000	0	无任何权限
--x	001	1	可执行
-w-	010	2	可写
-wx	011	3	可写可执行
r--	100	4	可读
r-x	101	5	可读可执行
rw-	110	6	可读可写
rwx	111	7	可读可写可执行

字符表示	八进制数字表示	备注
rw-r-----	640	属主可读写，属组可读，其它用户无任何权限
rw-r--r--	644	属主可读写，属组可读，其它用户可读
rwxr-xr-x	755	属主可读写，可执行，属组可读可执行，其它用户可读可执行

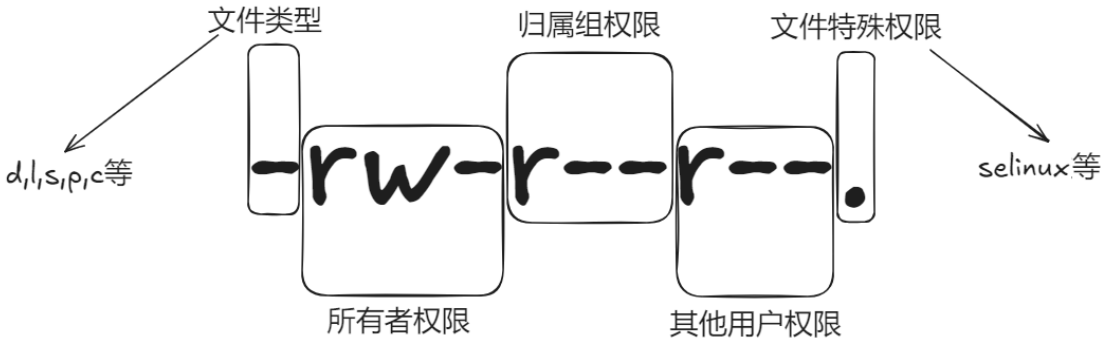
```
注意：
    默认情况下
        - 创建的目录权限是 755
        - 创建的文件权限是 644
```

简单实践

查看文件属性信息

```
查看文件
[root@rocky9 ~]# ls /etc/bashrc  -l
-rw-r--r--. 1 root root 2696  9月 23 06:13 /etc/bashrc
```

查看权限信息



关注点

```
r和w权限对root 用户无效，对没有读写权限的文件，root用户也可读可写
只要 所有者，所属组或other三者之一有x权限，root就可以执行
```

```
即使文件没有权限，但是对于超级用户root没有太多的意义
[root@rocky9 ~]# ll /etc/shadow
---------- 1 root root 1845  9月 25 12:04 /etc/shadow
[root@rocky9 ~]# ll /etc/gshadow
---------- 1 root root 886  9月 25 12:06 /etc/gshadow
```

1.3.2 属性操作

基础知识

chown 命令可以修改文件的属主，也可以修改文件属组

```
命令格式：
    chown [OPTION]... [OWNER][:[GROUP]] FILE...
    chown [OPTION]... --reference=RFILE FILE...
```

```
    注意：
        owner 表示所有者
        :group 和 .group 表示归属组
        owner. 和 owner: 表示所有者和归属组一样

常用选项
    -R|--recursive #递归操作


一般选项
    -c|--changes #同-v选项，但只显示更新成功的信息
    -f|--silent|--quiet #不显示错误信息
    -v|--verbose #显示过程
    --dereference #修改的是符号链接指向的文件，而不是链接文件本身
    -h|--no-dereference #    修改的是符号链接文件，而不是其指向的目标文件
                        #    只有当前系统支持修改符号链接文件属性时，此项才有效
    --from=user:group #根据原属主属组来修改，相当于一个查询条件
    --no-preserve-root # 不特别对待“/”，意思就是将根目录当成普通目录来执行
                       # 默认如此，所以不要对根目录进行操作
    --preserve-root     #不允许在"/"上递归操作
    --reference=RFILE #根据其它文件权限来操作，就是复制该文件的属主属组信息给指定文件

    下列选项配合 -R 使用
        -H #如果选项是指向目录的软链接，则只修改指向的目录，不改变目录里面文件的属主属组
        -L #更改所有遇到的符号链接指向的目录
        -P #不更改符号链接指向的目录
```

chgrp 命令可以只修改文件的属组

```
命令格式：
    chgrp [OPTION]... GROUP FILE...
    chgrp [OPTION]... --reference=RFILE FILE...

常用选项
    -R|--recursive #递归操作

一般选项
    参考 chown 的选项
```

chmod 命令可以修改文件的操作权限

```
命令格式
    chmod [OPTION]... MODE[,MODE]... FILE...
    chmod [OPTION]... OCTAL-MODE FILE...
    chmod [OPTION]... --reference=RFILE FILE...

常用选项
    -R|--recursive #递归操作
一般选项
    -c|--changes #同-v选项，但只显示更新成功的信息
    -f|--silent|--quiet #不显示错误信息
    -v|--verbose #显示过程
    --no-preserve-root #不特别对待“/”，意思就是将家目录当成普通目录来执行，默认如此，所以
    不要对根目录进行操作
    --preserve-root #不允许在"/"上递归操作
    --reference=RFILE #根据其它文件权限来操作,就是复制该文件的权限信息给指定文件
```

权限的增删

```
linux系统中，对于文件的属性和权限的操作，常见的动作主要有以下三种
    + 增加某些权限
    - 删除某些权限
    = 只保留某些权限，覆盖写法
```

```
操作示例：
    u+r             # 属主加读权限
    g-x             # 属组去掉执行权限
    ug=rx           # 属主属组权限改为读和执行
    o=              # other用户无任何权限
    a=rwx           # 所有用户都有读写执行权限
    u+r,g-x         # 同时指定
```

简单实践

准备工作

```
创建相关文件
[root@rocky9 ~]# mkdir chown
[root@rocky9 ~]# mkdir chown/dir{1..5}
[root@rocky9 ~]# echo file-a > chown/a.txt
[root@rocky9 ~]# echo file-b > chown/b.txt
[root@rocky9 ~]# echo file-c > chown/c.txt
[root@rocky9 ~]# echo file-d > chown/d.txt
[root@rocky9 ~]# echo file-e > chown/e.txt

查看文件权限
[root@rocky9 ~]# ll chown/
总用量 0
-rw-r--r-- 1 root root 0  9月 25 12:49 a.txt
-rw-r--r-- 1 root root 0  9月 25 12:49 b.txt
-rw-r--r-- 1 root root 0  9月 25 12:49 c.txt
drwxr-xr-x 2 root root 6  9月 25 12:48 dir1
drwxr-xr-x 2 root root 6  9月 25 12:48 dir2
drwxr-xr-x 2 root root 6  9月 25 12:48 dir3
drwxr-xr-x 2 root root 6  9月 25 12:48 dir4
drwxr-xr-x 2 root root 6  9月 25 12:48 dir5
-rw-r--r-- 1 root root 0  9月 25 12:49 d.txt
-rw-r--r-- 1 root root 0  9月 25 12:49 e.txt
结果显示：
    默认创建的文件权限是 644，也就是所，谁都可以读去文件内容
    默认创建的目录权限是 755，也就是说，谁都可以进入目录查看
```

