

## 第七章：用户管理

尚硅谷云计算 Linux 课程

版本：V1.0

讲师：沈超

### 一 用户相关文件

#### 1 /etc/passwd 用户信息文件

```
root:x:0:0:root:/root:/bin/bash
```

第一列： 用户名

第二列： 密码位

第三列： 用户 ID

✧ 0 超级用户 UID。如果用户 UID 为 0，代表这个账号是管理员账号。那 Linux 中如何把普通用户升级成为管理员呢？就是把其他用户的 UID 修改为 0 就可以了，这点和 Windows 是不同的。不过不建议建立多个管理员账号。

✧ 1-499 系统用户（伪用户）UID。这些 UID 账号是系统保留给系统用户的 UID，也就是说 UID 是 1-499 范围内的用户是不能登录系统的，而是用来运行系统或服务的。其中 1-99 是系统保留的账号，系统自动创建。100-499 是预留给用户创建系统账号的。

✧ 500-60000 普通用户 UID。建立的普通用户 UID 从 500 开始，最大到 60000。这些用户足够使用了，但是如果不够也不用害怕，2.6.x 内核以后的 Linux 系统用户 UID 已经可以支持  $2^{32}$  这么多了。

第四列：组 ID GID 添加用户时，如果不指定用户所属的初始组，那么会建立和用户名相同的组

第五列： 用户说明

第六列： 用户家目录 ~

第七列： 登录 shell /bin/bash

如何把普通用户变成超级用户：把用户 UID 改为 0

#### 2 /etc/shadow 影子文件

```
root:$6$9w5Td6lg$bgpsy3olsq9WwVvS5Sst2W3ZiJpuCGDY.4w4MRk3ob/i85fI38RH15wzVoomff9isV1Pzd  
cXmixzhnMVhMxbv0:15775:0:99999:7:::
```

第一列： 用户名

第二列： 加密密码

我们也可以在密码前人为的加入“!”或“\*”改变加密值让密码暂时失效，使这个用户无法登陆，达到暂时禁止用户登录的效果。

注意所有伪用户的密码都是“!!”或“\*”，代表没有密码是不能登录的。当然我新创建的用户如果不设定密码，它的密码项也是“!!”，代表这个用户没有密码，不能登录

第三列： 密码最近更改时间，1970 年 1 月 1 日作为标准时间

时间戳转日期

```
[root@localhost ~]# date -d "1970-01-01 15775 days"
2013 年 03 月 11 日 星期一 00:00:00 CST
```

日期转时间戳

```
[root@localhost ~]# echo $(( $(date --date="2013/03/11" +%s)/86400+1 ))
15775
```

第四列： 两次密码的修改间隔时间（和第 3 字段相比）

第五列： 密码有效期（和第 3 字段相比）

第六列： 密码修改到期前的警告天数（和第 5 字段相比）

第七列： 密码过期后的宽限天数（和第 5 字段相比）

第八列： 密码失效时间

这里同样要写时间戳，也就是用 1970 年 1 月 1 日进行时间换算。如果超过了失效时间，就算密码没有过期，用户也就失效无法使用了

第九列： 保留

### 3 /etc/group 组信息文件

```
root:x:0:root
```

第一列： 组名

第二列： 组密码位

第三列： GID

第四列： 此组中支持的其他用户. 附加组是此组的用户

初始组：每个用户初始组只能有一个，初始组只能有一个，一般都是和用户名相同的组作为初始组

附加组：每个用户可以属于多个附加组。要把用户加入组，都是加入附加组

#### 4 组密码文件/etc/gshadow

如果我给用户组设定了组管理员，并给该用户组设定了组密码，组密码就保存在这个文件当中。组管理员就可以利用这个密码管理这个用户组了。

#### 5 用户的家目录

#### 6 用户邮箱目录

这个邮箱在/var/spool/mail 目录当中，例如 user1 用户的邮箱就是/var/spool/mail/user1 文件

#### 7 用户模板目录

/etc/skel/

## 二 用户管理命令

### 1 添加用户

#### 1.1 手工删除用户

手工删除用户试验：手工删除，如果可以正常建立用户，证明用户删除干净。

```
/etc/passwd
/etc/shadow
/etc/group
/etc/gshadow
/home/user1
/var/spool/mail/user1  邮箱
```

#### 1.2 useradd 命令

useradd 选项 用户名

选项：

```
-u 550 指定 UID
-g 组名 指定初始组 不要手工指定
-G 组名 指定附加组，把用户加入组，使用附加组
-c 说明 添加说明
-d 目录 手工指定家目录，目录不需要事先建立
-s shell /bin/bash.
```

