

第 9 章 用户与用户组管理

本章介绍 Linux 上用户和用户组的管理。作为一种多用户的操作系统，Linux 可以允许多个用户同时登录到系统上，并响应每一个用户的请求。对于系统管理员而言，一个非常重要的工作就是对用户账户进行管理。这些工作包括添加和删除用户、分配用户主目录、限制用户的权限等。接下来将逐一讨论这些方面。

9.1 用户与用户组基础

计算机科学还没有进展到让每一台电脑都能通过生物学特征识别人的程度。在绝大多数情况下，用户名是身份的唯一标志，计算机通过用户提供的口令来验证这一标志。这种简单而实用的方式被广泛应用于几乎所有的计算机系统中。遗憾的是，也是由于这种“简单”的验证方式，使得在世界各地，每一天都有无数的账号被盗取。因此选择一个合适的用户名和一个不易被破解的密码非常重要。

Linux 也运用同样的方法来识别用户：用户提供用户名和密码，经过验证后登录到系统。Linux 会为每一个用户启动一个进程，然后由这个进程接受用户的各种请求。在建立用户的时候，需要限定其权限，例如不能修改系统配置文件，不能查看其他用户的目录等。就像在一个银行安全系统中，每一个人只能处理其职权范围内的事情。另外，系统中有一个特殊的 root 用户，这个用户有权对系统进行任何操作而不受限制。关于 root 的详细介绍，参见 3.1 节。

所谓“人以群分”，可以把几个用户归在一起，这样的组被称为“用户组”。可以设定一个用户组的权限，这样，这个组里的用户就自动拥有了这些权限。对于一个多人协作的项目而言，定义一个包含项目成员的组往往是非常有用的。

在某些服务器程序安装时，会生成一些特定的用户和用户组，用于对服务器进行管理。例如，可以使用 mysql 用户启动和停止 MYSQL 服务器。之所以不使用 root 用户启动某些服务，主要是出于安全性的考虑。因为当某个运行中进程的 UID 属于一个受限用户的话，那么即使这个进程出了什么问题，也不会对系统安全产生毁灭性打击（关于进程的相关知识，可以参考第 10 章）。

9.2 快速上手：为朋友 John 添加账户

John 的笔记本送去维修了，希望借台电脑用几天。但是电脑上有一些私人文件，或许不应该让 John 看到。因此应该为 John 单独添加一个账户，而不是使用当前系统上已有的

账户。

打开终端，输入：

```
$ sudo useradd -m john          ##添加一个用户名为 john 的用户，并自动建立主目录
```

注意在输入口令的时候，出于安全考虑，屏幕上并不会有任何显示（包括“*”号）。现在，应该把 John 叫过来，让他自己输入一个密码。

```
$ sudo passwd john              ##更改 john 的登录密码
输入新的 UNIX 口令：
重新输入新的 UNIX 口令：
passwd: 已成功更新密码
```

现在，John 可以使用自己的账号登录到系统了。只要将私人文件设置为他人不可读，那么就不用担心 John 会查看到这些文件。关于如何设置文件权限，请参考 6.5 节。

9.3 添加用户

添加用户是系统管理的例行工作，在“快速上手”环节，读者已经实践了添加用户的基本步骤。接下来将详细讨论 `useradd` 和 `groupadd` 命令的各个常用选项，以及如何使用图形化的用户管理工具。最后介绍如何追踪用户状态。

9.3.1 使用命令行工具：useradd 和 groupadd

在默认情况下，不带 `-m` 参数的 `useradd` 命令不会为新用户建立主目录。在这种情况下，用户可以登录到系统的 Shell，但不能登录到图形界面。这是因为桌面环境无论是 KDE 还是 GNOME，需要用到用户主目录中的一些配置文件。例如，以下面的方式使用 `useradd` 命令添加一个用户 nox。

```
$ useradd nox
$ passwd nox                      ##设置 nox 用户的口令
输入新的 UNIX 口令：
重新输入新的 UNIX 口令：
passwd: 已成功更新密码
```