更改文件用户属性

```
只修改所有者属性
[root@rocky9 ~]# chown sswang chown/a.txt

同时修改所有者和归属组的属性
[root@rocky9 ~]# chown sswang. chown/b.txt          # . 的这种方式，虽然管用，但是不推荐
[root@rocky9 ~]# chown sswang: chown/c.txt

查看效果
[root@rocky9 ~]# ll chown/{a..c}.txt
-rw-r--r-- 1 sswang root   7  9月 25 12:50 chown/a.txt
-rw-r--r-- 1 sswang sswang 7  9月 25 12:50 chown/b.txt
-rw-r--r-- 1 sswang sswang 7  9月 25 12:50 chown/c.txt
```

```
通过用户id的方式修改文件归属
[root@rocky9 ~]# id sswang1
用户id=1024(sswang1) 组id=1024(sswang1) 组=1024(sswang1)
[root@rocky9 ~]# chown 1024 chown/d.txt
[root@rocky9 ~]# ll chown/d.txt
-rw-r--r-- 1 sswang1 root 7  9月 25 12:50 chown/d.txt
```

```
仅修改文件的归属组的属性
[root@rocky9 ~]# chown .sswang1 chown/d.txt
[root@rocky9 ~]# ll chown/{d..e}.txt
-rw-r--r-- 1 sswang1 sswang1 7  9月 25 12:50 chown/d.txt
-rw-r--r-- 1 root    sswang1 7  9月 25 12:50 chown/e.txt
```

批量修改文件属性信息

```
[root@rocky9 ~]# chown -R sswang2 chown
[root@rocky9 ~]# ll
总用量 8
drwxr-xr-x  7 sswang2 root 131  9月 25 12:49 chown
[root@rocky9 ~]# ll chown/
总用量 20
-rw-r--r-- 1 sswang2 root    7  9月 25 12:50 a.txt
-rw-r--r-- 1 sswang2 sswang  7  9月 25 12:50 b.txt
-rw-r--r-- 1 sswang2 sswang  7  9月 25 12:50 c.txt
drwxr-xr-x 2 sswang2 sswang1 6  9月 25 12:48 dir1
drwxr-xr-x 2 sswang2 sswang1 6  9月 25 12:48 dir2
drwxr-xr-x 2 sswang2 root    6  9月 25 12:48 dir3
drwxr-xr-x 2 sswang2 root    6  9月 25 12:48 dir4
drwxr-xr-x 2 sswang2 root    6  9月 25 12:48 dir5
-rw-r--r-- 1 sswang2 sswang1 7  9月 25 12:50 d.txt
-rw-r--r-- 1 sswang2 sswang1 7  9月 25 12:50 e.txt
```

归属组信息设置

```
修改归属组信息
[root@rocky9 ~]# chgrp -R web chown/

检查效果
[root@rocky9 ~]# ll
总用量 8
```

```
drwxr-xr-x  7 sswang2 web  131  9月 25 12:49 chown
[root@rocky9 ~]# ll chown/
总用量 20
-rw-r--r-- 1 sswang2 web 7  9月 25 12:50 a.txt
-rw-r--r-- 1 sswang2 web 7  9月 25 12:50 b.txt
-rw-r--r-- 1 sswang2 web 7  9月 25 12:50 c.txt
drwxr-xr-x 2 sswang2 web 6  9月 25 12:48 dir1
drwxr-xr-x 2 sswang2 web 6  9月 25 12:48 dir2
drwxr-xr-x 2 sswang2 web 6  9月 25 12:48 dir3
drwxr-xr-x 2 sswang2 web 6  9月 25 12:48 dir4
drwxr-xr-x 2 sswang2 web 6  9月 25 12:48 dir5
-rw-r--r-- 1 sswang2 web 7  9月 25 12:50 d.txt
-rw-r--r-- 1 sswang2 web 7  9月 25 12:50 e.txt
```

文件权限修改实践

```
指定文件的其他用户都增加x权限
[root@rocky9 ~]# chmod o+x chown/{a..e}.txt
[root@rocky9 ~]# ll chown/{a..e}.txt
-rw-r--r-x 1 sswang2 web 7  9月 25 12:50 chown/a.txt
-rw-r--r-x 1 sswang2 web 7  9月 25 12:50 chown/b.txt
-rw-r--r-x 1 sswang2 web 7  9月 25 12:50 chown/c.txt
-rw-r--r-x 1 sswang2 web 7  9月 25 12:50 chown/d.txt
-rw-r--r-x 1 sswang2 web 7  9月 25 12:50 chown/e.txt
```

```
指定文件的 归属组权限+x，其他用户权限-x
[root@rocky9 ~]# chmod g+x,o-x chown/{a..e}.txt
[root@rocky9 ~]# ll chown/{a..e}.txt
-rw-r-xr-- 1 sswang2 web 7  9月 25 12:50 chown/a.txt
-rw-r-xr-- 1 sswang2 web 7  9月 25 12:50 chown/b.txt
-rw-r-xr-- 1 sswang2 web 7  9月 25 12:50 chown/c.txt
-rw-r-xr-- 1 sswang2 web 7  9月 25 12:50 chown/d.txt
-rw-r-xr-- 1 sswang2 web 7  9月 25 12:50 chown/e.txt
```

```
指定文件的 所有者权限为 rwx
[root@rocky9 ~]# chmod u=rwx chown/{a..e}.txt
[root@rocky9 ~]# ll chown/{a..e}.txt
-rwxr-xr-- 1 sswang2 web 7  9月 25 12:50 chown/a.txt
-rwxr-xr-- 1 sswang2 web 7  9月 25 12:50 chown/b.txt
-rwxr-xr-- 1 sswang2 web 7  9月 25 12:50 chown/c.txt
-rwxr-xr-- 1 sswang2 web 7  9月 25 12:50 chown/d.txt
-rwxr-xr-- 1 sswang2 web 7  9月 25 12:50 chown/e.txt
```

```
指定文件的 仅有其他人员才具有w权限
[root@rocky9 ~]# chmod 002 chown/{a..e}.txt
[root@rocky9 ~]# ll chown/{a..e}.txt
--------w- 1 sswang2 web 7  9月 25 12:50 chown/a.txt
--------w- 1 sswang2 web 7  9月 25 12:50 chown/b.txt
--------w- 1 sswang2 web 7  9月 25 12:50 chown/c.txt
--------w- 1 sswang2 web 7  9月 25 12:50 chown/d.txt
--------w- 1 sswang2 web 7  9月 25 12:50 chown/e.txt
```

