嵌入式开发直通车

# Linux Shell 编程艺术

## The Art of Linux Shell Scripting

张 泽 编著

195个实际案例+195个解决方案，

超大容量多媒体语音教学视频赠送，
总时长超过34小时

一线开发人员Linux Shell脚本编程
经验倾囊相授，帮你瞬间提升实战能力

含DVD光盘1张

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY [ http://www.phei.com.cn ]

嵌入式开发直通车

# Linux Shell 编程艺术

## The Art of Linux Shell Scripting

张 泽 编著

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

## 内 容 简 介

Shell 是 Unix/Linux 操作系统与用户进行交互的重要接口，是 Unix/Linux 系统中最重要的软件之一。一直以来，Shell 编程是系统管理员必备的高级技能，通过学习 Shell 编程，可以让计算机系统完成那些繁重并且琐碎的管理任务，从而可以节省大量的工作时间。本书通过大量的实例，以循序渐进的方式，由浅入深地逐步介绍 Shell 编程的各个知识点，从而引领读者轻松跨越 Shell 程序设计的门槛，最终摆脱繁忙而且低效率的工作状态，达到轻松管理系统的目的。

本书结合大量的实例，系统、全面地介绍了 Shell 脚本编程语言的语法格式、常用命令的使用、面对问题的分析方法，以及整个系统背后的运行原理等内容，力求使读者掌握从问题分析到代码实现，再到调试脚本、优化脚本的整个流程。学习完本书以后，读者能够具备较强的 Shell 程序设计能力，并对系统背后的运行原理有深入的理解。

本书适合系统管理员、网络管理员、Unix/Linux 系统爱好者学习参考。



**图书在版编目（CIP）数据**

Linux Shell 编程艺术 / 张泽编著. 北京：电子工业出版社，2014.1
（嵌入式开发直通车）
ISBN 978-7-121-22101-9

Ⅰ. ①L… Ⅱ. ①张… Ⅲ. ①Linux 操作系统－程序设计 Ⅳ. ①TP316.89

中国版本图书馆 CIP 数据核字（2013）第 294953 号

策划编辑：王敬栋（wangjd@phei.com.cn）
责任编辑：谭丽莎
印　　刷：三河市鑫金马印装有限公司
装　　订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787×1092　1/16　印张：29.75　字数：749 千字
印　　次：2014 年 1 月第 1 次印刷
印　　数：3 000 册　　定价：88.00 元（含 DVD 光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

# 前　言

Shell 是 Unix/Linux 操作系统与用户进行交互的重要接口，是 Unix/Linux 系统中最重要的软件之一。一直以来，Shell 编程是系统管理员必备的高级技能，通过学习 Shell 编程，可以让计算机系统完成那些繁重并且琐碎的管理任务，从而可以节省大量的工作时间。本书通过大量的实例，以循序渐进的方式，由浅入深地逐步介绍 Shell 编程的各个知识点，从而引领读者轻松跨越 Shell 程序设计的门槛，最终摆脱繁忙而且低效率的工作状态，达到轻松管理系统的目的。

如果你已经是一个 Unix/Linux 系统的使用者，或者希望将来能够从事 Unix/Linux 系统方面的工作，那么本书一定是你所需要的。本书并不要求你有多少系统管理方面的经验，如果你已经能够在命令行使用简单的命令，那么马上就可以上手编写脚本程序；如果你没有这方面的经验也不必担心，在本书的开始会给出这方面的介绍，一步一步带你进入精彩的脚本世界。

## 本书内容

本书结合大量的实例，系统、全面地介绍了 Shell 脚本编程语言的语法格式、常用命令的使用、面对问题的分析方法，以及整个系统背后的运行原理等内容，力求使读者掌握从问题分析到代码实现，再到调试脚本、优化脚本的整个流程。学习本书以后，读者能够具备较强的 Shell 程序设计能力，并对系统背后的运行原理有深入的理解。本书适合系统管理员、网络管理员、Unix/Linux 系统爱好者学习参考。

本书的主要内容如下。

（1）第 1 章 Shell 介绍和脚本基础：讨论了有关 Shell 编程的一些重要概念，以及 Shell 编程所使用的基本工具和基本方法。

（2）第 2 章操作文件和目录：介绍了在 Unix/Linux 系统中操作文件和目录所使用的各种命令工具，如对文件和目录的复制、移动等，同时还讲述了文件和目录访问权限的有关知识。

