国外经典教材·计算机科学与技术

PEARSON Prentice Hall

Understanding Unix/Linux Programming

A Guide to Theory and Practice

Unix/Linux编程实践教程

(美) Bruce Molay 著

杨宗源 译

黄海涛

Pearson Education

清华大学出版社

国外经典教材·计算机科学与技术

Unix/Linux 编程实践教程

（美）Bruce Molay 著
杨宗源 黄海涛 译

清华大学出版社
北 京

内容简介

操作系统是计算机最重要的系统软件。Unix操作系统历经了几十年，至今仍是主流的操作系统。本书通过解释Unix的工作原理，循序渐进地讲解实现Unix中系统命令的方法，让读者理解并逐步精通Unix系统编程，进而具有编制Unix应用程序的能力。书中采用启发式、举一反三、图示讲解等多种方法讲授，语言生动、结构合理、易于理解。每一章后均附有大量的习题和编程练习，以供参考。

本书适合作为高等院校计算机及相关专业的教材和教学参考书，亦可作为有一定系统编程基础的开发人员的自学教材和参考手册。

EISBN: 0-13-008396-8

北京市版权局著作权合同登记号　图字：01－2003－1785

本书封面贴有Pearson Education(培生教育出版集团)激光防伪标签，无标签者不得销售。

图书在版编目(CIP)数据

Unix/Linux编程实践教程/(美)莫雷(Molay,B)著；杨宗源，黄海涛译. —北京：清华大学出版社，2004.10
(国外经典教材·计算机科学与技术)
书名原文：Understanding Unix/Linux Programming
ISBN 7-302-09613-9

Ⅰ. U…　Ⅱ. ①莫…②杨…③黄…　Ⅲ. ①Unix操作系统－程序设计－高等学校－教材 ②Linux操作系统－程序设计－高等学校－教材　Ⅳ. TP316.81

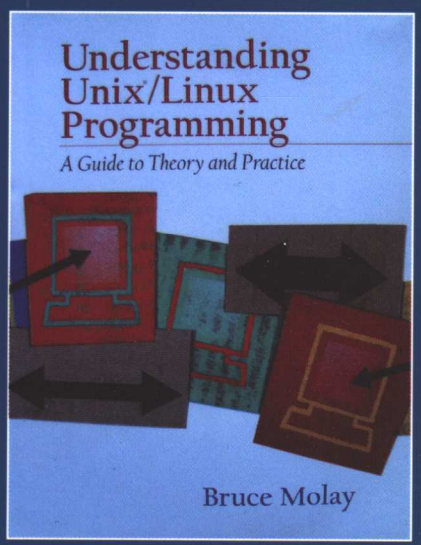
中国版本图书馆CIP数据核字(2004)第097095号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦
http://www.tup.com.cn　　**邮　　编**：100084
社 总 机：010-62770175　　**客户服务**：010-62776969
组稿编辑：许存权
文稿编辑：鲁秀敏
封面设计：秦　铭
版式设计：郑轶文
印 装 者：北京鑫霸印务有限公司
发 行 者：新华书店总店北京发行所
开　　本：185×260　**印张**：32.25　**字数**：740千字
版　　次：2004年10月第1版　2004年10月第1次印刷
书　　号：ISBN 7-302-09613-9/TP·6668
印　　数：1～5000
定　　价：56.00元(附光盘1张)

(美) Bruce Molay 著

作者简介

Bruce Molay，哈佛大学著名教授，从事Unix系统编程和教学十余年之久，本书就是在哈佛继续教育学院的Unix system Programming 课程的基础上，结合作者的实践、教学经验编写而成。。



杨宗源
黄海涛 译

译者简介

杨宗源，华东师范大学计算机科学系教授、系主任， 研究领域：软件工程、工具及环境：形式化、面向对象、组件、中间件、分布计算、过程管理、测试与度量、语言处理等。

译 者 序

操作系统是计算机最重要的系统软件，是计算机应用的基础。Unix 系统是迄今最优秀的操作系统，虽历经几十年，有许多变化，但基本的体系结构保持稳定。更难能可贵的是，在计算机发展如此迅速的今天，Unix 系统仍以其安全、稳定及强大的处理能力，仍为最主要的操作系统，计算机技术发展到今天，很多关键应用还依赖于 Unix 系统。