目录的写权限实践

```
准备目录
[root@rocky9 ~]# mkdir /chown/dir1 -p
[root@rocky9 ~]# ll /chown/
总用量 0
drwxr-xr-x 2 root root 6  9月 25 13:39 dir1

使用普通用户
[root@rocky9 ~]# su - sswang1
[sswang1@rocky9 ~]$ cd /chown/dir1/
[sswang1@rocky9 dir1]$ ls
```

```
修改权限后，检测效果
[root@rocky9 ~]# chmod 700 -R /chown/dir1/
[root@rocky9 ~]# su - sswang
[sswang@rocky9 ~]$ cd /chown/dir1/
-bash: cd: /chown/dir1/: 权限不够
```

```
更改权限
[root@rocky9 ~]# chown 006 -R /chown/dir1/
[root@rocky9 ~]# ll /chown/
总用量 0
d------rw- 2 root root 6  9月 25 13:39 dir1

检查效果 - 因为没有x权限，进不去
[root@rocky9 ~]# su - sswang1
[sswang1@rocky9 ~]$ cd /chown/dir1/
-bash: cd: /chown/dir1/: 权限不够
```

```
为目录添加x权限
[root@rocky9 ~]# chmod 005 -R /chown/dir1/

再次测试效果
[root@rocky9 ~]# su - sswang1
[sswang1@rocky9 ~]$ cd /chown/dir1/
[sswang@rocky9 dir1]$ ls
[sswang@rocky9 dir1]$ echo nihao > xxx.txt
-bash: xxx.txt: 权限不够
结果显示:
    目录可以进入，但是不能创建文件
```

面试题:

```
Linux中的目录和文件的权限区别？分别说明读，写和执行权限的区别
```

1.3.3 默认权限

基础知识

默认权限

linux系统在创建文件的时候，会有一个默认的权限。普通文件的权限是 644，普通目录文件的权限是755。在Linux系统中，文件创建时的默认权限与umask（用户文件创建掩码）紧密相关。umask决定了在创建新文件或目录时哪些权限位将被屏蔽（即不允许），从而间接决定了新文件或目录的默认权限。

umask简介

umask是用户文件创建掩码（User Mask）的缩写，它是一个八进制数值，用于指定在创建新文件或目录时需要屏蔽的权限位。umask的值决定了新创建的文件或目录的默认权限。

默认情况下，Linux系统中的umask值通常为022。这个值意味着：
对于新创建的文件，默认权限为644（即所有者具有读写权限，所属组和其他用户只有读权限）。
对于新创建的目录，默认权限为755（即所有者具有读、写、执行权限，所属组和其他用户具有读和执行权限）

计算公式

umask的值是从文件或目录的默认权限中减去得到的。Linux中，文件和目录的默认权限分别是：
文件：-rw-rw-rw-（即666，所有用户都具有读写权限,不需要执行）
目录：drwxrwxrwx（即777，所有用户都具有读、写、执行权限）

当创建新文件或目录时，实际权限是默认权限减去umask值后的结果。例如，
如果umask值为022，
- 则新文件的权限为666 - 022 = 644，
- 新目录的权限为777 - 022 = 755。

修改方法

如果需要更改默认权限，可以通过修改umask值来实现。这可以通过以下几种方式完成：

临时修改：
在命令行中使用umask命令后跟新的umask值来临时修改umask值。例如，umask 000会将umask值设置为000，但这种方式只在当前会话中有效，重启后失效。

永久修改：
要将umask值的更改永久生效，可以将新的umask值添加到用户的登录shell配置文件中（如.bashrc、.profile等）。这样，每次用户登录时都会自动应用新的umask值。

命令解读

命令格式：

```
umask [-p] [-S] [mode]
```

选项

```
-p #如果省略 MODE 模式，以可重用为输入的格式输入
-S #以字符显示
```

简单实践

查看umask

```
[root@rocky9 tmp]# umask
0022
[root@rocky9 tmp]# umask -p
umask 0022
[root@rocky9 tmp]# umask -S
u=rwx,g=rx,o=rx
```

设定umask实践

```
设定umask
[root@rocky9 tmp]# umask 123
[root@rocky9 tmp]# umask
0123

检测效果
[root@rocky9 tmp]# mkdir udir
[root@rocky9 tmp]# touch ufile
[root@rocky9 tmp]# ll
总用量 0
drw-r-xr-- 2 root root 6  9月 25 16:21 udir
-rw-r--r-- 1 root root 0  9月 25 16:21 ufile
```

临时使用

```
还原umask
[root@rocky9 tmp]# umask 022
[root@rocky9 tmp]# umask
0022

子shell创建文件
[root@rocky9 tmp]# ( umask 666; touch lsfile.txt )
当前shell创建文件
[root@rocky9 tmp]# touch dfile.txt

查看效果，
[root@rocky9 tmp]# ll
总用量 0
-rw-r--r-- 1 root root 0  9月 25 16:23 dfile.txt
---------- 1 root root 0  9月 25 16:22 lsfile.txt
drw-r-xr-- 2 root root 6  9月 25 16:21 udir
-rw-r--r-- 1 root root 0  9月 25 16:21 ufile
结果显示：
    互不影响。
```