（3）第 3 章输入/输出重定向和管道：讲述了系统中所有命令的共同点，即每一个命令都会有标准输入、标准输出和标准错误输出，同时还讲述了如何通过管道来改变这些输入/输出。

（4）第 4 章进程：讲述了如何启动和终止一个进程，以及前台进程与后台进程的区别。

（5）第 5 章文本处理和字符串操作：介绍了文本处理的相关工具和方法，这些工具与

管道结合在一起使用可以构成强大而复杂的功能。

（6）第 6 章变量：讲解了变量在 Shell 脚本程序中的作用及相关的使用方法，如如何创建和删除变量，此外还介绍了有关环境变量的知识。

（7）第 7 章流程控制：介绍了在 Shell 编程中所使用的各种流程控制语句，包括各种分支结构和循环结构。

（8）第 8 章替换：介绍了 4 种类型的替换及它们在脚本程序中所发挥的作用，它们分别是文件名替换、变量替换、命令替换及算术替换。

（9）第 9 章引用：讲解了引用的概念和各种引用的使用方法，以及它们之间的区别，同时还讲述了引用对于各种替换的效果。

（10）第 10 章函数：讨论了有关脚本中函数的用法。通过使用函数可以使脚本程序更加结构化，更加模块化，同时可以大大地提高脚本程序的开发效率。

（11）第 11 章其他常用工具：介绍了在系统管理及开发脚本过程中经常用到的一些工具，如 find 命令、xargs 命令、下载工具和数据文件的压缩备份工具。

（12）第 12 章处理信号：阐述了系统中信号的概念，同时讲解了如何发送信号及如何使用 trap 命令处理各种信号。

（13）第 13 章使用 sed 处理文本：讲述了正则表达式的基本知识，同时还结合正则表达式讲解了强大的流编辑器 sed。

（14）第 14 章使用 awk 处理文本：讲解了另外一个强大的文本过滤工具 awk，并且对 awk 与流编辑器 sed 进行了比较。

（15）第 15 章 grep 和高级正则表达式：介绍了文本内容搜索工具 grep 命令，并通过 grep 命令继续介绍了正则表达式的很多高级使用方法。

（16）第 16 章 Debug Shell 脚本：讲解了在 Shell 脚本的开发过程中经常会犯的一些错误，以及如何利用 Shell 所产生的错误信息来确定问题的位置，同时还讨论了 Shell 内建的各种 debug 工具，以及在 debug 过程中经常使用的一些方法。

## 如何阅读本书

本书试图通过实际案例和现场解答的方式，讲述 Shell 脚本编程过程中的各个知识点。读者在遇到每一个实际的需求案例时，可以先不直接查看解决方案，而是通过自己已有的知识结构看是否可以解决所面临的问题；也可以先查询系统中的手册，看是否能够找到解决问题的方法，然后再与作者给出的解决方案进行对比，这样做的好处是不仅加深了对知识点的理解，还可以锻炼读者分析问题、解决问题的能力。在计算机的世界里，你随时都可能遇到各种各样的问题，并不是所有的问题都可以在书本中找到答案，因此我们需要的是能够分析问题，对已有知识灵活运用并举一反三的能力，这也是本书作者最为看重的一点。因此，在每一个实际案例的分析中，本书都会从开发者面对当时问题的思考过程的角度来进行描述，一步步引领读者学会这种开发过程中的思维过程。

此外，由于 Shell 编程中的各部分知识是相互交织缠绕在一起的，本书中各章节的前后顺序只是作者认为相对比较合理的一条学习路径，所以难免会出现前面的知识要用到后面知识的情况，此时读者可以适当地“囫囵吞枣”，只要能够理解一段代码所实现的大体功能即

可，而不必在第一次阅读时就搞懂每一个细节，可以在后面专门介绍这个知识点的章节中再深入地学习。总之，如果眼前有不懂或想不明白的问题，可以暂时先忽略它，等到一段时间以后，这些问题自然会得到解决。

在本书中，读者可能会觉得有些很“简单”的问题都被列了出来，甚至显得有些啰嗦。这是因为对有些读者来说某一个问题可能很简单，但是另外一些读者可能对这部分知识比较生疏，从而对这个问题怎么也想不通，因此笔者宁可将内容写得啰嗦点，也不让一些重要的细节被遗漏。但是如果涉及的细节在前面的章节已经讲过，便会有意地略过。

