第 3 章

CHAPTER 3

Linux 基础知识

严格来讲，Linux 不算是一个操作系统，只是一个操作系统中的内核，即计算机软件与硬件之间通信的平台。Linux 的全称是 GNU/Linux，这才是一个真正意义上的操作系统。Linux 是一个多用户多任务的操作系统，也是一款自由软件，完全兼容 POSIX 标准，拥有良好的用户界面，支持多种处理器架构，移植方便。

本章是嵌入式 Linux 开发的前导章节，主要包括 Linux 的相关基础知识，包括 Linux 的常用命令、Linux 中 Shell 的应用、Linux 文件和目录管理、环境变量等相关知识。

3.1 Linux 和 Shell

在过去的 20 年里，Linux 系统主要被应用于服务器端、嵌入式开发和 PC 桌面三大领域。例如大型、超大型互联网企业（百度、腾讯、阿里等）都在使用 Linux 系统作为其服务器端的程序运行平台，全球及国内排名前 1000 的 90％以上的网站使用的主流系统都是 Linux 系统。

在所有 Linux 版本中，都会涉及以下几个重要概念。

内核：内核是操作系统的核心。内核直接与硬件交互，并处理大部分较低层的任务，如内存管理、进程调度、文件管理等。

命令和工具：日常工作中，用户会用到很多系统命令和工具，如 cp、mv、cat 和 grep 等。在 Linux 系统中，有 250 多个命令，每个命令都有多个选项。第三方工具也有很多，它们也扮演着重要角色。

文件和目录：Linux 系统中所有的数据都被存储到文件中，这些文件被分配到各个目录，构成文件系统。Linux 的目录与 Windows 的文件夹概念类似。

Shell：Shell 是一个处理用户请求的工具，它负责解释用户输入的命令，调用用户希望使用的程序。Shell 既是一种命令语言，又是一种程序设计语言。

接下来介绍 Shell。

用户通过 Shell 与 Linux 内核交互。Shell 是一个命令行解释工具，它将用户输入的命令转换为内核能够理解的语言（命令）。Shell 也是一种应用程序，这个应用程序提供了一个界面，用户通过这个界面访问操作系统内核的服务。Ken Thompson 的 sh 是第一种 UNIX Shell，Windows Explorer 就是一个典型的图形界面 Shell。

Shell 和 Shell 脚本是不一样的，Shell 脚本（Shell script），是一种为 Shell 编写的脚本程序。业界所说的 Shell 通常都是指 Shell 脚本。为简洁起见，本节的“Shell 编程”都是指 Shell 脚本编程，不是指开发 Shell 自身。

Shell 编程跟 JavaScript、php 编程一样，只要有一个能编写代码的文本编辑器和一个能解释执行的脚本解释器就可以了。

Linux 的 Shell 种类众多，常见的有：

Bourne Shell（/usr/bin/sh 或/bin/sh）

Bourne Again Shell（/bin/bash）

C Shell（/usr/bin/csh）

K Shell（/usr/bin/ksh）

Shell for Root（/sbin/sh）

…

本节关注的是 Bash，也就是 Bourne Again Shell，由于易用和免费，Bash 在日常工作中被广泛使用。同时，Bash 也是大多数 Linux 系统默认的 Shell。

在一般情况下，人们并不区分 Bourne Shell 和 Bourne Again Shell，所以，像#!/bin/sh，它同样也可以改为#! /bin/bash。其中符号#! 告诉系统其后路径所指定的程序即是解释此脚本文件的 Shell 程序。

接下来编写第一个 Shell 脚本。

打开文本编辑器（可以使用 vi/vim 命令来创建文件），新建一个文件 test.sh，扩展名为 sh（sh 代表 Shell），扩展名并不影响脚本执行。输入一些代码，第一行一般如下：

```
#!/bin/bash
echo "Hello World !"
```

#! 是一个约定的标记，它告诉系统这个脚本需要什么解释器来执行，即使用哪一种 Shell。Echo 命令用于向窗口输出文本。

运行 Shell 脚本有两种方法。

1）作为可执行程序

将第一个 Shell 脚本的代码保存为 test.sh，并切换到相应目录：

```
chmod +x ./test.sh                    #使脚本具有执行权限
./test.sh                             #执行脚本
```

注意，一定要写成./test.sh，而不是 test.sh，运行其他二进制的程序也一样，直接写 test.sh，Linux 系统会去 PATH 中寻找有没有 test.sh，而只有/bin、/sbin、/usr/bin、/usr/sbin 等在 PATH 中，因而用户的当前目录通常不在 PATH 中，所以写成 test.sh 无法找到命令，要用./test.sh 通知系统就在当前目录寻找。