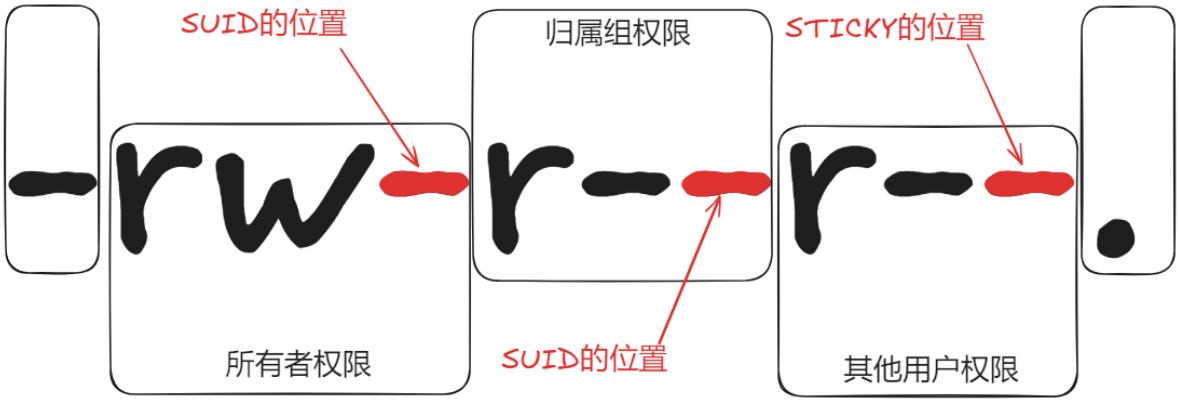
```
[root@rocky9 tmp]# mkdir dir1
[root@rocky9 tmp]# touch file
[root@rocky9 tmp]# ll
总用量 0
drwxr-xr-x 2 root root 6  9月 25 15:20 dir1
-rw-r--r-- 1 root root 0  9月 25 15:20 file
```

1.3.4 特殊权限

基础知识

特殊权限

> 在Linux文件系统中，除了基本的读（r）、写（w）、执行（x）权限外，还存在三种特殊权限，它们分别是SUID（Set User ID）、SGID（Set Group ID）和Sticky Bit。这些特殊权限用于在特定情况下提供更灵活的权限控制。



SUID（Set User ID）

> 功能：
>
> 当SUID权限设置在可执行文件上时，执行该文件的用户将暂时获得该文件所有者的权限。这意味着，即使执行者不是文件的所有者，他们也能以文件所有者的身份执行文件，并可能因此访问或修改只有文件所有者才能访问的资源。
>
> 表现：
>
> 在ls -l命令的输出中，如果可执行文件的"所有者"执行ls -权限位是s（小写），则表示该文件具有SUID权限。如果该文件没有执行权限，则表现为S（大写）。

> 示例：
>
> ```
> root@sswang:~# ls -l /usr/bin/* | grep -E 'rws'
> -rwsr-xr-x 1 root root       72792  4月  9 15:01 /usr/bin/chfn
> -rwsr-xr-x 1 root root       44760  4月  9 15:01 /usr/bin/chsh
> -rwsr-xr-x 1 root root       39296  4月  8 23:57 /usr/bin/fusermount3
> -rwsr-xr-x 1 root root       76248  4月  9 15:01 /usr/bin/gpasswd
> -rwsr-xr-x 1 root root       51584  4月  9 22:02 /usr/bin/mount
> -rwsr-xr-x 1 root root       40664  4月  9 15:01 /usr/bin/newgrp
> -rwsr-xr-x 1 root root       64152  4月  9 15:01 /usr/bin/passwd
> -rwsr-xr-x 1 root root       30952  4月  4 02:26 /usr/bin/pkexec
> -rwsr-xr-x 1 root root       55680  4月  9 22:02 /usr/bin/su
> -rwsr-xr-x 1 root root      277936  4月  8 22:50 /usr/bin/sudo
> -rwsr-xr-x 1 root root       39296  4月  9 22:02 /usr/bin/umount
> ```
>
> 解读：
>
> 以/usr/bin/passwd文件为例，Linux系统中的/usr/bin/passwd文件就具有SUID权限。这使得普通用户能够修改自己的密码，因为passwd程序在执行时会暂时获得root权限，从而能够写入只有root才能访问的/etc/shadow文件。

SGID（Set Group ID）

功能：
    SGID权限可以应用于可执行文件或目录。对于可执行文件，它与执行SUID文件类似，但影响的是执行者的组身份。然而，在实际应用中，SGID更多地被用于目录，以便在该目录下创建的新文件自动继承目录的组身份。
表现：
    在ls -l命令的输出中，如果"目录的组"执行权限位是s（小写），则表示该目录具有SGID权限。

```
root@sswang:~# ls -l /usr/bin/* | grep -E '\-sr'
-rwxr-sr-x 1 root shadow     72184  4月  9 15:01 /usr/bin/chage
-rwxr-sr-x 1 root crontab    39664  3月 31 08:06 /usr/bin/crontab
-rwxr-sr-x 1 root shadow     27152  4月  9 15:01 /usr/bin/expiry
-rwxr-sr-x 1 root _ssh      309688  4月  6 03:30 /usr/bin/ssh-agent
```

解读：
    在多用户环境中，SGID目录可以确保在该目录下创建的所有文件都属于同一个组，从而方便组成员之间的文件共享和管理。

Sticky Bit

功能：
    Sticky Bit权限仅对目录有效。当一个目录被设置为Sticky Bit时，只有该目录的所有者、文件的所有者或root用户才能删除或重命名该目录下的文件。这有助于防止其他用户删除或移动他们不拥有但可能有权访问的文件。
表现：
    在ls -l命令的输出中，如果目录的"其他用户"执行权限位是t（小写），则表示该目录具有Sticky Bit权限。

```
root@sswang:~# ll / | grep tmp
drwxrwxrwt  18 root root       4096  9月 26 09:17 tmp/
```

解读：
    Sticky Bit通常用于需要多用户访问但又要限制删除权限的目录，

其他信息

权限	字符表示	八进制表示	备注
SUID	s(小写)	4	如果原属主没有可执行权限，再加SUID权限，则显示为S
SGID	s(小写)	2	如果原属组没有可执行权限，再加SGID权限，则显示为S
STICKY	t(小写)	1	如果other没有可执行权限，再加STICKY权限，则显示为T

设定方式

```
SUID设定方式: u+s、u-s、4xxx
SGID设定方式: g+s、g-s、2xxx
STICKY设定方式: o+t、o-t、1xxx
```

其他方式 -- 看看就得了，基本没用

字符表示	二进制表示	八进制表示	备注
---------	000	0	无任何特殊权限
--------t	001	1	Sticky
-----s---	010	2	SGID
-----s--t	011	3	SGID,Sticky
--s------	100	4	SUID
--s-----t	101	5	SUID,Sticky
--s---s---	110	6	SUID,SGID
--s--s--t	111	7	SUID,SGID,Sticky