当使用 nox 用户账号登录 GNOME 时，系统会提示无法找到用户主目录，并拒绝登录，如图 9.1 和图 9.2 所示。

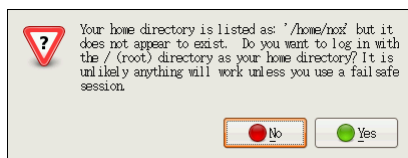


图 9.1 无法找到用户主目录



图 9.2 无法登录图形界面

如果在字符界面的 2 号控制台（可以使用快捷键 `Ctrl+Alt+F2` 进入。）使用 nox 账号

登录，系统会引导 nox 用户进入根目录。此后，用户可以继续操作，如图 9.3 所示。

```
Ubuntu 8.04.1 lewis-laptop tty2
lewis-laptop login: nox
Password:
Linux lewis-laptop 2.6.24-19-generic #1 SMP Wed Jun 18 14:43:41 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No directory, logging in with HOME=/
$ _
```

图 9.3 登录 2 号控制台

`useradd` 命令中另一个比较常用的参数是 `-g`。该参数用于指定用户所属的组。下面这条命令建立名为 `mike` 的用户账号，并指定其属于 `users` 组。

```
$ sudo useradd -g users mike
```

在用户建立的时候为其指定一个组看上去是一个很不错的想法，但遗憾的是，这样的设置增加了用户由于不经意地设置权限而能够彼此读取文件的可能性，尽管这通常不是用户的本意。因此一个好的建议是，在新建用户的时候单独创建一个同名的用户组，然后把用户归入这个组中——这正是带 `-g` 参数的 `useradd` 命令的默认行为。

`useradd` 的 `-s` 参数用于指定用户登录后所使用的 Shell。下面的命令建立名为 `mike` 的用户账号，并指定其登录后使用 `bash` 作为 Shell。

```
$ sudo useradd -s /bin/bash mike
```

可以在 `/bin` 目录下找到特定的 Shell。常用的有 `BASH`、`TCSH`、`ZSH`（Z-Shell）、`SH`（Bourne Shell）等。如果不指定 `-s` 参数，那么默认将使用 `sh`（在大部分系统中，这是指向 `BASH` 的符号链接）登录系统。

添加组可以使用 `groupadd` 命令，下面这条命令在系统中添加一个名为 `newgroup` 的组。

```
$ sudo groupadd newgroup
```

9.3.2 使用图形化的管理工具

除了传统的命令行方式，Linux 还提供图形化工具对用户和用户组进行管理。相比较 `useradd` 等命令而言，图形化工具提供了更为友好的用户接口——当然，这是以牺牲一定灵活性为代价的。下面以 Ubuntu 下的“用户和组”管理工具为例进行介绍。其他的发行版工具可以遵循类似的步骤操作。

(1) 选择“系统”|“系统管理”|“用户和组”命令可以打开这个工具。初始状态下，所有的功能都被禁用，如图 9.4 所示，直到用户单击“解锁”按钮——此时系统将要求输入管理员口令，并对此进行验证。

(2) 单击“添加用户”按钮，打开“新建用户账户”对话框，如图 9.5 所示。



图 9.4 打开“用户和组”工具

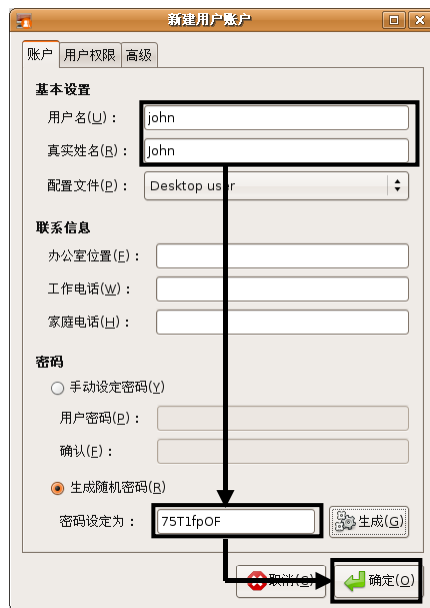


图 9.5 新建用户账户——基本设置

(3) 在“用户名”文本框中填写用户名，并在“密码设定为”文本框中设置相关的密码。比较有用的一个选项是“生成随机密码”单选按钮，系统会随机以数字和大小写字母生成一个 8 位的口令，对于临时给新用户设置口令，而又需要保证口令安全性的管理员而言，这个工具非常有用。