本书的代码示例力求简单易懂，目的是为了尽可能地讲清楚每一个知识点，因此有些实例程序在现实脚本世界中是不会存在的，它们仅仅是为了学习之用，暂时并不考虑实际的用途。当然，如果读者有更好的实例程序，也欢迎与作者进行沟通。

## 实例程序的获得和使用

本书中的实例代码可以通过登录华信教育资源网（www.hexdu.com.cn）免费注册后下载获得，下载以后得到的是一个源代码程序的 zip 压缩包，在 Linux 系统中使用解压工具解开以后，可以看到所有的程序实例都按章节进行了分类。读者在学习某一章时，可以进入相应的章节目录，阅读并执行实例程序。同时，在本书中引用源代码实例的地方，也都用[filename.sh]的形式标识了对应的程序文件名，以方便读者测试使用。对于那些在 Shell 命令行输入的命令，并没有对应的源代码，需要读者自己在 Linux 系统中输入这些命令来进行验证。

## 致谢

本书由张泽编著，在本书完成之际，感谢我的朋友和家人的支持和鼓励，是他们让我一直坚持下来，并最终完成这部著作。

参加本书编写和审定工作的还有王坚宁、李龙、魏勇、张玉兰、高克臻、张秀梅、张云霞、周兴国、李辉、刘峰、徐浩、马建军、朱丽云、许小荣等。在此，编者对以上人员致以诚挚的谢意！

## 疑问与讨论

由于作者的水平有限，加上写作时间仓促，书中难免会出现一些错误或不准确的地方，恳请读者批评和指正。如果有任何疑问或错误，可以通过 Email 与笔者共同讨论。笔者的联系邮箱：zhangze.linux@gmail.com。

编　者

2013 年 3 月

# 目　录

# 第 1 章

# Shell 介绍和脚本基础

操作系统通常需要采用某种方式来接受用户的指令，因此它需要一个和用户交互的接口。我们所熟悉的 Windows 操作系统是通过完善的图形用户界面（GUI）来与用户交互的，此外它也有接收命令的命令行接口，而本书所要讲述的是在 Unix/Linux 环境中操作系统与用户交互的接口，这个接口就是 Unix/Linux 系统中的 Shell 程序。现在的 Unix/Linux 操作系统提供了很强大的图形用户界面，只要单击鼠标就可以非常方便地完成很多操作，但是不论操作系统的图形用户界面多么完善，也不能完全替代命令行的作用，因为总有相当一部分的操作在图形界面下无法执行，或者在 Shell 命令行下执行会更有效率，因此能够熟练地使用 Shell 和系统命令是管理系统所必需的。除此以外，在管理远程服务器方面，使用 Shell 命令进行远程操作也会更加容易。本章主要介绍有关 Shell 编程的一些概念和基础知识。

## 本章要学习的知识点

（1）Shell 在系统中的作用；
（2）用户登录系统的过程；
（3）如何编辑和执行命令；
（4）在哪里可以找到命令的帮助信息；
（5）执行脚本的几种方式及它们的区别；
（6）vim 编辑器的常用操作；
（7）如何登录远程服务器运行 Shell 脚本。

## 1.1 登录到系统

### 1. 需求 1

Unix/Linux 系统是一个多用户的操作系统，在使用系统以前，需要先登录到系统中，才可以执行各种命令或进行 Shell 脚本编程，因此我们的第一个需求就是登录 Unix/Linux 系统，该如何做呢？