1.3.5 SUID实践

SUID文件属性确认

```
准备源文件
[root@rocky9 ~]# ll /usr/bin/{cat,passwd,tail}
-rwxr-xr-x 1 root root 39384  4月  5 22:36 /usr/bin/cat*
-rwsr-xr-x 1 root root 64152  4月  9 15:01 /usr/bin/passwd*
-rwxr-xr-x 1 root root 64032  4月  5 22:36 /usr/bin/tail*

[root@rocky9 ~]# ll /etc/shadow
-rw-r----- 1 root shadow 1484  9月 26 10:17 /etc/shadow
```

```
[root@rocky9 ~]# ll /usr/bin/{cat,passwd,tail}
-rwxr-xr-x. 1 root root 36440  4月 20 20:43 /usr/bin/cat
-rwsr-xr-x. 1 root root 32656  5月 15  2022 /usr/bin/passwd
-rwxr-xr-x. 1 root root 69656  4月 20 20:43 /usr/bin/tail
[root@rocky9 ~]# ll /etc/shadow
---------- 1 root root 1845  9月 25 12:04 /etc/shadow
```

```
根据我们之前对于 文件属性和权限的了解，
    cat、passwd、tail 对于普通用户来说，也具有 x执行权限
    /etc/shadow 文件只有 root用户具有读写能力，shadow小组具有读权限，普通用户没有读取权限。
    对于passwd来说，它具有SUID的权限，也就是说，普通用户通过执行该文件，具有了操作
        - 原本普通用户没有资格编辑 /etc/shadow 的文件权限
```

普通用户尝试SUID实践

```
使用普通用户实践
[root@rocky9 ~]# su - sswang1
[sswang1@rocky9 ~]$ cat /etc/shadow
cat: /etc/shadow: 权限不够

[sswang1@rocky9 ~]$ passwd
更改用户 sswang1 的密码 。
当前的密码：
新的密码：                    # compre123
重新输入新的密码：            # compre123
passwd：所有的身份验证令牌已经成功更新。
```

root用户为cat命令添加suid属性

```
[root@rocky9 ~]# chmod u+s /usr/bin/cat
[root@rocky9 ~]# ll /usr/bin/cat
-rwsr-xr-x. 1 root root 36440  4月 20 20:43 /usr/bin/cat
结果显示：
    cat命令的拥有者具有了suid的权限属性
```

普通用户验证cat的suid属性

```
查看原本普通用户不能查看的文件
[sswang1@rocky9 ~]$ cat /etc/shadow
...
sswang1:$6$BIOG6pNgkD.3RkkX$n1i8NkU.di8Rk8LcjkjBaCp7fJYgSeNO33IUEwpSFKBUMM2TyrsY
cTSovGqUNMKQ6rXVO3fXt2cdMjgoBsuIW/:19992:0:99999:7:::

确认密码效果
[sswang1@rocky9 ~]$ openssl passwd -salt BIOG6pNgkD.3RkkX -6 compre123
$6$BIOG6pNgkD.3RkkX$n1i8NkU.di8Rk8LcjkjBaCp7fJYgSeNO33IUEwpSFKBUMM2TyrsYcTSovGqU
NMKQ6rXVO3fXt2cdMjgoBsuIW/
结果显示：
    一旦为 cat 赋予了suid属性，普通用户，就暂时具有了该命令所有者的一些属性
```

suid权限恢复

```
使用root用户，对suid权限进行复原
[root@rocky9 ~]# chmod u-s /usr/bin/cat
```

1.3.6 SGID实践

在/tmp 目录下创建文件

```
使用sswang1用户操作
[sswang1@rocky9 ~]$ mkdir /tmp/sswang1
[sswang1@rocky9 ~]$ touch /tmp/sswang1/sswang1.txt

使用root用户操作
[root@rocky9 ~]# touch /tmp/sswang1/root.txt
```

查看权限效果

```
[sswang1@rocky9 ~]$ ll /tmp/
总用量 0
drwxr-xr-x 2 sswang1 sswang1 41  9月 26 04:49 sswang1
[sswang1@rocky9 ~]$ ll /tmp/sswang1/
总用量 0
-rw-r--r-- 1 root    root    0  9月 26 04:48 root.txt
-rw-r--r-- 1 sswang1 sswang1 0  9月 26 04:49 sswang1.txt
结果显示：
    在/tmp 目录下执行文件的创建，文件的所有者、归属组，都是各自的用户身份
```

为 sswang1目录添加SGID属性

```
[root@rocky9 ~]# chmod g+s /tmp/sswang1
[root@rocky9 ~]# ll /tmp/
总用量 0
drwxr-sr-x 2 sswang1 sswang1 41  9月 26 04:49 sswang1
```

root用户再次创建文件

```
root用户执行相同的文件创建动作
[root@rocky9 ~]# touch /tmp/sswang1/root-sgid.txt
[root@rocky9 ~]# mkdir /tmp/sswang1/root
[root@rocky9 ~]# ll /tmp/sswang1/
总用量 0
drwxr-sr-x 2 root    sswang1 6  9月 26 04:54 root
-rw-r--r-- 1 root    sswang1 0  9月 26 04:51 root-sgid.txt
-rw-r--r-- 1 root    root    0  9月 26 04:48 root.txt
-rw-r--r-- 1 sswang1 sswang1 0  9月 26 04:49 sswang1.txt
结果显示：
    当 sswang1 目录具有了SGID的属性后，无论是哪个用户在 sswang1目录下创建文件，
        文件的归属组都是 sswang1 目录的数组属性
```

SGID属性恢复

```
[root@rocky9 ~]# chmod g-s /tmp/sswang1/
[root@rocky9 ~]# ll /tmp/
总用量 0
drwxr-xr-x 3 sswang1 sswang1 74  9月 26 04:54 sswang1
```

1.3.7 STICKY实践

准备工作

```
创建一个谁都可以为所谓欲的工作目录
[root@rocky9 ~]# mkdir /sticky/dir -p
[root@rocky9 ~]# chmod -R 777 /sticky           # 类似与 /tmp 目录
[root@rocky9 ~]# ll -a /sticky/
总用量 4
drwxrwxrwx   3 root root   17  9月 26 04:59 .
dr-xr-xr-x. 23 root root 4096  9月 26 04:59 ..
drwxrwxrwx   2 root root    6  9月 26 04:59 dir
```