(4) 除了在“账户”选项卡中对基本个人信息进行设置，还有其他一些选项可供选择。“用户权限”选项卡决定新添加的用户拥有的权限，如图 9.6 所示；“高级”选项卡中可以设置用户主目录、登录的 Shell、用户所属的组，以及用户 ID，如图 9.7 所示。用户 ID 用于唯一标示系统中的用户，在大多数情况下，这并不需要管理员进行设置，使用系统分配的默认值就可以了（关于用户 ID 的详细信息可参考 9.9 节）。

(5) 完成所有这些设置后，单击“确定”按钮就可以新建一个用户了。

完成用户的添加后，可以看到新用户（在本例中是 John）出现在列表中，如图 9.8 所示。单击 John 所在的行，使其高亮显示，单击“属性”按钮将弹出同刚才类似的界面。在这里可以对 John 的属性进行更改。单击“确定”按钮保存所作的修改。

单击“管理组”按钮可以对用户组进行管理。在“新建组”对话框中，用户可以输入组名，并选择将哪些用户添加至这个组中，如图 9.9 所示。一个用户可以属于多个组，就像一个人在生活中往往需要扮演不同角色一样。同样，可以选定一个组，然后对其属性进行修改。

完成所有这些工作后，单击“关闭”按钮退出程序。



图 9.6 新建用户账户——设置用户权限



图 9.7 新建用户账户——高级选项



图 9.8 选中新创建的用户



图 9.9 管理用户组

9.3.3 记录用户操作：history

Linux，准确地说是 Shell，会记录用户的每一条命令。通过 history 命令，用户可以看到自己曾经执行的操作。

```
$ history
16 cd /media/fishbox/software/
17 ls
18 sudo tar zxvf ies4linux-latest.tar.gz
19 cd ies4linux-2.99.0.1/
20 ls
21 vi README
22 ./ies4linux
...
```


注意：history 命令仅在 BASH 中适用。

`history` 会列出所有使用过的命令并加以编号。这些信息被存储在用户主目录的 `.bash_history` 文件中，这个文件默认情况下可以存储 1000 条命令记录。当然，一次列出那么多命令除了让人迷茫外，没有其他什么用途。为此，可以指定让 `history` 列出最近几次输入的命令。

```
$ history 10                                ##列出最近使用的 10 条命令
508 cd /home/john/
509 vi .bash_history
510 sudo vi .bash_history
511 cd
512 ls -al
513 ls -al | grep bash_history
514 history
515 history | more
516 vi .bash_history
517 history 10
```

但是，`history` 只能列出当前用户的操作记录。对于管理员而言，有时候需要查看其他用户的操作记录，此时可以读取该用户主目录下的 `.bash_history` 文件。现在看看 `john` 都干了些什么。

```
$ cd /home/john/                            ##进入 john 的主目录
$ sudo cat .bash_history                     ##查看 .bash_history 文件
cd /home/lewis/
ls
cd c_class/
ls
cd ..
ls
cd c_class/
./a.out
exit
```

 **注意：**`.bash_history` 这个文件对于其他受限用户是不可读的，这也正是为什么要使用 `sudo` 的原因。

9.3.4 直接编辑 `passwd` 和 `shadow` 文件

在 Linux 中所作的一切基本配置最终都将反映到配置文件中，用户管理也不例外。所有的用户信息都登记在 `/etc/passwd` 文件中，而 `/etc/shadow` 文件则保存着用户的登录密码。

诸如 `useradd` 这样的工具实际上对用户隐藏了用户管理的细节。可以通过手动编辑 `passwd` 和 `shadow` 这两个文件实现 `useradd` 等工具的所有功能。其中，`passwd` 文件对所有用户可读，而 `shadow` 则只能用 `root` 账号查看。为了保证口令的安全性，这一点很容易理解。当然，修改这两个文件都需要 `root` 权限。

不推荐初学者通过直接编辑这两个文件实现用户管理，尽管这种做法带来了很大程度上的灵活性。如果读者的确需要了解如何编辑 `passwd` 和 `shadow`，可以参考本章的“进阶”部分。

9.4 删除用户：userdel

`userdel` 命令用于删除用户账号。下面这条命令删除 `mike` 这个账号。

```
$ sudo userdel mike
```

在默认情况下，`userdel` 并不会删除用户的主目录。除非使用了 `-r` 选项。下面这条命令将 `john` 的账号删除，同时删除其主目录。

```
$ sudo userdel -r john
```