2）作为解释器参数

这种运行方式是直接运行解释器，其参数就是 Shell 脚本的文件名，如：

```
/bin/sh test.sh
/bin/php test.php
```

这种方式运行的脚本，不需要在第一行指定解释器信息。

限于篇幅限制，这里不再对 Shell 进行详细介绍，希望获得更多 Shell 知识的读者可以去 Linux 官网或者社区学习。

3.2 常见 Linux 发行版本

在 Linux 内核的发展过程中，各种 Linux 发行版本起了巨大的作用，正是它们推动了 Linux 的应用，从而让更多的人开始关注 Linux。因此，把 Red Hat、Ubuntu、SUSE 等直接说成 Linux 其实是不确切的，它们是 Linux 的发行版本，更确切地说，应该叫作“以 Linux 为核心的操作系统软件包”。Linux 的各个发行版本使用的是同一个 Linux 内核，因此在内核层不存在兼容性问题。

Linux 的发行版本可以大体分为两类：商业公司维护的发行版本，以著名的 Red Hat 为代表。社区组织维护的发行版本，以 Debian 为代表。

接下来简要介绍主流 Linux 发行版本。

1. Red Hat Linux

Red Hat(红帽公司)创建于 1993 年，是目前世界上资深的 Linux 厂商，也是最获认可的 Linux 品牌。Red Hat 公司的产品主要包括 RHEL(Red Hat Enterprise Linux，收费版本)和 CentOS(RHEL 的社区克隆版本，免费版本)、Fedora Core(由 Red Hat 桌面版发展而来，免费版本)。Red Hat 是在我国国内使用人群最多的 Linux 版本，资料丰富，而且大多数 Linux 教程都是以 Red Hat 为基础来讲解的。

2. Ubuntu Linux

Ubuntu 基于知名的 Debian Linux 发展而来，界面友好，容易上手，对硬件的支持非常全面，是目前最适合作为桌面系统的 Linux 发行版本，而且 Ubuntu 的所有发行版本都是免费的。

3. SuSE Linux

SuSE Linux 以 Slackware Linux 为基础，原来是德国的 SuSE Linux AG 公司发布的 Linux 版本，1994 年发行了第一版，早期只有商业版本，2004 年被 Novell 公司收购后，成立了 OpenSuSE 社区，推出了自己的社区版本 OpenSuSE。SuSE Linux 在欧洲较为流行，在我国国内也有较多应用。SuSE Linux 可以非常方便地实现与 Windows 的交互，硬件检测非常优秀，拥有界面友好的安装过程、图形管理工具，对于终端用户和管理员来说使用非常方便。

4. Gentoo Linux

Gentoo 最初由 Daniel Robbins(FreeBSD 的开发者之一)创建，首个稳定版本发布于 2002 年。Gentoo 是所有 Linux 发行版本里安装最复杂的，到目前为止仍采用源代码包编译安装操作系统。不过，它是安装完成后最便于管理的版本，也是在相同硬件环境下运行最快的版本。自从 Gentoo 1.0 面世后给 Linux 世界带来了巨大的惊喜，同时也吸引了大量的用户和开发者投入 Gentoo Linux 的怀抱。尽管安装时可以选择预先编译好的软件包，但是大部分使用 Gentoo 的用户都选择自己手动编译。这也是为什么 Gentoo 适合比较有 Linux 使用经验的老手使用。

5. 其他 Linux 发行版

除以上 4 种 Linux 发行版外，还有很多其他版本，表 3-1 罗列了几种常见的 Linux 发行版及其特点。

表 3-1 Linux 发行版及其特点

版本名称	特　点	软件包管理器
Debian Linux	开放的开发模式，且易于进行软件包升级	apt
Fedora Core	拥有数量庞大的用户，优秀的社区技术支持，并且有许多创新	up2date(rpm)，yum(rpm)
CentOS	CentOS 是一种对 RHEL 源代码再编译的产物，由于 Linux 是开发源代码的操作系统，并不排斥基于源代码的再分发，CentOS 将商业的 Linux 操作系统 RHEL 进行源代码再编译后分发，并在 RHEL 的基础上修正了不少已知的漏洞	rpm
SuSE Linux	专业的操作系统，易用的 YaST 软件包管理系统	YaST(rpm)，第三方 apt(rpm)软件库(repository)
Mandriva	操作界面友好，使用图形配置工具，有庞大的社区进行技术支持，支持 NTFS 分区的大小变更	rpm
KNOPPIX	可以直接在 CD 上运行，具有优秀的硬件检测和适配能力，可作为系统的急救盘使用	apt
Gentoo Linux	高度的可定制性，使用手册完整	portage
Ubuntu	优秀的桌面环境，基于 Debian 构建	apt，dpkg，tasksel