使用普通用户执行操作

```
使用普通用户创建各自的文件
[root@rocky9 ~]# su - sswang
[sswang@rocky9 ~]$ touch /sticky/dir/sswang.txt

[root@rocky9 ~]# su - sswang1
[sswang1@rocky9 ~]$ touch /sticky/dir/sswang1.txt

[root@rocky9 ~]# su - sswang2
[sswang2@rocky9 ~]$ touch /sticky/dir/sswang2.txt
```

```
查看创建文件后的效果
[sswang2@rocky9 ~]$ ll /sticky/
总用量 0
drwxrwxrwx 2 root root 62  9月 26 05:02 dir
[sswang2@rocky9 ~]$ ll /sticky/dir/
总用量 0
-rw-r--r-- 1 sswang1 sswang1 0  9月 26 05:01 sswang1.txt
-rw-r--r-- 1 sswang2 sswang2 0  9月 26 05:02 sswang2.txt
-rw-r--r-- 1 sswang  sswang  0  9月 26 05:01 sswang.txt
结果显示：
    指定目录下的文件，用户属性都是各个用户的。
```

```
该目录下的所有文件，任何用户都可以为所欲为
[sswang2@rocky9 ~]$ rm -rf /sticky/dir/*
[sswang2@rocky9 ~]$ ll /sticky/dir/
总用量 0
```

STICKY属性实践

```
使用root用户为指定目录添加sticky属性
[root@rocky9 ~]# chmod o+t /sticky/dir
[root@rocky9 ~]# ll /sticky/
总用量 0
drwxrwxrwt 2 root root 6  9月 26 05:04 dir
[root@rocky9 ~]# file /sticky/dir/
/sticky/dir/: sticky, directory
```

```
使用多个用户在dir目录下创建文件
[root@rocky9 ~]# su - sswang
[sswang@rocky9 ~]$ touch /sticky/dir/sswang.txt

[root@rocky9 ~]# su - sswang1
[sswang1@rocky9 ~]$ touch /sticky/dir/sswang1.txt

[root@rocky9 ~]# su - sswang2
[sswang2@rocky9 ~]$ touch /sticky/dir/sswang2.txt
```

```
查看各自用户创建后的效果
[sswang2@rocky9 ~]$ ll /sticky/dir/
总用量 0
-rw-r--r-- 1 sswang1 sswang1 0  9月 26 05:07 sswang1.txt
-rw-r--r-- 1 sswang2 sswang2 0  9月 26 05:08 sswang2.txt
-rw-r--r-- 1 sswang  sswang  0  9月 26 05:07 sswang.txt
```

```
尝试用sswang2用户删除文件
[sswang2@rocky9 ~]$ rm -rf /sticky/dir/sswang.txt
rm: 无法删除 '/sticky/dir/sswang.txt': 不允许的操作
[sswang2@rocky9 ~]$ rm -rf /sticky/dir/sswang1.txt
rm: 无法删除 '/sticky/dir/sswang1.txt': 不允许的操作
[sswang2@rocky9 ~]$ rm -rf /sticky/dir/sswang2.txt
[sswang2@rocky9 ~]$ ll /sticky/dir/
总用量 0
-rw-r--r-- 1 sswang1 sswang1 0  9月 26 05:07 sswang1.txt
-rw-r--r-- 1 sswang  sswang  0  9月 26 05:07 sswang.txt
结果显示：
    一旦dir目录添加了STICKY属性，那么用户只能删除自己的文件
```

```
为 dir目录取消sticky属性
[root@rocky9 ~]# chmod o-t /sticky/dir
[root@rocky9 ~]# ll /sticky/
总用量 0
drwxrwxrwx 2 root root 43  9月 26 05:09 dir
```

```
再次尝试 sswang2 用户删除不属于自己的文件
[root@rocky9 ~]# su - sswang2
[sswang2@rocky9 ~]$ rm -rf /sticky/dir/*
[sswang2@rocky9 ~]$ ll /sticky/dir/ -a
总用量 0
drwxrwxrwx 2 root root  6  9月 26 05:11 .
drwxrwxrwx 3 root root 17  9月 26 04:59 ..
```

1.3.8 特殊属性

基础知识

特殊属性

```
    在Linux系统中，文件的特殊属性提供了 "额外的" 安全性和稳定性保障。这些特殊属性通过chattr命令来设置，通过lsattr命令来查看。
```

chattr命令用于改变文件或目录的扩展属性。这些属性可以增强文件的安全性，防止被意外修改、删除或重命名。

```
命令格式
    chattr [+-=][ASacdistu...] 文件或目录名称

常用选项
    -p project  # 设置文件项目编号
    -R          # 递归执行
    -V          # 显示过程，并输出chattr 版本
    -f          # 不输出错误信息
    -v version  # 设置版本

常用动作
    +表示添加属性，-表示移除属性，=表示设置唯一属性，覆盖其他所有属性。
```

```
常用属性
    a：只允许追加数据，不能删除或修改文件内容。
    i：设置文件为不可修改，即不能删除、重命名、修改内容或添加链接。
    s：同步写入磁盘，修改立即生效，而非默认的非同步方式。
    u：当文件被删除时，其内容仍保留在磁盘上，直到空间被其他文件覆盖。
    A：访问文件时不更新访问时间。
    c：自动压缩文件。
    d：防止目录被dump备份。
```

lsattr命令用于查看文件或目录的扩展属性。

```
命令格式
    lsattr [选项] 文件或目录

常用选项
    -a：显示所有文件和目录的属性，包括以.开头的隐藏文件。
    -d：如果目标是目录，则只显示目录本身的属性，而不显示目录内文件的属性。
    -R：递归地列出目录及其所有子目录和文件的属性。
```

注意事项

```
1 使用chattr和lsattr命令时，需要确保有足够的权限，通常是以root用户身份执行。
2 某些文件系统可能不支持所有chattr属性，具体取决于文件系统的类型和版本。
3 修改文件或目录的特殊属性可能会对系统的正常操作产生影响，因此请谨慎使用这些命令。
```

简单实践

添加不可修改的属性

```
准备专属目录
[root@rocky9 ~]# mkdir /attr/dir -p

对文件进行属性增加
[root@rocky9 ~]# chattr +i /attr/dir/file{1,3,5}.txt
```

```
检查文件权限
[root@rocky9 ~]# lsattr /attr/dir/
----i---------------- /attr/dir/file1.txt
-------------------- /attr/dir/file2.txt
----i---------------- /attr/dir/file3.txt
-------------------- /attr/dir/file4.txt
----i---------------- /attr/dir/file5.txt
[root@rocky9 ~]# ls /attr/dir/ -l
总用量 0
-rw-r--r-- 1 root root 0  9月 26 05:39 file1.txt
-rw-r--r-- 1 root root 0  9月 26 05:39 file2.txt
-rw-r--r-- 1 root root 0  9月 26 05:39 file3.txt
-rw-r--r-- 1 root root 0  9月 26 05:39 file4.txt
-rw-r--r-- 1 root root 0  9月 26 05:39 file5.txt
```