例如：

```
[root@localhost ~]# groupadd lamp1
#先手工添加 lamp1 用户组，因为我一会要把 lamp1 用户的初始组指定过来，如果不事先建立，会报错用户组不存在
[root@localhost ~]# useradd -u 550 -g lamp1 -G root -d /home/lamp1 \
-c "test user" -s /bin/bash lamp1
#建立用户 lamp1 的同时指定了 UID (550)，初始组 (lamp1)，附加组 (root)，家目录 (/home/lamp1)，用户说明(test user)和用户登录 shell (/bin/bash)
[root@localhost ~]# grep "lamp1" /etc/passwd /etc/shadow /etc/group
#同时查看三个文件
/etc/passwd:lamp1:x:550:502:test user:/home/lamp1:/bin/bash
#用户的 UID、初始组、用户说明、家目录和登录 shell 都和命令手工指定的一致
/etc/shadow:lamp1:!!:15710:0:99999:7:::
#lamp1 用户还没有设定密码
/etc/group:root:x:0:lamp1
#lamp1 用户加入了 root 组，root 组是 lamp1 用户的附加组
/etc/group:lamp1:x:502:
#GID502 的组是 lamp1 组
```

```
[root@localhost ~]# ll -d /home/lamp1/
drwx----- 3 lamp1 lamp1 4096 1月 6 01:13 /home/lamp1/
#家目录也建立了啊。不需要手工建立家目录
```

### 1.3 useradd 默认值

useradd 添加用户时参考的默认值文件主要有两个，分别是/etc/default/useradd 和 /etc/login.defs

#### 1) /etc/default/useradd

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

挨个解释下：

✧ GROUP=100

这个选项是建立用户的默认组，也就是说添加每个用户时，用户的初始组就是 GID 为 100 的这个用户组。目前我们采用的机制私有用户组机制。

✧ HOME=/home

这个选项是用户的家目录的默认位置，所以所有的新建用户的家目录默认都在/home/下。

✧ INACTIVE=-1

这个选项就是密码过期后的宽限天数，也就是/etc/shadow 文件的第七个字段。如果是天数，比如 10 代表密码过期后 10 天后失效；如果是 0，代表密码过期后立即失效；如果是-1，则代表密码永远不会失效。这里默认值是-1，所以所有新建的用户密码都不会失效。

✧ EXPIRE=

这个选项是密码失效时间，也就是/etc/shadow 文件的第八个字段。也就是说用户到达这个日期后就会直接失效。当然这里也是使用时间戳来表示日期的。默认值是空，所以所有新建用户没有失效时间，永久有效。

✧ SHELL=/bin/bash

这个选项是用户的默认 shell 的。/bin/bash 是 Linux 的标志 shell，所以所有新建的用户默认都具备 shell 赋予的权限。

✧ SKEL=/etc/skel

这个选项就是定义用户的模板目录的位置，/etc/skel/目录中的文件都会复制到新建用户的家目录当中。

✧ CREATE\_MAIL\_SPOOL=yes

这个选项定义是否给新建用户建立邮箱，默认是创建，也就是说所有的新建用户系统都会新建一个邮箱，放在/var/spool/mail/下和用户名相同。

#### 2) /etc/login.defs

```
[root@localhost ~]# vi /etc/login.defs
```

#这个文件有些注释，把注释删除掉，文件内容就变成下面这个样子了

```
MAIL_DIR      /var/spool/mail
```

```
PASS_MAX_DAYS 99999
```

```
PASS_MIN_DAYS 0
```

```
PASS_MIN_LEN  5
```

```
PASS_WARN_AGE 7
```

```
UID_MIN        500
```

```
UID_MAX        60000
```

```
GID_MIN        500
```

```
GID_MAX        60000
```

```
CREATE_HOME    yes
```

```
UMASK          077
```

```
USERGROUPS_ENAB yes
```

```
ENCRYPT_METHOD SHA512
```