3.3 Linux 文件管理

在 Linux 中很多工作都是通过命令完成的，学好 Linux，首先要掌握常用命令。本章结合常用命令来介绍 Linux 相关基础知识。

Linux 中的所有数据都被保存在文件中，所有的文件被分配到不同的目录。目录是一种类似于树的结构。当用户使用 Linux 时，大部分时间都会和文件打交道，通过本节可以了解基本的文件操作，如创建文件、删除文件、复制文件、重命名文件以及为文件创建链接等。

在 Linux 中，有 3 种基本的文件类型。

1. 普通文件

普通文件是以字节为单位的数据流，包括文本文件、源代码文件、可执行文件等。文本和二进制对 Linux 来说并无区别，对普通文件的解释由处理该文件的应用程序负责。

2. 目录

目录可以包含普通文件和特殊文件，目录相当于 Windows 和 macOS 中的文件夹。

3. 设备文件

Linux 与外部设备(例如光驱、打印机、终端、调制解调器等)是通过一种被称为设备文件的文件来进行通信的。Linux 与外部设备之间输入输出的关系和输入输出到一个文件的方式是相同的。在 Linux 和一个外部设备通信之前，这个设备必须首先要有一个设备文件

存在。例如，每一个终端都有自己的设备文件来供 Linux 写数据（比如出现在终端屏幕上）和读取数据（比如用户通过键盘输入）。

设备文件和普通文件不一样，设备文件中并不包含任何数据。最常见的设备文件有两种类型：字符设备文件和块设备文件。

字符设备文件以字母“c”开头。字符设备文件向设备传送数据时，一次传送一个字符。典型的通过字符传送数据的设备有打印机、绘图仪、调制解调器等。字符设备文件有时也被称为“raw”设备文件。

块设备文件以字母“b”开头。块设备文件向设备传送数据时，先从内存中的 buffer 中读或写数据，而不是直接传送数据到物理磁盘。磁盘和 CD-ROM 既可以使用字符设备文件，也可以使用块设备文件。

3.3.1 查看文件

查看当前目录下的文件和目录可以使用 ls 命令，例如：

```
$ ls
bin         hosts     lib       res.03
ch07        hw1       pub       test_results
ch07.bak    hw2       res.01    users
docs        hw3       res.02    work
```

通过 ls 命令的 -l 选项，可以获取更多文件信息，例如：

```
$ ls -l
total 1962188
drwxrwxr-x    2  amrood    amrood    4096        Dec 25 09:59    uml
-rw-rw-r--    1  amrood    amrood    5341        Dec 25 08:38    uml.jpg
drwxr-xr-x    2  amrood    amrood    4096        Feb 15  2021    univ
drwxr-xr-x    2  root      root      4096        Dec  9  2021    urlspedia
...
```

每一列的含义如下所示。

第一列：文件类型及文件的操作权限。

第二列：表示文件个数。如果是文件，那么就是 1。如果是目录，那么就是该目录中文件的数目。

第三列：文件的所有者，即文件的创建者。

第四列：文件所有者所在的用户组。在 Linux 中，每个用户都隶属于一个用户组。

第五列：文件大小（以字节计）。

第六列：文件被创建或上次被修改的时间。

第七列：文件名或目录名。

注意：每一个目录都有一个指向它本身的子目录“.”和指向它上级目录的子目录“..”，所以对于一个空目录，第二列应该为 2。

表 3-2 通过“ls-l”列出的文件，每一行都是以 a、d、-或 l 开头，这些字符表示文件类型。

表 3-2 字符前缀和文件类型

前　缀	描　述
-	普通文件。如文本文件、二进制可执行文件、源代码等
b	块设备文件。硬盘可以使用块设备文件
c	字符设备文件。硬盘也可以使用字符设备文件
d	目录文件。目录可以包含文件和其他目录
l	符号链接(软链接)。可以链接任何普通文件,类似于 Windows 中的快捷方式
p	具名管道。管道是进程间的一种通信机制
s	用于进程间通信的套接字

3.3.2 元字符

元字符是具有特殊含义的字符。比如 * 和? 都是元字符，* 可以匹配多个任意字符，而? 匹配一个字符。

例如：

```
$ ls ch*.doc
```

可以显示所有以 ch 开头、以.doc 结尾的文件。

```
ch01-1.doc   ch010.doc  ch02.doc    ch03-2.doc
ch04-1.doc   ch040.doc  ch05.doc    ch06-2.doc
ch01-2.doc   ch02-1.doc c
```