有了禁止操作权限，禁止了很多动作

```
[root@rocky9 ~]# rm -rf /attr/dir/*
```

```
rm: 无法删除 '/attr/dir/file1.txt': 不允许的操作
rm: 无法删除 '/attr/dir/file3.txt': 不允许的操作
rm: 无法删除 '/attr/dir/file5.txt': 不允许的操作
[root@rocky9 ~]# ls /attr/dir/
file1.txt  file3.txt  file5.txt
[root@rocky9 ~]# echo "2345" > /attr/dir/file1.txt
-bash: /attr/dir/file1.txt: 不允许的操作
[root@rocky9 ~]# echo "2345" >> /attr/dir/file1.txt
-bash: /attr/dir/file1.txt: 不允许的操作
[root@rocky9 ~]# mv /attr/dir/file1.txt /attr/dir/file-1.txt
mv: 无法将'/attr/dir/file1.txt' 移动至'/attr/dir/file-1.txt': 不允许的操作
[root@rocky9 ~]# mv /attr/dir/file1.txt /tmp/
mv: 无法将'/attr/dir/file1.txt' 移动至'/tmp/file1.txt': 不允许的操作
```

但是也有很多可以做的动作

```
[root@rocky9 ~]# cp /attr/dir/file1.txt{,.bak}
[root@rocky9 ~]# ls /attr/dir/
file1.txt  file1.txt.bak  file3.txt  file5.txt
[root@rocky9 ~]# lsattr /attr/dir/
----i----------------- /attr/dir/file1.txt
----i----------------- /attr/dir/file3.txt
----i----------------- /attr/dir/file5.txt
---------------------- /attr/dir/file1.txt.bak
[root@rocky9 ~]# cat /attr/dir/file1.txt
```

设定仅允许增加的动作属性

```
[root@rocky9 ~]# chattr +a /attr/dir/file1.txt
[root@rocky9 ~]# lsattr /attr/dir/
----ia---------------- /attr/dir/file1.txt
----i----------------- /attr/dir/file3.txt
----i----------------- /attr/dir/file5.txt
---------------------- /attr/dir/file1.txt.bak
```

```
尝试操作
[root@rocky9 ~]# echo "nihao" > /attr/dir/file1.txt
-bash: /attr/dir/file1.txt: 不允许的操作
[root@rocky9 ~]# echo "nihao" >> /attr/dir/file1.txt
-bash: /attr/dir/file1.txt: 不允许的操作

因为携带了i属性，所以需要取消i属性
[root@rocky9 ~]# chattr -i /attr/dir/file1.txt
[root@rocky9 ~]# echo "nihao" > /attr/dir/file1.txt     # 不允许覆盖操作
-bash: /attr/dir/file1.txt: 不允许的操作
[root@rocky9 ~]# echo "nihao" >> /attr/dir/file1.txt    # 只允许追加信息操作
[root@rocky9 ~]# cat /attr/dir/file1.txt
nihao
结果显示：
    仅能够 追加内容，而不可覆盖等操作
```

对目录进行操作

```
[root@rocky9 ~]# chattr +i /attr/dir
[root@rocky9 ~]# lsattr /attr/
----i----------------- /attr/dir
```

```
无法进行增删操作
[root@rocky9 ~]# touch /attr/dir/nihao.txt
touch: 正在设置 '/attr/dir/nihao.txt' 的时间: 没有那个文件或目录
[root@rocky9 ~]# ls /attr/dir/
file1.txt  file1.txt.bak  file3.txt  file5.txt
[root@rocky9 ~]# rm -f /attr/dir/file1.txt.bak
rm: 无法删除 '/attr/dir/file1.txt.bak': 不允许的操作
```

```
不影响文件的正常使用
[root@rocky9 ~]# echo xxx >> /attr/dir/file1.txt
[root@rocky9 ~]# cat /attr/dir/file1.txt
nihao
xxx
[root@rocky9 ~]# echo xxx >> /attr/dir/file1.txt.bak
[root@rocky9 ~]# cat /attr/dir/file1.txt.bak
xxx
```

1.3.9 ACL

基础知识

ACL

```
    传统意义层面的文件系统级别的 User + Group + Other + RWX 的授权体系，主要针对的对象是以文件为基本单位的。其实并不能实现 一个文件针对不同的用户设定形式各异的操作权限。

    Linux系统的ACL（Access Control Lists，访问控制列表）是一种用于控制对文件和目录访问权限的机制。它扩展了传统的文件权限模型，为系统管理员提供了更加灵活和精细的权限控制手段。
    - CentOS7 默认创建的xfs和ext4文件系统具有ACL功能
    - CentOS7 之前版本，默认手工创建的ext4文件系统无ACL功能，需手动增加
    - 对于ubuntu系统来说，如果没有的话，需要安装 acl 命令
```

setfacl 可设置ACL权限

```
命令格式：
    setfacl [-bkndRLPvh] [{-m|-x} acl_spec] [{-M|-X} acl_file] file ...

常见选项
    -m|--modify=acl             #修改acl权限
    -x|--remove=acl             #删除文件acl 权限
    -b|--remove-all             #删除文件所有acl权限
    --set=acl                   #用新规则替换旧规则，会删除原有ACL项，用新的替代，
                                #一定要包含UGO的设置，不能象 -m一样只有 ACL
一般选项
    -M|--modify-file=file       #从文件读取规则
    -X|--remove-file=file       #从文件读取规则
    -k|--remove-default         #删除默认acl规则
    --set-file=file             #从文件读取新规则
```

```
    --mask                      #重新计算mask值
    -n|--no-mask                #不重新计算mask值
    -d|--default                #在目录上设置默认acl
    -R|--recursive              #递归执行
    -L|--logical                #将acl 应用在软链接指向的目标文件上，与-R一起使用
    -P|--physical               #将acl 不应用在软链接指向的目标文件上，与-R一起使用

注意：
    ACL 也可以对 mask 进行操作，但是它的作用想要有效的限制还是比较高的。
        - mask只影响除所有者和other的之外的人和组的最大权限
        - mask需要与用户的权限进行逻辑与运算后，才能变成有限的权限
        - 用户或组的设置必须存在于mask权限设定范围内才会生效
    也就是说，对于mask，acl 不欢迎你，也没有什么意义。
    演示示例：setfacl -m mask::rx file
```

getfacl 可查看设置的ACL权限

```
命令格式：
    getfacl file ...
```

简单实践

默认ACL的实践

```
[root@rocky9 ~]# mkdir /acl/dir -p
[root@rocky9 ~]# echo "acl" > /acl/dir/acl.txt
```

```
查看默认的acl策略
[root@rocky9 ~]# getfacl /acl/dir/acl.txt
getfacl: Removing leading '/' from absolute path names
# file: acl/dir/acl.txt
# owner: root
# group: root
user::rw-
group::r--
other::r--
```