本书从解释 Unix 的工作原理、讲解系统命令的功能入手，由浅入深，抽丝剥茧般地将 Unix 系统的实现机理逐渐展示给读者，同时针对不同的实现方法展开了深入的讨论。书中用大量的篇幅剖析了多个 Unix 系统命令的实现方法，循序渐进地让读者理解并逐步精通 Unix 系统编程，进而具有编制 Unix 应用程序的能力。书中采用启发式、举一反三、类比分析、图示讲解等多种方法讲授，语言生动、结构合理、易于理解。每一章后均附有大量的习题和编程练习，读者可以参考练习。本书十分适合初学者阅读，各章总是先给出一个最简单的例子，然后不断地加入新的特性，最终达到实用的程度。通过这种方法，使读者对系统的理解逐步深入。但这并不影响本书的深度，书中涉及了很多 Unix 的高级特性，并进行了深入浅出、入木三分的分析，相信资深的开发人员也能够从中获益。

本书适合作为高等院校计算机及相关专业的教材和教学参考书，亦可作为有一定系统编程基础的开发人员的自学教材和参考手册。

在本书的翻译过程中，除杨宗源、黄海涛外，嵇海明、朱羚、徐国庆、莫杰众、李晶、陈玲、许峰兵、查冰等也参与了部分翻译和校对工作。限于译者的水平，译文中定有错误和不妥之处，恳请读者指正。

译 者

2004 年 2 月于上海

前　言

理解 Unix 编程

关于 Unix

写作本书的目的是解释 Unix 的工作原理以及如何编写 Unix 系统程序。Unix 诞生 30 年来，至今仍在进行着不断的改进，并且变得越来越复杂。但它并未因此而难以理解，最初的基本结构和设计原则仍然适用。通过理解它的结构、原理和历史，读者可以阅读、加强和增添不断积累起来的巨大的 Unix 程序库。同时，在这个过程中，相信读者也可以感受到许多乐趣。

为了使讲解更加清晰明了，书中采用了图片、类推、伪代码、源代码、实验、练习和特性点等多种形式。而且这些讲解内容都是从实际的问题和项目中提炼出来的。

本书的适用对象

阅读本书的读者要有一定的 C 语言基础。如果已经学过 C++，理解书中的代码将会更加容易，并会很快适应本书。读者应该了解数组、结构、指针和链表的概念，并具有用它们阅读和编写程序的能力。

但这里并不要求读者用过 Unix，也不要求读者了解操作系统的内核原理。在每一章的开头都首先讲解 Unix 的用户级特性。通过“该命令有什么功能?”这个问题很自然地将读者引向了另外一个系统级的问题“该功能是如何实现的?”。

学习过程中，需要读者登录 Unix 系统并亲自做一些实验。

可以学到什么

书中介绍了 Unix 系统的组成部分，并讲解了它们的功能、工作原理及如何使用它们进行编程。在这个过程中，读者还可以领悟到这些组件是怎样组合成这个统一、智能的操作系统的。

本书源于我从 1990 年开始在哈佛大学职业教育学院(Harvard Extension School)执教的一门课程——Unix 系统编程。在课程评估和毕业几年后学生给我发来的邮件中，学生们向我描述了他们在这门课中学到的东西，一个学生说这门课给了他“通往国王宝座的钥匙”。无论用户级、系统级还是理论级，他对 Unix 都有了很好的理解，他觉得他已经可以应对各个方面的情况，并可以解决所碰到的大多数问题。还有一个学习过这门课程的内科医生，他说他很喜欢这种实例教学法，并将其比作见习医生在医院里通过实际病例来学习。

还有一个毕业后在开放软件公司担任项目主管的学生说，这门课使他掌握了他在工作中所需的知识和技能。

适用的 Unix 版本

本书适用于包括 GNU 和 Linux 在内的几乎所有的 Unix 版本。书中重点讲述构成所有 Unix 版本基础的结构和技能，而不是随各个版本变化的具体细节。只要掌握了这些基本知识，那些细节的学习将会很容易上手。