在删除用户的同时删除其主目录，以释放硬盘空间，这看起来无可厚非。但是，在输入 `-r` 选项之前，仍然有一个问题需要问问自己：需要这么着急吗？如果被删除的用户又要恢复，或者用户的某些文件还需要使用（这样的情况在服务器上经常出现）那么有必要暂时保留这些文件。比较妥当的方法是，将被删用户的主目录保留几周，然后再手动删除。在实际的工作环境中，这个做法显得尤为重要。

9.5 管理用户账号：usermod

可以使用 `usermod` 命令来修改已有的用户账号。这个命令有多个不同的选项，对应于账号的各个属性。如表 9.1 列举了各个常用选项及其含义。

表 9.1 `usermod` 命令的常用选项

| 选 项 | 含 义 |
|-----------------|-----------------------------------|
| <code>-d</code> | 修改用户主目录 |
| <code>-e</code> | 修改账号的有效期限。以公元月/日/年的形式表示（MM/DD/YY） |
| <code>-g</code> | 修改用户所属的组 |
| <code>-l</code> | 修改用户账号名称 |
| <code>-s</code> | 修改用户登录后所使用的 Shell |

下面这条命令将 `john` 改名为 `mike`，主目录改为 `/home/mike`，并设置账号有效期至 2010 年 12 月 31 日。

```
$ sudo usermod -l mike -d /home/mike -e 12/31/10 john
```

和 `useradd` 的原理一样，`usermod` 也通过修改 `/etc/passwd`、`/etc/shadow`、`/etc/group` 这 3 个文件来实现用户属性的设置。`usermod` 的完整选项可以查看其用户手册。

9.6 查看用户信息：id

`id` 命令用于查看用户的 UID、GID 及其所属的组。这个命令以用户名作为参数，下面

这条命令显示了 nobody 用户的 UID、GID 及其属于的组信息。

```
$ id nobody
uid=65534(nobody) gid=65534(nogroup) 组=65534(nogroup)
```


 **提示：**关于用户的 UID 和 GID，请参考 9.9 节。

使用不带任何参数的 id 命令显示当前登录用户的信息。

```
$ id
uid=1000(lewis) gid=1000(lewis) 组
=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46
(plugdev),107(fuse),109(lpadmin),115(admin),125(vboxusers),127(sambas-
hare),1000(lewis)
```

9.7 用户间切换：su

在第 3 章的内容中曾经介绍过，使用 root 账号一个比较好的做法是使用 su 命令。不带任何参数的 su 命令会将用户提升至 root 权限，当然首先需要提供 root 口令。通过 su 命令所获得的特权将一直持续到使用 exit 命令退出为止。


 **注意：**Ubuntu Linux 的限制非常严格。在默认情况下，系统没有合法的 root 口令。这意味着不能使用 su 命令提升至 root 权限，而必须用 sudo 来获得 root 访问权。

也可以使用 su 命令切换到其他用户。下面这个命令将当前身份转变为 john。

```
$ su john
```

系统会要求输入 john 口令。通过验证后，就可以访问 john 账号了。通过 exit 命令回到之前的账号。

```
$ exit
```

 **安全性提示：**尽量通过绝对路径使用 su 命令，这个命令通常保存在/bin 目录下。这将在一定程度上防止溜入到搜索路径下的名为 su 的程序窃取用户口令。关于搜索路径，参见 21.3 节。

9.8 受限的特权：sudo

使用 su 命令提升权限已经让系统安全得多了，但 root 权限的不可分割让事情变得有些棘手。如果用户 john 想要运行某个特权命令，那他除了向管理员索取 root 口令外别无他法。仅仅为了一个特权操作而赋予用户控制系统的完整权限，这种做法听起来有点可笑，但这确实存在于某些不规范的管理环境中。

最常见的解决方法是使用 sudo 程序。这个程序接受命令行作为参数，并以 root 身份

(或者也可以是其他用户) 执行它。在执行命令之前, `sudo` 会首先要求用户输入自己的口令, 口令只需要输入一次。出于安全性的考虑, 如果用户在一段时间内(默认是 5 分钟)没有再次使用 `sudo`, 那么此后必须再次输入口令。这样的设置避免了特权用户不经意间将自己的终端留给那些并不受欢迎的人。