如果希望显示所有以 .doc 结尾的文件,则可以使用：

```
$ ls *.doc
```

3.3.3 隐藏文件

隐藏文件的第一个字符为英文句号或点号(.),Linux 程序(包括 Shell)通常使用隐藏文件来保存配置信息。下面是一些常见的隐藏文件。

.profile：Bourne Shell(sh)初始化脚本。

.kshrc：Korn Shell(ksh)初始化脚本。

.cshrc：C Shell(csh)初始化脚本。

.rhosts：Remote Shell(rsh)配置文件。

查看隐藏文件需要使用 ls 命令的 -a 选项。

```
$ ls -a
.          .profile      docs     lib     test_results
..         .rhosts       hosts    pub     users
.emacs     bin           hw1      res.01  work
.exrc      ch07          hw2      res.02
.kshrc     ch07.bak      hw3      res.03
$
```

与 3.3.1 节的介绍一样，一个点号(.)表示当前目录，两个点号(..)表示上级目录。
注意：输入密码时，星号(*)作为占位符，代表输入的字符个数。

3.3.4 查看文件内容

可以使用 cat 命令来查看文件内容，下面是一个简单的例子。

```
$ cat filename
This is Linux file....I created it for the first time.....
I'm going to save this content in this file.
$
```

可以通过 cat 命令的 -b 选项来显示行号，例如：

```
$ cat -b filename
1    This is Linux file....I created it for the first time.....
2    I'm going to save this content in this file.
$
```

3.3.5 统计单词数目

可以使用 wc 命令来统计当前文件的行数、单词数和字符数，下面是一个简单的例子。

```
$ wc filename
  2  19  103  filename
$
```

第一列：文件的总行数。第二列：单词数目。第三列：文件的字节数，即文件的大小。第四列：文件名。也可以一次查看多个文件的内容，例如：

```
$ wc filename1 filename2 filename3
```

3.3.6 复制文件

可以使用 cp 命令来复制文件，cp 命令的基本语法如下：

```
$ cp source_file destination_file
```

下面的例子将会复制 filename 文件。

```
$ cp filename copyfile
$
```

现在在当前目录中会多出一个和 filename 一模一样的 copyfile 文件。

3.3.7 重命名文件

重命名文件可以使用 mv 命令，语法为：

```
$ mv old_file new_file
```

下面的例子将会把 filename 文件重命名为 newfile。

```
$ mv filename newfile
$
```

现在在当前目录下，只有一个 newfile 文件。mv 命令其实是一个移动文件的命令，不但可以更改文件的路径，也可以更改文件名。

3.3.8 删除文件

rm 命令可以删除文件，语法为：

```
$ rm filename
```

注意：删除文件是一种危险的行为，因为文件内可能包含有用信息，建议结合-i 选项来使用 rm 命令。

下面的例子会彻底删除一个文件。

```
$ rm filename
$
```

也可以一次删除多个文件。

```
$ rm filename1 filename2 filename3
$
```

3.4 Linux 目录

目录也是一个文件，它的功能是用来保存文件及其相关信息。所有的文件，包括普通文件、设备文件和目录文件，都会被保存到目录中。

3.4.1 主目录

登录后，用户所在的位置就是主目录(或登录目录)，接下来主要是在这个目录下进行操作，如创建文件、删除文件等。使用下面的命令可以随时进入主目录。

```
$ cd ~
$
```

这里～就表示主目录。如果希望进入其他用户的主目录，可以使用下面的命令。

```
$ cd ~username
$
```

返回进入当前目录前所在的目录可以使用下面的命令。

```
$ cd -
$
```

3.4.2 绝对路径和相对路径

Linux 的目录有清晰的层次结构，/代表根目录，所有的目录都位于/下面。文件在层次结构中的位置可以用路径来表示。

如果一个路径以/开头，则称为绝对路径。它表示当前文件与根目录的关系。举例如下：

```
/etc/passwd
/users/sjones/chem/notes
/dev/rdsk/0s3
```

不以/开头的路径称为相对路径，它表示文件与当前目录的关系。例如：

```
chem/notes
personal/res
```

获取当前所在的目录可以使用 pwd 命令。

```
$ pwd
/user0/home/amrood
$
```

查看目录中的文件可以使用 ls 命令。

```
$ ls dirname
```

下面的例子将遍历/usr/local 目录下的文件。

```
$ ls /usr/local

X11       bin          gimp       jikes       sbin
ace       doc          include    lib         share
atalk     etc          info       man         ami
```

3.4.3 创建目录

可以使用 mkdir 命令来创建目录，语法为：

```
$ mkdir dirname
```

dirname 可以为绝对路径，也可以为相对路径。例如：

```
$ mkdir mydir
$
```

该命令会在当前目录下创建 mydir 目录。又如：

```
$ mkdir /tmp/test-dir
$
```

该命令会在 /tmp 目录下创建 test-dir 目录。mkdir 成功创建目录后不会输出任何信息。

也可以使用 mkdir 命令同时创建多个目录，例如：

```
$ mkdir docs pub
$
```

该命令会在当前目录下创建 docs 和 pub 两个目录。

使用 mkdir 命令创建目录时，如果上级目录不存在，就会报错。在下面的例子中，mkdir 会输出错误信息。

```
$ mkdir /tmp/amrood/test
mkdir: Failed to make directory "/tmp/amrood/test";
No such file or directory
$
```