设置ACL实践

```
设定acl权限
[root@rocky9 ~]# setfacl -m u:sswang:- /acl/dir/acl.txt

查看效果
[root@rocky9 ~]# getfacl /acl/dir/acl.txt
getfacl: Removing leading '/' from absolute path names
# file: acl/dir/acl.txt
# owner: root
# group: root
user::rw-
user:sswang:---
group::r--
mask::r--
other::r--
```

```
查看文件权限
[root@rocky9 ~]# ll /acl/dir/acl.txt
-rw-r--r--+ 1 root root 4  9月 26 08:00 /acl/dir/acl.txt
结果显示：
    多了一个 +
```

```
sswang用户不可读
[root@rocky9 ~]# su - sswang -c "cat /acl/dir/acl.txt"
cat: /acl/dir/acl.txt: 权限不够

其他的普通用户可读但是不可写
[root@rocky9 ~]# su - sswang1 -c "cat /acl/dir/acl.txt"
acl
[root@rocky9 ~]# su - sswang1 -c "echo xxx >> /acl/dir/acl.txt"
-bash:行1: /acl/dir/acl.txt: 权限不够
```

修改ACL写权限

```
[root@rocky9 ~]# setfacl -m u:sswang1:rw /acl/dir/acl.txt
[root@rocky9 ~]# getfacl /acl/dir/acl.txt
getfacl: Removing leading '/' from absolute path names
# file: acl/dir/acl.txt
# owner: root
# group: root
user::rw-
user:sswang:---
user:sswang1:rw-
group::r--
mask::rw-
other::r--
```

```
检测sswang1的普通用户效果
[root@rocky9 ~]# su - sswang1 -c "cat /acl/dir/acl.txt"
acl
[root@rocky9 ~]# su - sswang1 -c "echo xxx >> /acl/dir/acl.txt"

使用其他普通用户查看
[root@rocky9 ~]# su - sswang2 -c "cat /acl/dir/acl.txt"
acl
xxx
[root@rocky9 ~]# su - sswang2 -c "echo xxx >> /acl/dir/acl.txt"
-bash:行1: /acl/dir/acl.txt: 权限不够
```

给组加ACL实践

```
指定组的信息
[root@rocky9 ~]# getent group sswang2
sswang2:x:1025:

为组添加acl信息
[root@rocky9 ~]# setfacl -m g:sswang2:rwx /acl/dir/acl.txt

确认效果
[root@rocky9 ~]# getfacl /acl/dir/acl.txt
getfacl: Removing leading '/' from absolute path names
```

```
# file: acl/dir/acl.txt
# owner: root
# group: root
user::rw-
user:sswang:---
user:sswang1:rw-
group::r--
group:sswang2:rwx
mask::rwx
other::r--
```

```
检查效果
[root@rocky9 ~]# su - sswang2 -c "echo xxx >> /acl/dir/acl.txt"
[root@rocky9 ~]# su - sswang2 -c "cat /acl/dir/acl.txt"
acl
xxx
xxx
```

ACL策略的复制

```
准备空白文件
[root@rocky9 ~]# echo "sswang" > /acl/dir/sswang.txt
```

```
查看两个文件的acl权限区别
[root@rocky9 ~]# getfacl /acl/dir/acl.txt /acl/dir/sswang.txt
getfacl: Removing leading '/' from absolute path names
# file: acl/dir/acl.txt
# owner: root
# group: root
user::rw-
user:sswang:---
user:sswang1:rw-
group::r--
group:sswang2:rwx
mask::rwx
other::r--

# file: acl/dir/sswang.txt
# owner: root
# group: root
user::rw-
group::r--
other::r--
```

```
从文件1里面提取acl规则，然后交给另外一个文件
[root@rocky9 ~]# getfacl /acl/dir/acl.txt | setfacl --set-file=-
/acl/dir/sswang.txt
getfacl: Removing leading '/' from absolute path names

确认效果
[root@rocky9 ~]# getfacl /acl/dir/sswang.txt
getfacl: Removing leading '/' from absolute path names
# file: acl/dir/sswang.txt
# owner: root
# group: root
```

```
user::rw-
user:sswang:---
user:sswang1:rw-
group::r--
group:sswang2:rwx
mask::rwx
other::r--
```

ACL规则的移除

```
移除acl规则
[root@rocky9 ~]# setfacl -x u:sswang /acl/dir/sswang.txt
[root@rocky9 ~]# setfacl -x g:sswang2 /acl/dir/sswang.txt

查看效果
[root@rocky9 ~]# getfacl /acl/dir/sswang.txt
getfacl: Removing leading '/' from absolute path names
# file: acl/dir/sswang.txt
# owner: root
# group: root
user::rw-
user:sswang1:rw-
group::r--
mask::rw-
other::r--
```

移除所有定制ACL权限 - 恢复默认的acl权限

```
恢复默认的ACL权限
[root@rocky9 ~]# setfacl -b /acl/dir/sswang.txt
[root@rocky9 ~]# setfacl -b /acl/dir/acl.txt

确认效果
[root@rocky9 ~]# getfacl /acl/dir/*
getfacl: Removing leading '/' from absolute path names
# file: acl/dir/acl.txt
# owner: root
# group: root
user::rw-
group::r--
other::r--

# file: acl/dir/sswang.txt
# owner: root
# group: root
user::rw-
group::r--
other::r--
```

同时定制多属性信息

```
通过 --set 同时定制多个属性信息
[root@rocky9 ~]# setfacl --set u::rw,u:jerry:-,g::-,o::- /acl/dir/sswang.txt
```

检查效果

```
[root@rocky9 ~]# getfacl /acl/dir/sswang.txt
getfacl: Removing leading '/' from absolute path names
# file: acl/dir/sswang.txt
# owner: root
# group: root
user::rw-
user:jerry:---
group::---
mask::---
other::---
```