答谢

这本书的写成得益于许多朋友的帮助。

感谢 Petra Recter 为我提供了这样一个机会并在这项课题中给了我诸多指导，也要感谢 Gregory Dulles，他和我共同完成了实例方面的工作。

我要感谢本书初稿的审阅者们，他们给出了许多关切、鼓励和具体建议，他们是 Ben Abbott、John B. Connely、Geoff Sutcliffe、Louis Taber、Sam R. Thangiah 和 Lawrence B. Wells。也要感谢提供了图片软件重要信息的 Peggy Bustamante 和 Amit Chatterjee，以及在整个项目过程中无数次与我沟通给我精神和实际支持的 Yuriko Kuwabara。

我感谢在 Unix 编程课程中学习的许多学生和助教们，他们在讨论课中的问题、观点以及与我在个别指导过程中的谈话，对构成本书的整体框架、讲解、比喻和图片等方面都起到了很大的作用。尤其要感谢的是多年来一直作为助教的 Larry deLuca 以及他为第 13 章提供的材料。

目　录

第 1 章　Unix 系统编程概述 …… **1**
1.1　介绍 …… 1
1.2　什么是系统编程 …… 1
1.2.1　简单的程序模型 …… 1
1.2.2　系统模型 …… 2
1.2.3　操作系统的职责 …… 3
1.2.4　为程序提供服务 …… 4
1.3　理解系统编程 …… 4
1.3.1　系统资源 …… 4
1.3.2　目标：理解系统编程 …… 5
1.3.3　方法：通过三个问题来理解 …… 5
1.4　从用户的角度来理解 Unix …… 6
1.4.1　Unix 能做些什么 …… 6
1.4.2　登录－运行程序－注销 …… 6
1.4.3　目录操作 …… 8
1.4.4　文件操作 …… 10
1.5　从系统的角度来看 Unix …… 12
1.5.1　用户和程序之间的连接方式 …… 12
1.5.2　网络桥牌 …… 12
1.5.3　bc：Unix 的计算器 …… 13
1.5.4　从 bc/dc 到 Web …… 16
1.6　动手实践 …… 16
1.7　工作步骤与概要图 …… 23
1.7.1　接下来的工作步骤 …… 23
1.7.2　Unix 的概要图 …… 23
1.7.3　Unix 的发展历程 …… 23
小结 …… 24

第 2 章　用户、文件操作与联机帮助：编写 who 命令 …… **25**
2.1　介绍 …… 25
2.2　关于命令 who …… 26
2.3　问题 1：who 命令能做些什么 …… 27
2.4　问题 2：who 命令是如何工作的 …… 28

2.5 问题 3：如何编写 who …… 32
2.5.1 问题：如何从文件中读取数据结构 …… 33
2.5.2 答案：使用 open、read 和 close …… 34
2.5.3 编写 who1.c …… 36
2.5.4 显示登录信息 …… 37
2.5.5 编写 who2.c …… 39
2.5.6 回顾与展望 …… 44
2.6 编写 cp(读和写) …… 44
2.6.1 问题 1：cp 命令能做些什么 …… 44
2.6.2 问题 2：cp 命令是如何创建/重写文件的 …… 44
2.6.3 问题 3：如何编写 cp …… 45
2.6.4 Unix 编程看起来好像很简单 …… 47
2.7 提高文件 I/O 效率的方法：使用缓冲 …… 48
2.7.1 缓冲区的大小对性能的影响 …… 48
2.7.2 为什么系统调用需要很多时间 …… 48
2.7.3 低效率的 who2.c …… 49
2.7.4 在 who2.c 中运用缓冲技术 …… 50
2.8 内核缓冲技术 …… 53
2.9 文件读写 …… 54
2.9.1 注销过程：做了些什么 …… 54
2.9.2 注销过程：如何工作的 …… 54
2.9.3 改变文件的当前位置 …… 55
2.9.4 编写终端注销的代码 …… 57
2.10 处理系统调用中的错误 …… 58
小结 …… 59