为 mkdir 命令增加-p 选项，可以逐级创建所需要的目录，即使上级目录不存在也不会报错。例如：

```
$ mkdir -p /tmp/amrood/test
$
```

该命令会创建所有不存在的上级目录。

3.4.4 删除目录

可以使用 rmdir 命令来删除目录，例如：

```
$ rmdir dirname
$
```

注意：删除目录时请确保目录为空，不会包含其他文件或目录。

也可以使用 rmdir 命令同时删除多个目录。

```
$ rmdir dirname1 dirname2 dirname3
$
```

如果 dirname1、dirname2、dirname3 为空，则会被删除。rmdir 成功删除目录后不会输出任何信息。

3.4.5 改变所在目录

可以使用 cd 命令来改变当前所在目录，进入任何有权限的目录，语法为：

```
$ cd dirname
```

其中，dirname 为路径，既可以为相对路径，也可以为绝对路径。例如：

```
$ cd /usr/local/bin
$
```

可以进入/usr/local/bin 目录。可以使用相对路径从这个目录进入/usr/home/amrood 目录。

```
$ cd ../../home/amrood
$
```

3.4.6 重命名目录

mv(move)命令也可以用来重命名目录,语法为:

```
$ mv olddir newdir
```

下面的例子将会把 mydir 目录重命名为 yourdir 目录。

```
$ mv mydir yourdir
$
```

3.5 Linux 文件权限和访问模式

为了更加安全地存储文件,Linux 为不同的文件赋予了不同的权限,每个文件都拥有下面 3 种权限。

- 所有者权限:文件所有者能够进行的操作。
- 组权限:文件所属用户组能够进行的操作。
- 外部权限(其他权限):其他用户可以进行的操作。

3.5.1 查看文件权限

使用“ls-l”命令可以查看与文件权限相关的信息。

```
$ ls -l /home/amrood
-rwxr-xr--   1 amrood    users 1024  Nov 2 00:10  myfile
drwxr-xr--- 1 amrood    users 1024  Nov 2 00:10  mydir
```

第一列包含了文件或目录的权限。第一列的第一个字符代表文件类型,-代表是普通文件,d 代表是文件夹。而接下来的字符所对应的权限一共分成 3 组,3 个一组,分别属于文件所有者、文件所属用户组和其他用户。权限中的每个字符都代表不同的权限,其中分别为读取(r)、写入(w)和执行(x)。

第一组字符(2~4)表示文件所有者的权限,-rwxr-xr- -表示所有者拥有读取(r)、写入(w)和执行(x)的权限。

第二组字符(5~7)表示文件所属用户组的权限,-rwxr-xr- -表示该组拥有读取(r)和执行(x)的权限,但没有写入权限。

第三组字符(8~10)表示所有其他用户的权限,rwxr-xr- -表示其他用户只能读取(r)文件。

3.5.2 文件访问模式

文件权限是 Linux 系统的第一道安全防线，基本的权限有读取(r)、写入(w)和执行(x)。

读取：用户能够读取文件信息，查看文件内容。

写入：用户可以编辑文件，可以向文件写入内容，也可以删除文件内容。

执行：用户可以将文件作为程序来运行。

3.5.3 目录访问模式

目录的访问模式和文件类似，但是稍有不同。

读取：用户可以查看目录中的文件。

写入：用户可以在当前目录中删除文件或创建文件。

执行：执行权限赋予用户遍历目录的权利，例如，执行 cd 和 ls 命令。

3.5.4 改变权限

可以使用 chmod(change mode)命令来改变文件或目录的访问权限，权限可以使用符号或数字来表示。

1. 使用符号表示权限

对于初学者来说，最简单的就是使用符号来改变文件或目录的权限，用户可以增加(+)和删除(-)权限，也可以指定特定权限。表 3-3 列举了权限更改符号。

表 3-3 权限更改符号

符　号	说　明
+	为文件或目录增加权限
-	删除文件或目录的权限
=	设置指定的权限

下面的例子将会修改 testfile 文件的权限。

```
$ ls -l testfile
-rwxrwxr--      1 amrood     users 1024     Nov 2 00:10     testfile
$ chmod o+wx testfile
$ ls -l testfile
-rwxrwxrwx      1 amrood     users 1024     Nov 2 00:10     testfile
$ chmod u-x testfile
$ ls -l testfile
-rw-rwxrwx      1 amrood     users 1024     Nov 2 00:10     testfile
$ chmod g=rx testfile
$ ls -l testfile
-rw-r-xrwx     1 amrood     users 1024     Nov 2 00:10     testfile
```