我们一行一行解释下文件内容：

✧ MAIL\_DIR /var/spool/mail

这行指定了新建用户的默认邮箱位置。比如 user1 用户的邮箱就是/var/spool/mail/user1。

✧ PASS\_MAX\_DAYS 99999

这行指定的是密码的有效期，也就是/etc/shadow 文件的第五字段。代表多少天之后必须修改密码，默认值是 99999。

✧ PASS\_MIN\_DAYS 0

这行指定的是两次密码的修改间隔时间，也就是/etc/shadow 文件的第四字段。代表第一次修改密码之后，几天后才能再次修改密码。默认值是 0。

✧ PASS\_MIN\_LEN 5

这行代表密码的最小长度，默认不小于 5 位。但是我们现在用户登录时验证已经被 PAM 模块取代，所以这个选项并不生效。

✧ PASS\_WARN\_AGE 7

这行代表密码修改到期前的警告天数，也就是/etc/shadow 文件的第六字段。代表密码到底有效期前多少天开始进行警告提醒，默认值是 7 天。

✧ UID\_MIN 500

✧ UID\_MAX 60000

这两行代表创建用户时，最小 UID 和最大的 UID 的范围。我们 2.6.x 内核开始，Linux 用户的 UID 最大可以支持  $2^{32}$  这么多，但是真正使用时最大范围是 60000。还要注意如果我手工指定了一个用户的 UID 是 550，那么下一个创建的用户的 UID 就会从 551 开始，哪怕 500-549 之间的 UID 没有使用（小于 500 的 UID 是给伪用户预留的）。

✧ GID\_MIN 500

✧ GID\_MAX 60000

这两行指定了 GID 的最小值和最大值之间的范围。

✧ CREATE\_HOME yes

这行指定建立用户时是否自动建立用户的家目录，默认是建立

✧ UMASK 077

这行指定的是建立的用户家目录的默认权限，因为 umask 值是 077，所以新建的用户家目录的权限是 700，umask 的具体作用和修改方法我们可以参考下一章权限设定章节。

✧ USERGROUPS\_ENAB yes

这行指定的是使用命令 userdel 删除用户时，是否删除用户的初始组，默认是删除。

✧ ENCRYPT\_METHOD SHA512

这行指定 Linux 用户的密码使用 SHA512 散列模式加密，这是新的密码加密模式，原先的 Linux 只能用 DES 或 MD5 方式加密

## 2 设定密码

```
[root@localhost ~]#passwd [选项] 用户名
```

选项：

- l: 暂时锁定用户。仅 root 用户可用
- u: 解锁用户。仅 root 用户可用
- stdin: 可以将通过管道符输出的数据作为用户的密码。主要在批量添加用户时使用

```
[root@localhost ~]#passwd
```

*#passwd 直接回车代表修改当前用户的密码*

也可以使用字符串作为密码：

```
[root@localhost ~]# echo "123" | passwd --stdin user1
```

更改用户 user1 的密码。

可以通过命令，把密码修改日期归零（shadow 第 3 字段）。这样用户一登陆就要修改密码，例如：

```
[root@localhost ~]# chage -d 0 user1
```

## 3 用户信息修改

usermod 命令是修改已经添加的用户的信息的，命令如下：

```
[root@localhost ~]#usermod [选项] 用户名
```

选项：

- u UID: 修改用户的 UID
- d 家目录: 修改用户的家目录。家目录必须写绝对路径
- c 用户说明: 修改用户的说明信息，就是/etc/passwd 文件的第五个字段
- g 组名: 修改用户的初始组，就是/etc/passwd 文件的第四个字段
- G 组名: 修改用户的附加组，其实就是把用户加入其他用户组
- s shell: 修改用户的登录 Shell。默认是/bin/bash
- e 日期: 修改用户的失效日期，格式为“YYYY-MM-DD”。也就是/etc/shadow 文件的第八个字段
- L: 临时锁定用户（Lock）
- U: 解锁用户（Unlock）

有学员突发奇想，问超哥，那用户可以修改用户名吗？当然可以：

```
[root@localhost ~]# usermod -l 新名 旧名
#改名
```

但是真不建议改名，这样及其容易把管理员自己搞晕菜，建议删除旧用户，再建立新用户！

## 4 删除用户

```
[root@localhost ~]# userdel [-r] 用户名
```

选项：

-r: 在删除用户的同时删除用户的家目录

## 5 切换用户身份

su 命令可以切换成不同的用户身份，命令格式如下：

```
[root@localhost ~]# su [选项] 用户名
```

选项：

-: 选项只使用“-”代表连带用户的环境变量一起切换

-c 命令: 仅执行一次命令，而不切换用户身份

“-”不能省略，它代表切换用户身份时，用户的环境变量也要切换成新用户的环境变量。

## 三 组管理命令

### 1 添加用户组: groupadd

添加用户组的命令是 groupadd，命令格式如下：

```
[root@localhost ~]# groupadd [选项] 组名
```

选项：

-g GID: 指定组 ID

添加用户组的命令比较简单，举个例子：

```
[root@localhost ~]# groupadd group1
```

#添加 group1 组

```
[root@localhost ~]# grep "group1" /etc/group
```

```
group1:x:502:
```