管理员通过配置 `/etc/sudoers` 指定用户可以执行的特权命令, 下面是 Ubuntu 中 `sudoers` 文件的默认设置。

```
# User privilege specification
root    ALL=(ALL) ALL

# Members of the admin group may gain root privileges
%admin   ALL=(ALL) ALL
```

按照惯例, “#” 开头的行是注释行。以 “`root ALL=(ALL) ALL`” 这句话为例, 这段配置指定 `root` 用户可以使用 `sudo` 在任何机器上(第 1 个 `ALL`) 以任何用户身份(第 2 个 `ALL`) 执行任何命令(第 3 个 `ALL`)。最后一行用 “`%admin`” 替代了所有属于 `admin` 组的用户。在 Ubuntu 中, 安装时创建的那个用户会自动被加入 `admin` 组。

总体来说, `sudoers` 中的每一行权限说明包含了下面这些内容。

- ☐ 该权限适用的用户;
- ☐ 这一行配置在哪些主机上适用;
- ☐ 该用户可以运行的命令;
- ☐ 该命令应该以哪个用户身份执行。


下面来看一段稍复杂一些的配置。这段配置涉及 3 个用户, 并为他们设置了不同的权限。

```
Host_Alias    STATION = web1, web2, databank

Cmdnd_Alias   DUMP = /sbin/dump, /sbin/restore

lewis         STATION = ALL
mike          ALL = (ALL) ALL
john         ALL = (operator) DUMP
```

这段配置的开头两行使用关键字 `Host_Alias` 和 `Cmnd_Alias` 分别定义了主机组和命令组。后面就可以用 `STATION` 替代主机 `web1`、`web2` 和 `databank`; 用 `DUMP` 替代命令 `/sbin/dump` 和 `/sbin/restore`。这种设置可以让配置文件更清晰, 同时也更容易维护。

 **注意:** `sudoers` 中的命令应该使用绝对路径来指定, 这样可以防止一些人以 `root` 身份执行自己的脚本程序。

接下来的 3 行配置了用户的权限。第 1 行是关于用户 `lewis` 的。`lewis` 可以在 `STATION` 组的计算机上(`web1`、`web2` 和 `databank`) 执行任何命令。由于在代表命令的 `ALL` 之前没有使用小括号 “`()`” 指定用户, 因此 `lewis` 将以 `root` 身份执行这些命令。

第 2 行是关于用户 `mike` 的。`mike` 可以在所有的计算机上运行任何命令。由于小括号中的用户列表使用了关键字 `ALL`, 因此 `mike` 可以用 `sudo` 以任何用户身份执行命令。可以使用带 `-u` 选项的 `sudo` 命令改变用户身份。例如 `mike` 可以这样以用户 `peter` 的身份建立文件。

```
$ sudo -u peter touch new_file
```

最后一行是关于用户 `john` 的。`john` 可以在所有主机上执行 `/sbin/dump` 和 `/sbin/restore` 这两个命令——但必须以 `operator` 的身份。为此，`john` 必须像这样使用 `dump` 命令。

```
$ sudo -u operator /sbin/dump backup /dev/sdb1
```

修改 `sudoers` 文件应该使用 `visudo` 命令。这个命令依次执行下面这些操作。

- (1) 检查以确保没有其他人正在编辑这个文件。
- (2) 调用一个编辑器编辑该文件。
- (3) 验证并确保编辑后的文件没有语法错误。
- (4) 安装使 `sudoers` 文件生效。

现在看起来 `sudo` 的确要比 `su` 灵活和有效得多。但没有什么解决方案是十全十美的。使用 `sudo` 实际上增加了系统中特权用户的数量，如果其中一个用户的口令被人破解了，那么整个系统就面临威胁。保证每个拥有特权的用户保管好自己的口令显然比自己保管一个 `root` 口令困难得多——尽管除此之外并没有什么好办法。

9.9 进阶 1: `/etc/passwd` 文件

本节简要介绍 `/etc/passwd` 文件，这是 Linux 中用于存储用户信息的文件。在早期的 Linux 中，`/etc/passwd` 是管理用户的唯一场所，包括用户口令在内的所有信息都记录在这个文件中。出于安全性考虑，现在用户口令已经转而保存在 `/etc/shadow` 中了，这个文件将在 9.10 节讨论。