也可以同时使用多个符号。

```
$ chmod o+wx,u-x,g=rx testfile
$ ls -l testfile
-rw-r-xrwx   1 amrood    users 1024   Nov 2 00:10   testfile
```

2. 使用数字表示权限

除了符号，也可以使用八进制数字来指定具体权限，如表 3-4 所示。

表 3-4 使用数字表示权限

数 字	说 明	权 限
0	没有任何权限	---
1	执行权限	--x
2	写入权限	-w-
3	执行权限和写入权限：1(执行)+2(写入)=3	-wx
4	读取权限	r--
5	读取和执行权限：4(读取)+1(执行)=5	r-x
6	读取和写入权限：4(读取)+2(写入)=6	rw-
7	所有权限：4(读取)+2(写入)+1(执行)=7	rwx

下面的例子，首先使用“ls-l”命令查看 testfile 文件的权限，然后使用 chmod 命令更改权限。

```
$ ls -l testfile
-rwxrwxr--     1 amrood    users 1024    Nov 2 00:10    testfile
$ chmod 755 testfile
$ ls -l testfile
-rwxr-xr-x     1 amrood    users 1024    Nov 2 00:10    testfile
$ chmod 743 testfile
$ ls -l testfile
-rwxr---wx     1 amrood    users 1024    Nov 2 00:10    testfile
$ chmod 043 testfile
$ ls -l testfile
----r---wx     1 amrood    users 1024    Nov 2 00:10    testfile
```

3.5.5 更改所有者和用户组

在 Linux 中，每添加一个新用户，就会为它分配一个用户 ID 和群组 ID，上面提到的文件权限也是基于用户和群组来分配的。

有如下两个命令可以改变文件的所有者或群组。

chown：chown 命令是 change owner 的缩写，用来改变文件的所有者。

chgrp：chgrp 命令是 change group 的缩写，用来改变文件所在的群组。

chown 命令用来更改文件所有者，其语法如下。

```
$ chown user filelist
```

user 可以是用户名或用户 ID，例如：

```
$ chown amrood testfile
$
```

该命令将 testfile 文件的所有者改为 amrood。

注意：超级用户 root 可以不受限制地更改文件的所有者和用户组，但是普通用户只能更改所有者是自己的文件或目录。

chgrp 命令用来改变文件所属群组，其语法为：

```
$ chgrp group filelist
```

group 可以是群组名或群组 ID，例如：

```
$ chgrp special testfile
$
```

上述命令将文件 testfile 的群组改为 special。

在 Linux 中，一些程序需要特殊权限才能完成用户指定的操作。例如，用户的密码保存在/etc/shadow 文件中，出于安全考虑，一般用户没有读取和写入的权限。但是当我们使用 passwd 命令来更改密码时，需要对/etc/shadow 文件有写入权限。这就意味着，passwd 程序必须要给我们一些特殊权限，才可以向/etc/shadow 文件写入内容。

Linux 通过给程序设置 SUID(Set User ID)和 SGID(Set Group ID)位来赋予普通用户特殊权限。当运行一个带有 SUID 位的程序时，就会继承该程序所有者的权限。如果程序不带 SUID 位，则会根据程序使用者的权限来运行。

SGID 也是一样。一般情况下程序会根据用户的组权限来运行，但是给程序设置 SGID 后，就会根据程序所在组的组权限运行。

如果程序设置了 SUID 位，则会在表示文件所有者可执行权限的位置上出现's'。同样，如果设置了 SGID，则会在表示文件群组可执行权限的位置上出现's'。如下所示：

```
$ ls -l /usr/bin/passwd
-r-sr-xr-x  1   root   bin  19031 Feb 7 13:47  /usr/bin/passwd*
$
```

执行命令后的第四个字符不是'x'或'-'，而是's'，说明 /usr/bin/passwd 文件设置了 SUID 位，这时普通用户会以 root 用户的权限来执行 passwd 程序。

注意：小写字母's'说明文件所有者有执行权限(x)，大写字母'S'说明程序所有者没有执行权限(x)。

如果在表示群组权限的位置上出现 SGID 位，那么也仅有 3 类用户可以删除该目录下的文件：目录所有者、文件所有者、超级用户 root。

为一个目录设置 SUID 和 SGID 位可以使用下面的命令。

```
$ chmod ug+s dirname
$ ls -l
drwsr-sr-x 2 root root  4096 Jun 19 06:45 dirname
$
```

3.6 Linux 环境变量

在 Linux 中，环境变量是一个很重要的概念。环境变量可以由系统、用户、Shell 及其他程序来设定。这里变量就是一个可以被赋值的字符串，赋值范围包括数字、文本、文件名、设