### 2 删除用户组: groupdel

groupdel 命令用于删除用户组，命令格式如下：

```
[root@localhost ~]#groupdel 组名
```

例子：



```
[root@localhost ~]# groupdel testgrp
#删除 testgrp 组
```

不过大家要注意，要删除的组不能是其他用户的初始组，也就是说这个组中没有初始用户才可以删除。如果组中有附加用户，则删除组时不受影响。

## 4 把用户添加进组或从组中删除：gpasswd

其实 gpasswd 命令是用来设定组密码并指定组管理员的，不过我们在前面已经说了，组密码和组管理员功能很少使用，而且完全可以被 sudo 命令取代，所以 gpasswd 命令现在主要用于把用户添加进组或从组中删除。命令格式如下：

```
[root@localhost ~]# gpasswd [选项] 组名
选项：
-a 用户名： 把用户加入组
-d 用户名： 把用户从组中删除
```

举个例子：

```
[root@localhost ~]# groupadd grouptest
#添加组 grouptest
[root@localhost ~]# gpasswd -a user1 grouptest
Adding user user1 to group grouptest
#把用户 user1 加入 grouptest 组
[root@localhost ~]# grep "user1" /etc/group
user1:x:501:
grouptest:x:505:user1
#查看一下，user1 用户已经作为附加用户加入 grouptest 组
[root@localhost ~]# gpasswd -d user1 grouptest
Removing user user1 from group grouptest
#把用户 user1 从组中删除
[root@localhost ~]# grep "grouptest" /etc/group
grouptest:x:505:
#组中没有 user1 用户了
```

大家注意，也可以使用 usermod 命令把用户加入某个组，不过 usermod 命令的操作对象是用户，命令是“usermod -G grouptest user1”，把用户名作为参数放在最后；而 gpasswd 命令的操作对象是组，命令是“gpasswd -a user1 grouptest”，把组名作为参数放在最后。

## 5 改变有效组：newgrp

我们说过，每个用户可以属于一个初始组（用户是这个组的初始用户），也可以属于多个附加组（用户是这个组的附加用户）。既然用户可以属于这么多用户组，那么用户在创建文件后，默认生效的组身份是哪个呢？当然是初始用户组的组身份生效了，因为初始组是用户一旦登录就直接获得的组身份。也就是说，用户在创建文件后，文件的属组是用户的初始组，因为用户的有效组默认是初始组。



既然用户属于多个用户组，那么能不能改变用户的有效组呢？使用命令 `newgrp` 就可以切换用户的有效组。命令格式如下：

```
[root@localhost ~]# newgrp 组名
```

举个例子，我们已经有了普通用户 `user1`，默认会建立 `user1` 用户组，`user1` 组是 `user1` 用户的初始组。我们再把 `user1` 用户加入 `group1` 组，那么 `group1` 组就是 `user1` 用户的附加组。当 `user1` 用户创建文件 `test1` 时，`test1` 文件的属组是 `user1` 组，因为 `user1` 组是 `user1` 用户的有效组。通过 `newgrp` 命令就可以把 `user1` 用户的有效组变成 `group1` 组，当 `user1` 用户创建文件 `test2` 时，就会发现 `test2` 文件的属组就是 `group1` 组。命令如下：

```
[root@localhost ~]# groupadd group1
#添加组 group1
[root@localhost ~]# gpasswd -a user1 group1
Adding user user1 to group group1
#把 user1 用户加入 group1 组
[root@localhost ~]# grep "user1" /etc/group
user1:x:501:
group1:x:503:user1
#user1 用户既属于 user1 组，也属于 group1 组
[root@localhost ~]# su - user1
#切换成 user1 身份，超级用户切换成普通用户不用密码
[user1@localhost ~]$ touch test1
#创建文件 test1
[user1@localhost ~]$ ll test1
-rw-rw-r-- 1 user1 user1 0 1月 14 05:43 test1
#test1 文件的默认属组是 user1 组
[user1@localhost ~]$ newgrp group1
#切换 user1 用户的有效组为 group1 组
[user1@localhost ~]$ touch test2
#创建文件 test2
[user1@localhost ~]$ ll test2
-rw-r--r-- 1 user1 group1 0 1月 14 05:44 test2
#test2 文件的默认属组是 group1 组
```

通过这个例子明白有效组的作用了吗？其实就是当用户属于多个组时，在创建文件时哪个组身份生效。使用 `newgrp` 命令可以在多个组身份之间切换。

## 6. 组权限实验