第 3 章 目录与文件属性：编写 ls …… 63
3.1 介绍 …… 63
3.2 问题 1：ls 命令能做什么 …… 63
3.2.1 ls 可以列出文件名和文件的属性 …… 63
3.2.2 列出指定目录或文件的信息 …… 64
3.2.3 经常用到的命令行选项 …… 65
3.2.4 问题 1 的答案 …… 65
3.3 文件树 …… 65
3.4 问题 2：ls 是如何工作的 …… 66
3.4.1 什么是目录 …… 66
3.4.2 是否可以用 open、read 和 close 来操作目录 …… 66
3.4.3 如何读目录的内容 …… 67

3.5 问题3：如何编写ls …… 69
3.6 编写ls -l …… 71
3.6.1 问题1：ls -l能做些什么 …… 71
3.6.2 问题2：ls -l是如何工作的 …… 72
3.6.3 用stat得到文件信息 …… 72
3.6.4 stat提供的其他信息 …… 74
3.6.5 如何实现 …… 75
3.6.6 将模式字段转换成字符 …… 75
3.6.7 将用户/组ID转换成字符串 …… 79
3.6.8 编写ls2.c …… 81
3.7 三个特殊的位 …… 86
3.7.1 set-user-ID位 …… 86
3.7.2 set-group-ID位 …… 87
3.7.3 sticky位 …… 87
3.7.4 用ls -l看到的特殊属性 …… 87
3.8 ls小结 …… 88
3.9 设置和修改文件的属性 …… 88
3.9.1 文件类型 …… 88
3.9.2 许可位与特殊属性位 …… 89
3.9.3 文件的链接数 …… 90
3.9.4 文件所有者与组 …… 90
3.9.5 文件大小 …… 91
3.9.6 时间 …… 91
3.9.7 文件名 …… 91
小结 …… 92

第4章 文件系统：编写pwd …… 96
4.1 介绍 …… 96
4.2 从用户的角度看文件系统 …… 97
4.2.1 目录和文件 …… 97
4.2.2 目录命令 …… 97
4.2.3 文件操作命令 …… 97
4.2.4 针对目录树的命令 …… 99
4.2.5 目录树的深度几乎没有限制 …… 99
4.2.6 Unix文件系统小结 …… 100
4.3 Unix文件系统的内部结构 …… 100
4.3.1 第一层抽象：从磁盘到分区 …… 100
4.3.2 第二层抽象：从磁盘到块序列 …… 100

4.3.3 第三层抽象：从块序列到三个区域的划分 …… 100
4.3.4 文件系统的实现：创建一个文件的过程 …… 101
4.3.5 文件系统的实现：目录的工作过程 …… 102
4.3.6 文件系统的实现：cat 命令的工作原理 …… 104
4.3.7 i-节点和大文件 …… 105
4.3.8 Unix 文件系统的改进 …… 106
4.4 理解目录 …… 107
4.4.1 理解目录结构 …… 107
4.4.2 与目录树相关的命令和系统调用 …… 109
4.5 编写 pwd …… 113
4.5.1 pwd 的工作过程 …… 113
4.5.2 pwd 的一种版本 …… 114
4.6 多个文件系统的组合：由多棵树构成的树 …… 116
4.6.1 装载点 …… 117
4.6.2 多重 i-节点号和设备交叉链接 …… 118
4.6.3 符号链接 …… 119
小结 …… 120

第 5 章 连接控制：学习 stty …… 124
5.1 为设备编程 …… 124
5.2 设备就像文件 …… 124
5.2.1 设备具有文件名 …… 125
5.2.2 设备和系统调用 …… 125
5.2.3 例子：终端就像文件 …… 126
5.2.4 设备文件的属性 …… 126
5.2.5 编写 write 程序 …… 127
5.2.6 设备文件和 i-节点 …… 128
5.3 设备与文件的不同之处 …… 129
5.4 磁盘连接的属性 …… 130
5.4.1 属性 1：缓冲 …… 130
5.4.2 属性 2：自动添加模式 …… 132
5.4.3 用 open 控制文件描述符 …… 133
5.4.4 磁盘连接小结 …… 134
5.5 终端连接的属性 …… 135
5.5.1 终端的 I/O 并不如此简单 …… 135
5.5.2 终端驱动程序 …… 137
5.5.3 stty 命令 …… 137
5.5.4 编写终端驱动程序：关于设置 …… 138