**1）解决方案**

有两种方式可以登录该系统：一种是从字符终端登录系统；另一种是通过图形用户界面的终端模拟程序登录到系统的命令行。

执行如下操作。

（1）从字符终端登录系统：

#在 Linux 系统启动以后，按下组合键 Ctrl + Alt + F1 显示字符终端，如图 1-1 所示。

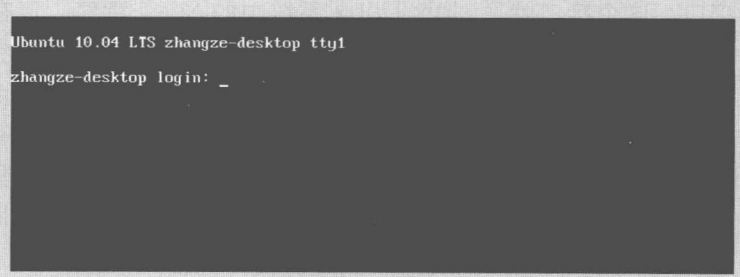


图 1-1　字符终端

#上面的信息表明系统正在等待用户登录，如图 1-2 所示为输入用户名和密码。

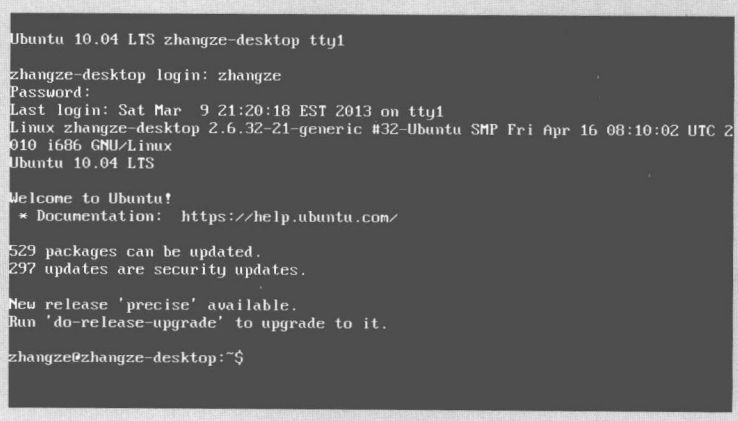


图 1-2　输入用户名和密码

#shell 打印出欢迎信息，并显示命令提示符$，等待用户输入命令。

（2）从终端模拟程序登录系统：

#按下组合键 Ctrl+Alt+F7，切换到图形终端，如图 1-3 所示。

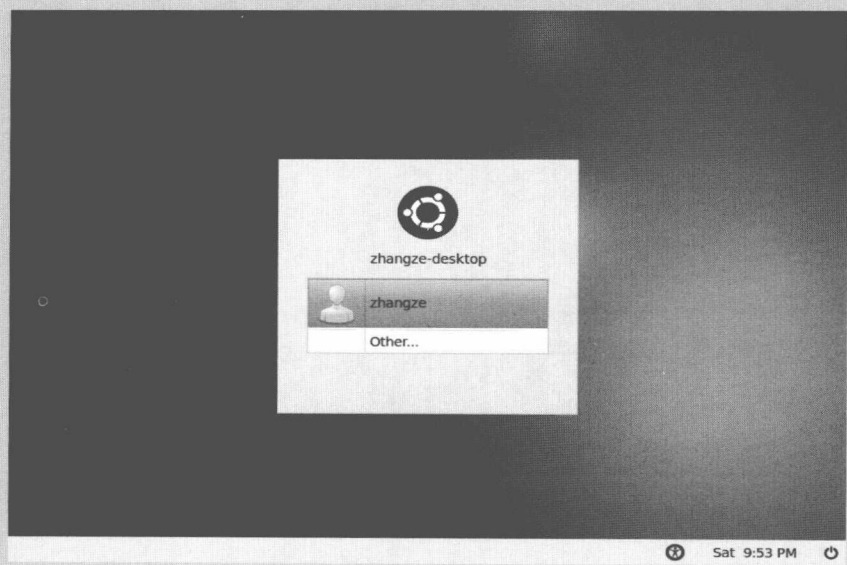


图 1-3　图形终端

#如图 1-4 所示，输入用户名和密码。