备及其他类型的数据。

下面的例子将为变量 TEST 赋值，然后使用 echo 命令输出。

```
$ TEST = "Linux Programming"
$ echo $ TEST
Linux Programming
```

注意：变量赋值时前面不能加 $ 符号，变量输出时必须要加 $ 前缀。退出 Shell 时，变量将消失。

登录系统后，Shell 会有一个初始化的过程，用来设置环境变量。这个阶段，Shell 会读取/etc/profile 和. profile 两个文件，过程如下：

Shell 首先检查/etc/profile 文件是否存在，如果存在，则读取内容，否则就跳过，但是不会报错。

然后检查主目录(登录目录)中是否存在. profile 文件，如果存在，就读取内容，否则就跳过，也不会报错。

读取完上面两个文件，Shell 就会出现 $ 命令提示符。

```
$
```

在这个提示符后就可以输入命令并调用相应的程序了。注意：上面是 Bourne Shell 的初始化过程，bash 和 ksh 在初始化过程中还会检查其他文件。

3.6.1 .profile 文件

/etc/profile 文件包含了通用的 Shell 初始化信息，由 Linux 管理员维护，一般用户无权修改。但是用户可以修改主目录下的. profile 文件，增加一些特定初始化信息，包括：设置默认终端类型和外观样式；设置 Shell 命令查找路径，即 PATH 变量；设置命令提示符等。

3.6.2 设置终端类型

一般情况下，用户使用的终端是由 login 或 getty 程序设置的，可能会不符合用户的习惯。对于没有使用过的终端，可能会比较生疏，不习惯命令的输出样式，交互起来略显吃力。所以，一般用户会将终端设置成下面的类型。

```
$ TERM = vt100
$
```

vt100 是 virtual terminate 100 的缩写。vt100 是被绝大多数 Linux 系统所支持的一种虚拟终端规范，常用的还有 ansi、xterm 等。

3.6.3 设置 PATH 变量

在命令提示符下输入一个命令时，Shell 会根据 PATH 变量来查找该命令对应的程序，PATH 变量指明了这些程序所在的路径。

一般情况下，PATH 变量的设置如下：

```
$ PATH = /bin:/usr/bin
$
```

多个路径使用冒号(：)分隔。如果用户输入的命令在 PATH 设置的路径下没有找到，就会报错，例如：

```
$ hello
hello: not found
$
```

3.6.4 PS1 和 PS2 变量

PS1 变量用来保存命令提示符，可以随意修改，如果用户不习惯使用 $ 作为提示符，也可以改成其他字符。PS1 变量被修改后，提示符会立即改变。

例如，把命令提示符设置成'=>'。

```
$ PS1 = '=>'
=>
```

也可以将提示信息设置成当前目录，例如：

```
=> PS1 = "[\u@\h \w]\ $ "
[root@ip - 72 - 167 - 112 - 17 /var/www/tutorialspoint/Linux] $
```

命令提示信息包含了用户名、主机名和当前目录。表 3-5 中的转义字符可以被用作 PS1 的参数，以丰富命令提示符信息。

表 3-5 转义字符

转 义 字 符	描　　述
\t	当前时间，格式为 HH：MM：SS
\d	当前日期，格式为 Weekday Month Date
\n	换行
\W	当前所在目录
\w	当前所在目录的完整路径
\u	用户名
\h	主机名(IP 地址)
#	输入的命令的个数，每输入一个新的命令就会加 1
\ $	如果是超级用户 root，提示符为 #，否则为 $

用户可以在每次登录的时候修改提示符，也可以在. profile 文件中增加 PS1 变量，这样每次登录时会自动修改提示符。如果用户输入的命令不完整，则 Shell 会使用第二提示符来等待用户完成命令的输入。默认的第二命令提示符是>，保存在 PS2 变量中，可以随意修改。

下面的例子使用默认的第二命令提示符。

```
$ echo "this is a
> test"
```

```
this is a
test
$
```

下面的例子通过 PS2 变量改变提示符。

```
$ PS2 = "secondary prompt - >"
$ echo "this is a
secondary prompt - > test"
this is a
test
$
```

3.6.5 常用环境变量

表 3-6 列出了部分重要的环境变量，这些变量可以通过 3.6.4 节提到的方式修改。

表 3-6 部分重要的环境变量

变 量	描 述
DISPLAY	用来设置将图形显示到何处
HOME	当前用户的主目录
IFS	内部域分隔符
LANG	LANG 可以让系统支持多语言。例如，将 LANG 设为 pt_BR，则可以支持(巴西)葡萄牙语
PATH	指定 Shell 命令的路径
PWD	当前所在目录，即 cd 命令查到的目录
RANDOM	生成一个 0～32 767 范围内的随机数
TERM	设置终端类型
TZ	时区。可以是 AST(大西洋标准时间)或 GMT(格林尼治标准时间)等
UID	以数字形式表示的当前用户 ID，Shell 启动时会被初始化