5.5.5　编写终端驱动程序：关于函数 …… 139
5.5.6　编写终端驱动程序：关于位 …… 140
5.5.7　编写终端驱动程序：几个程序例子 …… 142
5.5.8　终端连接小结 …… 146
5.6　其他设备编程：ioctl …… 146
5.7　文件、设备和流 …… 147
小结 …… 147

第6章　为用户编程：终端控制和信号 …… 153
6.1　软件工具与针对特定设备编写的程序 …… 153
6.2　终端驱动程序的模式 …… 154
6.2.1　规范模式：缓冲和编辑 …… 155
6.2.2　非规范处理 …… 156
6.2.3　终端模式小结 …… 157
6.3　编写一个用户程序：play_again.c …… 158
6.4　信号 …… 168
6.4.1　Ctrl-C 做什么 …… 168
6.4.2　信号是什么 …… 168
6.4.3　进程该如何处理信号 …… 170
6.4.4　信号处理的例子 …… 171
6.5　为处理信号做准备：play_again4.c …… 173
6.6　进程终止 …… 176
6.7　为设备编程 …… 176
小结 …… 176

第7章　事件驱动编程：编写一个视频游戏 …… 180
7.1　视频游戏和操作系统 …… 180
7.2　任务：单人弹球游戏(Pong) …… 182
7.3　屏幕编程：curses 库 …… 182
7.3.1　介绍 curses …… 182
7.3.2　curses 内部：虚拟和实际屏幕 …… 185
7.4　时间编程：sleep …… 185
7.5　时钟编程 1：Alarms …… 188
7.5.1　添加时延：sleep …… 189
7.5.2　sleep()是如何工作的：使用 Unix 中的 Alarms …… 189
7.5.3　调度将要发生的动作 …… 191
7.6　时间编程 2：间隔计时器 …… 191
7.6.1　添加精度更高的时延：usleep …… 192
7.6.2　三种计时器：真实、进程和实用 …… 192

7.6.3 两种间隔：初始和重复 …… 192
7.6.4 用间隔计时器编程 …… 193
7.6.5 计算机有几个时钟 …… 196
7.6.6 计时器小结 …… 197
7.7 信号处理 1：使用 signal …… 198
7.7.1 早期的信号处理机制 …… 198
7.7.2 处理多个信号 …… 198
7.7.3 测试多个信号 …… 200
7.7.4 信号机制其他的弱点 …… 202
7.8 信号处理 2：sigaction …… 203
7.8.1 处理多个信号：sigaction …… 203
7.8.2 信号小结 …… 206
7.9 防止数据损毁(Data Corruption) …… 206
7.9.1 数据损毁的例子 …… 206
7.9.2 临界区(Critical Sections) …… 206
7.9.3 阻塞信号：sigprocmask 和 sigsetops …… 207
7.9.4 重入代码(Reentrant Code)：递归调用的危险 …… 208
7.9.5 视频游戏中的临界区 …… 208
7.10 kill：从另一个进程发送的信号 …… 209
7.11 使用计时器和信号：视频游戏 …… 210
7.11.1 bounce1d.c：在一条线上控制动画 …… 210
7.11.2 bounce2d.c：两维动画 …… 213
7.11.3 完成游戏 …… 218
7.12 输入信号：异步 I/O …… 218
7.12.1 使用异步 I/O …… 218
7.12.2 方法 1：使用 O_ASYNC …… 218
7.12.3 方法 2：使用 aio_read …… 221
7.12.4 弹球程序中需要异步读入吗 …… 224
7.12.5 异步输入、视频游戏和操作系统 …… 224
小结 …… 224

第 8 章 进程和程序：编写命令解释器 sh …… 228
8.1 进程＝运行中的程序 …… 228
8.2 通过命令 ps 学习进程 …… 229
8.2.1 系统进程 …… 231
8.2.2 进程管理和文件管理 …… 232
8.2.3 内存和程序 …… 232
8.3 shell：进程控制和程序控制的一个工具 …… 233