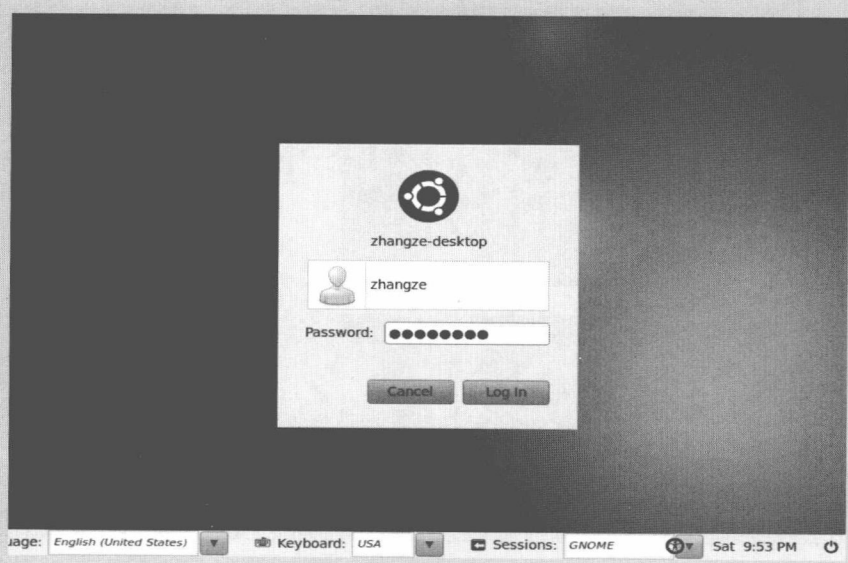


图 1-4　输入用户名和密码

#如图 1-5 所示，登录到系统的图形环境。

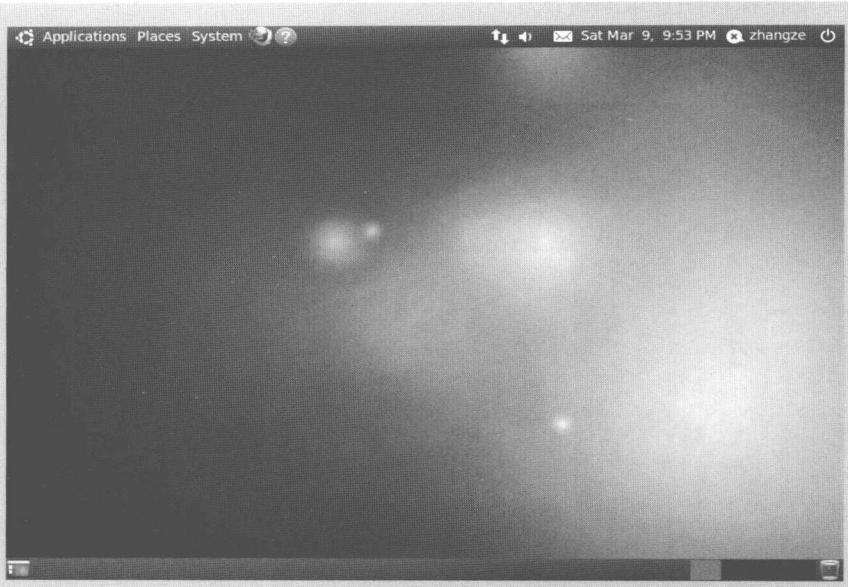


图 1-5　登录到图形环境的效果

#如图 1-6 所示，在图形用户界面的菜单栏中，通过“Applications→Accessories→Terminal”找到终端模拟程序的菜单项。

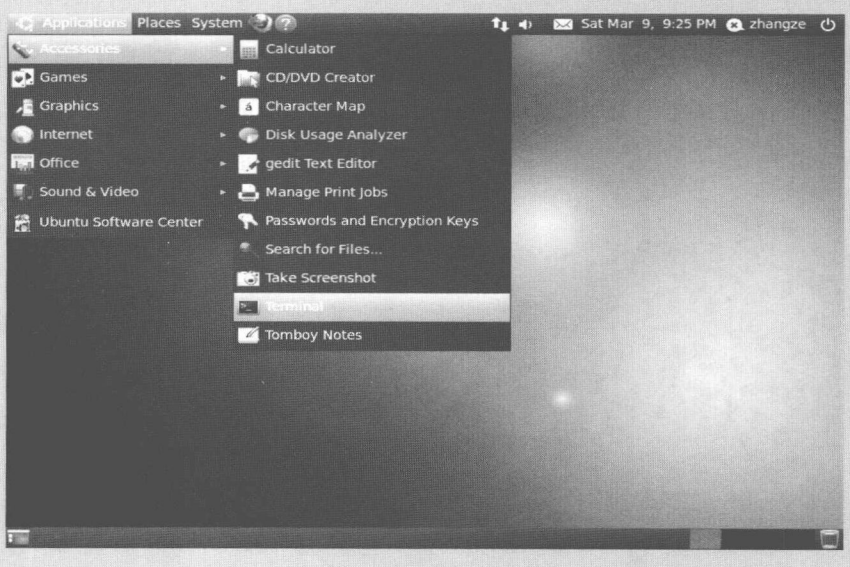


图 1-6　终端模拟程序的菜单项

#单击此菜单项即可打开如图 1-7 所示的终端模拟程序，在终端模拟窗口中同样会有 Shell 提示符，用来等待用户输入命令。


图 1-7　终端模拟程序

**2）讨论**

Unix/Linux 系统是一个多用户的操作系统，即同一时刻可以有多个用户同时登录到系统中使用系统资源，因此在使用系统以前，用户需要先登录到系统中，这个过程叫做 login。