下面的例子中使用了部分环境变量。

```
$ echo $HOME
/root
]$ echo $DISPLAY

$ echo $TERM
xterm
$ echo $PATH
/usr/local/bin:/bin:/usr/bin:/home/amrood/bin:/usr/local/bin
$
```

3.7 Linux yum 命令

Linux yum(yellow dog updater，modified)是一个在 Fedora 和 RedHat 及 SUSE 中的 Shell 前端软件包管理器。

Linux yum 基于 RPM 包管理，能够从指定的服务器自动下载 RPM 包并且安装，可以自动处理依赖性关系，并且一次安装所有依赖的软件包，无须重复下载、安装。

Linux yum 提供了查找、安装、删除某一个或一组甚至全部软件包的命令，而且命令简洁。

Linux yum 语法如下所示。

```
yum [options] [command] [package ...]
```

其中，options：可选，选项包括-h(帮助)、-y(当安装过程提示选择全部为 yes)、-q(不显示安装的过程)。command：要进行的操作。package：安装的包名。

Linux yum 常用命令如下所述。

(1) 列出所有可更新的软件清单命令：yum check-update。

(2) 更新所有软件命令：yum update。

(3) 仅安装指定的软件命令：yum install < package_name >。

(4) 仅更新指定的软件命令：yum update < package_name >。

(5) 列出所有可安装的软件清单命令：yum list。

(6) 删除软件包命令：yum remove < package_name >。

(7) 查找软件包命令：yum search < keyword >。

(8) 清除缓存命令包括：

- yum clean packages——清除缓存目录下的软件包。
- yum clean headers——清除缓存目录下的 headers。
- yum clean oldheaders——清除缓存目录下旧的 headers。
- yum clean，yum clean all(＝yum clean packages；yum clean oldheaders)——清除缓存目录下的软件包及旧的 headers。

3.8 Linux apt 命令

Linux apt(advanced packaging tool)是一个在 Debian 和 Ubuntu 中的 Shell 前端软件包管理器。

Linux apt 命令提供了查找、安装、升级、删除某一个或一组甚至全部软件包的命令，而且命令十分简洁。

Linux apt 命令执行需要超级管理员权限(root)。Linux sudo 是 Linux 系统管理指令，是允许普通用户执行一些或者全部 root 命令的一个工具，如 halt(关闭系统)、reboot(重启系统)、su(变更使用者身份)等。这样不仅减少了 root 用户的登录和管理时间，也提高了安全性。

Linux apt 语法如下：

```
apt [options] [command] [package...]
```

参数设置和 yum 相同。Linux apt 常用命令如下所述。

(1) 列出所有可更新的软件清单：sudo apt update。

(2) 升级软件包：sudo apt upgrade。

(3) 列出可更新的软件包及版本信息：apt list-upgradeable。

(4) 升级软件包，升级前先删除需要更新的软件包：sudo apt full-upgrade。

(5) 安装指定的软件：sudo apt install < package_name >。

(6) 安装多个软件包：sudo apt install < package_1 > < package_2 > < package_3 >。

(7) 更新指定的软件：sudo apt update < package_name >。

(8) 显示软件包具体信息(例如，版本号、安装大小、依赖关系等)：sudo apt show < package_name >。

(9) 删除软件包：sudo apt remove < package_name >。

(10) 清理不再使用的依赖和库文件：sudo apt autoremove。

(11) 移除软件包及配置文件：sudo apt purge < package_name >。

(12) 查找软件包：sudo apt search < keyword >。

(13) 列出所有已安装的软件包：apt list-installed。

(14) 列出所有已安装的软件包的版本信息：apt list --all-versions。

3.9 本章小结

本章是为后续嵌入式 Linux 开发起到铺垫作用的章节。由于 Linux 开放源代码、易于移植、资源丰富、免费等优点，使得它在服务器和 PC 桌面之外的嵌入式领域越来越流行。更重要的一点，由于嵌入式 Linux 与 PC Linux 源于同一套内核代码，只是裁剪的程度不一样，这使得很多为 PC 开发的软件再次编译之后，可以直接在嵌入式设备上运行。本章介绍了 Linux 的很多基础命令和组成单元，但是 Linux 本身是代码数量在千万行之上的庞大操作系统，资源非常丰富，本章只着重介绍了在嵌入式 Linux 领域中会用到的 Linux 相关知识。

习题

1. 查阅相关资料，进一步了解 Linux 的主要发行版本之间的异同。
2. 尝试下载典型 Linux 发行版本并使用常用的 Linux 命令。
3. 尝试阅读下载的 Linux 发行版本的环境变量。