9.9.1 `/etc/passwd` 文件概览

用户的基本信息被储存在 `/etc/passwd` 文件中。这个文件的每一行代表一个用户，使用 `cat` 命令查看到的文件内容大致如下：

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
...
```

每一行由 7 个字段组成，字段间使用冒号分隔。下面是各字段的含义。

- ❑ 登录名。
- ❑ 口令占位符。
- ❑ 用户 ID 号 (UID)。
- ❑ 默认组 ID 号 (GID)。
- ❑ 用户的私人信息：包括全名、办公室、工作电话、家庭电话等。
- ❑ 用户主目录。
- ❑ 登录 Shell。

其中大部分字段的作用是显而易见的，并在先前的章节中已做过相应介绍。下面将针对加密字段、UID 和 GID 作详细讲解。

9.9.2 加密的口令

把加密口令放在这里讲解似乎显得有点不合时宜。正如读者所看到的，在 `passwd` 的口令字段中，只有一个 `x` 摆放在那里。难道用户的口令就是 `x` 吗？这显然不可能，由于 `passwd` 文件需要对所有用户可读，因此另找一个地方存放口令显得很有必要。如今绝大多数系统都将用户口令经过加密后存放在 `/etc/shadow` 文件中（这个文件只对 `root` 可读），然后在 `passwd` 文件的口令字段放入一个 `x` 作为占位符。

无论加密口令被存放在哪，其原理总是相同的。大部分 Linux 发行版可以识别多种不同的加密算法。通过分析加密后的数据，系统可以知道使用的是哪一种算法。因此，可以在一套系统上使用多种不同的加密方式。

目前在 Linux 上使用最广泛的加密算法是 MD5。MD5 可以对任意长度的口令进行加密，并且不会产生损失，所以一般来说，口令越长越安全。无论加密前的口令多长，经过 MD5 加密之后的长度是一个固定值（34 个字符）——数学总是能让人惊叹。在加密过程中，MD5 算法会随机加入一些被称作“盐（salt）”的数据，从而使一个口令可以对应多个不同的加密后的形式。因此，检查加密后的口令并不会发现两个用户使用相同口令这样的情况。

常用的加密算法总能够通过前缀来识别。MD5 算法总是以“\$1\$”开头，另一种常用的加密算法 Blowfish 以“\$2a\$”开头。不管使用哪一种算法，都应该使用 `passwd` 工具设置相应字段——没有人会选择纸和笔来完成这项工作。

9.9.3 UID 号

UID 号用于唯一标识系统中的用户。这是一个 32 位无符号整数。Linux 规定 `root` 用户的 UID 为 0。而其他一些虚拟用户如 `bin`、`daemon` 等被分配到一些比较小的 UID 号，这些用户通常被安排在 `passwd` 文件的开头部分。从一个比较大的数开始分配真实用户（如这里的 `john`）的 UID 号是一个好习惯，这样能为虚拟用户提供足够的余地。在笔者的系统上，真实用户的 UID 是从 1000 开始分配的。

应该保证每个用户 UID 号的唯一性。如果多个用户共有一个 UID，那么在诸如 NFS 这样的系统中将产生安全隐患。可以有多个用户的 UID 号均为 0，那么这些用户将同时拥有 `root` 权限。但通常不推荐这样做，这同样是出于安全方面的考虑。如果有这样的需求的话，应该使用类似于 `sudo` 这样的工具。

9.9.4 GID 号

GID 号用于在用户登录时指定其默认所在的组。和 UID 号一样，这是一个 32 位整数。组在 `/etc/group` 文件中定义，其中 `root` 组的 GID 号为 0。

在确定一个用户对某个文件是否具有访问权限时，系统会考察这个用户所在的所有组（在 `/etc/group` 文件中定义）。默认组 ID 只是在用户创建文件和目录时才有用。举例来说，`john` 同时属于 `john`、`students`、`workmates` 这 3 个组，默认组是 `john`。那么对于所有属于这

3 个组的文件和目录，john 都有权访问。当 john 新建了一个文件，那么这个文件所属的组就是 john。关于文件权限的详细讨论，参见 6.5 节。

9.10 进阶 2: /etc/shadow 文件

/etc/shadow 文件用于保存用户的口令，当然是使用加密后的形式。shadow 文件仅对 root 用户可读，这是为了保证用户口令的安全性。尽管所有的口令都经过加密，但让任何人都会有机会接触这些口令是非常危险的，如果口令不够强的话，完全可以通过暴力破解获取其加密前的形式。关于如何合理地选择和保管口令，参见 3.1 节。

和/etc/passwd 文件类似，/etc/shadow 文件的每一行代表一个用户，并以冒号分隔每一个字段。其中，只有用户名和口令字段是要求非空的。一条典型的记录如下：