在上面的例子中，首先按下组合键 Ctrl+Alt+F1，其作用是切换到系统的第一个字符终端 tty1，如图 1-1 所示（因为 Ubuntu Linux 默认会进入图形用户界面）。所谓的终端其实是 Unix/Linux 系统中的一个概念，读者可以简单地把显示器和键盘的组合理解为终端，一个负责输入命令，另一个负责输出结果。而按下组合键 Ctrl+Alt+F1 所得到的是字符终端，也就是说，此时屏幕上没有绘制任何像窗口那样的图形，用户与计算机是通过一个个字符来沟通的。随着 Unix 系统的不断发展，通过图形显示服务器 X Server 实现了图形用户界面以后，除了字符终端以外还有图形终端。从图 1-1 中可以看到，字符终端显示出当前我们使用的操作系统是 Ubuntu 10.04 LTS，计算机名为 zhangze-desktop，而所在的终端为 tty1。其实 Linux 系统中默认有六个字符终端和一个图形终端，分别命名为 tty1，tty2，tty3，一直到 tty7。我们通过组合键 Ctrl+Alt+F1 切换到的只是其中的第一个字符终端 tty1。此外，读者还可以使用组合键 Ctrl+Alt+F2（F3～F6）切换到其余的六个字符终端，通过组合键 Ctrl+Alt+F7 切换回图形终端。在图 1-1 中，字符终端在显示完当前主机的信息以后，显示出 login:提示符，等待用户输入登录的用户名。输入用户名以后，会继续显示出 Passwd:的提示符，等待用户输入相应的密码。如果输入的密码正确，就会显示出命令行提示符 zhangze@zhangze-desktop:~$，这就说明已经成功地登录到 Linux 系统，否则会显示出 Login incorrect 的信息并让用户重新输入用户名和密码。当输入用户名和密码时，Linux 系统会检查保存用户信息的配置文件/etc/passwd 和/etc/shadow 来对输入的用户名和密码进行校验，只有正确地匹配了这两个文件中的信息，才能成功地登录到 Linux 系统。

下面来看第二种方式，也就是如何使用终端模拟程序登录系统。首先按下组合键

Ctrl+Alt+F7 切换到图形终端，如图 1-3 所示，使用用户名和密码登录到系统中，如图 1-4 和图 1-5 所示，然后在菜单栏 Applications 中依次选择 Accessories 菜单项和 Terminal 菜单项，打开终端模拟程序，如图 1-6 所示，此时可以看到图 1-7 中的模拟终端窗口，同时可以看到命令行提示符 zhangze@zhangze-desktop:~$，登录成功。有关命令行提示符，在后面的章节中会做详细介绍。

成功登录到 Linux 系统中以后，就可以在命令行提示符 zhangze@zhangze-desktop:~$的后面输入要执行的命令，也就可以开始我们的 Shell 之旅了。

## 2. 需求 2

成功登录到命令行以后，我们就可以开始使用 Shell 了，那么如何查看当前所使用的 Shell 呢？

**1）解决方案**

有两种方式：第一种是通过环境变量 SHELL 来查看当前使用的 Shell；另一种是查看配置文件/etc/passwd 中为用户所设置的登录 Shell。

在命令行执行如下命令：

```
#当前使用的 Shell 保存在环境变量 SHELL 中
zhangze@zhangze-desktop:~$ echo $SHELL

#查看文件/etc/passwd 可以得到当前使用过的 Shell
zhangze@zhangze-desktop:~$ grep "$USER" /etc/passwd | awk -F: '{print $7}'
```

**2）运行结果**

运行结果：

```
#当前使用的 Shell 保存在环境变量 SHELL 中
zhangze@zhangze-desktop:~$ echo $SHELL
/bin/bash

#查看文件/etc/passwd 可以得到当前使用过的 Shell
zhangze@zhangze-desktop:~$ grep "$USER" /etc/passwd | awk -F: '{print $7}'
/bin/bash
```