```
mike:$1$F6003P9D$250FhpLPgsJINANs7j93Z0:14166:0:180:7::14974:
```

以下是各个字段的含义。

- ❑ 登录名；
- ❑ 加密后的口令；
- ❑ 上次修改口令的日期；
- ❑ 两次修改口令之间的天数（最少）；
- ❑ 两次修改口令之间的天数（最多）；
- ❑ 提前多少天提醒用户修改口令；
- ❑ 在口令过期多少天后禁用该账号；
- ❑ 账号过期的日期；
- ❑ 保留，目前为空。

以 mike 这个账号为例，mike 上次修改其口令是在 2008 年 10 月 14 日，口令必须在 180 天内再次修改。在口令失效前的 7 天，mike 会接到必须修改口令的警告。该账号将在 2010 年 12 月 31 日过期。

注意到在 shadow 文件中，绝对日期是从 1970 年 1 月 1 日至今的天数，这个时间很难计算，但总是可以使用 usermod 命令来设置过期字段（以 MM/DD/YY 的格式）。下面这条命令设置 mike 用户的过期日期为 2010 年 12 月 31 日。

```
$ sudo usermod -e 12/31/2010
```

9.11 进阶 3: /etc/group 文件

/etc/group 文件中保存有系统中所有组的名称，以及每个组中的成员列表。文件中的每一行表示一个组，由 4 个冒号分隔的字段组成。一条典型的记录如下。

```
admin:x:115:lewis,rescuer
```

以下是这 4 个字段的含义。

- ❑ 组名；
- ❑ 组口令占位符；
- ❑ 组 ID (GID) 号；
- ❑ 成员列表，用逗号分开（不能加空格）。

和 `passwd` 文件一样，如果口令字段为一个 `x` 的话，就表示还有一个 `/etc/gshadow` 文件用于存放组口令。但一般来说，组口令很少会用到，因此不必太在意这个字段。即便这个字段为空，也不需要担心安全问题。

GID 用于标识一个组。和 UID 一样，应该保证 GID 的唯一性。如果一个用户属于 `/etc/passwd` 中所指定的某个组，但没有出现在 `/etc/group` 文件相应的组中，那么应该以 `/etc/passwd` 文件中的设置为准。实际上，用户所属的组是 `passwd` 文件和 `group` 文件中相应组的并集。但为了管理上的有序性，应该保持两个文件一致。

9.12 小 结

- ❑ Linux 通过用户名和口令来验证用户的身份。
- ❑ 几个用户可以组成一个“用户组”。
- ❑ `useradd` 工具添加用户；`groupadd` 命令添加用户组。也可以使用图形化工具完成这些任务。
- ❑ `history` 命令查看用户在 Shell 中执行命令的历史记录。
- ❑ `userdel` 命令删除用户账号。
- ❑ `usermod` 命令修改已有的用户信息。
- ❑ `id` 命令查看特定用户的 UID、GID 及其所属的组。
- ❑ `su` 命令临时切换用户身份。不带任何参数的 `su` 命令切换到 `root` 身份，这是一种比较安全的使用 `root` 权限的方式。
- ❑ `sudo` 程序以更细的粒度分解系统特权。Ubuntu 只允许使用 `sudo`，而不能使用 `su`。
- ❑ UID 唯一标识系统中的用户，`root` 用户的 UID 为 0；类似地，GID 唯一标识系统中的用户组。
- ❑ 系统中的用户信息保存在 `/etc/passwd` 文件中，口令保存在 `/etc/shadow` 文件中。这两个文件应该妥善保管。
- ❑ `/etc/group` 文件保存系统中的组信息。