**3）讨论**

成功登录到系统中以后，可以看到命令提示符 zhangze@zhangze-desktop:~$的后面有一个光标在不停地闪烁，意思是等待用户的输入，此时如果输入命令 ls 或 pwd，并按下 Enter 键就可以运行这些命令。在命令行输入的命令是如何在操作系统中运行起来的呢？我们知道，所谓的命令本质上也是一个可执行程序，要运行它需要在操作系统内核中启动一个进程，而在 Unix/Linux 系统中创建一个进程的方式是首先调用 fork()系统调用[①]创建子进程，然后在子进程中调用系统调用 execve()执行一个可执行程序。由此可见，需要有一个父进程

注：① fork()后面的“系统调用”是一个整体，专业名词，system call，表示操作系统内核提供的最基础的功能函数。

调用 fork()系统调用，才能创建子进程运行命令，这个父进程中所运行的程序就是本书讲述的对象 Shell。

当我们登录到 Linux 系统中以后，并没有直接和操作系统内核对话，而是和一个叫做 Shell 的程序进行交互。所谓的内核其实是操作系统的核心，它负责管理整个计算机的硬件和软件资源，通常正在运行的程序会通过内核提供的系统调用接口来和内核通信，而用户是没有办法直接和内核交互的，因此需要一个软件来充当内核与用户之间的接口，它就是 Shell 程序。用户在登录到系统中以后只和 Shell 程序进行交互，通过给 Shell 程序发送命令来告知系统该做什么事情，而 Shell 程序要么解释这些输入的命令，要么让操作系统内核运行这些命令，因此 Shell 程序的作用是把操作系统内核和用户分隔开来，就像它的名字 Shell 所代表的含义一样，是操作系统内核外面的一个壳，如图 1-8 所示。通过以上的分析可知，只要我们成功地登录到 Unix/Linux 系统，Shell 就已经开始为我们工作了。

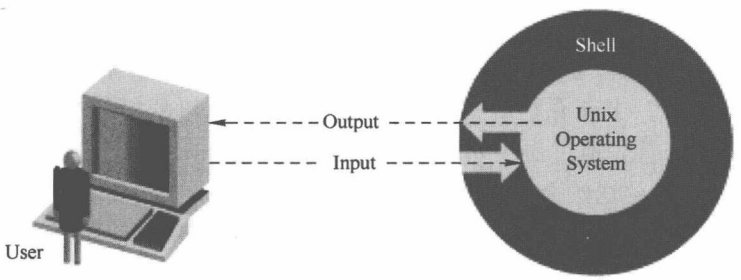


图 1-8　Shell 程序的交互

Shell 是 Unix/Linux 系统的一个软件，它读取用户输入的命令，然后让操作系统运行这些命令，运行结束以后，Shell 会把运行的结果返回给用户。由于 Shell 是 Unix/Linux 操作系统中的一个独立程序，所以程序员们为系统开发了很多 Shell 程序，从最早期的 Bourne Shell 到第一个被广泛使用的 Shell——C Shell，再到其他一些流行的 Shell，包括 Korn Shell，TC Shell，Z Shell 和 Bash（Bourne Again Shell）等。它们在作用上大体类似，都是充当用户和系统的接口，只是在一些语法格式和某些功能上会有差别，因此如果读者会使用其中的一种 Shell，只要通过学习一下它们之间的区别就可以很快地适应其他的 Shell。本书所描述的对象是目前最为流行的 bash，目前几乎所有主要的 Linux 发行版和苹果公司的 Mac OS X 操作系统都使用 bash 作为它们的默认 Shell。bash 诞生于 1988 年的 1 月 10 日，最初开发它的目的就是把它作为 GNU 系统的标准 Shell。和其他的 GNU 项目一样，bash 是自由软件，它的源代码可以从其项目的网站 http://www.gnu.org/software/bash/上免费获得。

之所以会选择 bash 作为我们学习 Shell 的对象，是因为它包含了很多优秀的特性，如它的命令行支持编辑模式，即用户可以很自由地移动光标从而修改拼写错误；此外，bash 还支持自定义函数，包括更多的流程控制结构，可以进行整数运算，执行 I/O 重定向等，因此 bash 相对于其他 Shell 拥有更强的编程能力。bash 作为目前最为流行的 Shell，还体现在它拥有很好的兼容性。它不仅能够兼容其他 Shell 的大部分语法，还从其他 Shell 中吸取了很多优秀的特性，如 bash 能够很好地兼容 Bourne Shell，而且还从 C Shell 中借鉴了作业控制的功能，使得用户可以随意暂停、恢复和终止正在执行的程序。