8.4 shell 是如何运行程序的 …… 234
8.4.1 shell 的主循环 …… 234
8.4.2 问题 1：一个程序如何运行另一个程序 …… 235
8.4.3 问题 2：如何建立新的进程 …… 240
8.4.4 问题 3：父进程如何等待子进程的退出 …… 244
8.4.5 小结：shell 如何运行程序 …… 249
8.5 实现一个 shell：psh2.c …… 250
8.6 思考：用进程编程 …… 254
8.7 exit 和 exec 的其他细节 …… 255
8.7.1 进程死亡：exit 和_exit …… 255
8.7.2 exec 家族 …… 256
小结 …… 257

第 9 章 可编程的 shell、shell 变量和环境：编写自己的 shell …… 260
9.1 shell 编程 …… 260
9.2 什么是以及为什么要使用 shell 脚本语言 …… 260
9.3 smsh1——命令行解析 …… 263
9.4 shell 中的流程控制 …… 270
9.4.1 if 语句做些什么 …… 270
9.4.2 if 是如何工作的 …… 271
9.4.3 在 smsh 中增加 if …… 272
9.4.4 smsh2.c：修改后的代码 …… 273
9.5 shell 变量：局部和全局 …… 278
9.5.1 使用 shell 变量 …… 279
9.5.2 变量的存储 …… 280
9.5.3 增加变量命令：Built-ins …… 280
9.5.4 效果如何 …… 283
9.6 环境：个性化设置 …… 284
9.6.1 使用环境 …… 285
9.6.2 什么是环境以及它是如何工作的 …… 286
9.6.3 在 smsh 中增加环境处理 …… 288
9.6.4 varlib.c 的代码 …… 290
9.7 已实现的 shell 的功能 …… 295
小结 …… 296

第 10 章 I/O 重定向和管道 …… 299
10.1 shell 编程 …… 299
10.2 一个 shell 应用程序：监视系统用户 …… 300
10.3 标准 I/O 与重定向的若干概念 …… 301

10.3.1 概念 1：3 个标准文件描述符 …… 302
10.3.2 默认的连接：tty …… 302
10.3.3 程序都输出到 stdout …… 303
10.3.4 重定向 I/O 的是 shell 而不是程序 …… 303
10.3.5 理解 I/O 重定向 …… 304
10.3.6 概念 2："最低可用文件描述符(Lowest-Available-fd)"原则 …… 304
10.3.7 两个概念的结合 …… 305
10.4 如何将 stdin 定向到文件 …… 305
10.4.1 方法 1：close then open …… 305
10.4.2 方法 2：open..close..dup..close …… 308
10.4.3 系统调用 dup 小结 …… 310
10.4.4 方法 3：open..dup2..close …… 310
10.4.5 shell 为其他程序重定向 stdin …… 310
10.5 为其他程序重定向 I/O：who> userlist …… 310
10.6 管道编程 …… 314
10.6.1 创建管道 …… 314
10.6.2 使用 fork 来共享管道 …… 317
10.6.3 使用 pipe、fork 以及 exec …… 318
10.6.4 技术细节：管道并非文件 …… 320
小结 …… 321

第 11 章 连接到近端或远端的进程：服务器与 Socket(套接字) …… 325
11.1 产品和服务 …… 325
11.2 一个简单的比喻：饮料机接口 …… 326
11.3 bc：Unix 中使用的计算器 …… 327
11.3.1 编写 bc：pipe、fork、dup、exec …… 328
11.3.2 对协同进程的讨论 …… 332
11.3.3 fdopen：让文件描述符像文件一样使用 …… 332
11.4 popen：让进程看似文件 …… 332
11.4.1 popen 的功能 …… 332
11.4.2 实现 popen：使用 fdopen 命令 …… 334
11.4.3 访问数据：文件、应用程序接口(API)和服务器 …… 336
11.5 socket：与远端进程相连 …… 337
11.5.1 类比："电话中传来声音：现在时间是……" …… 338
11.5.2 因特网时间、DAP 和天气服务器 …… 340
11.5.3 服务列表：众所周知的端口 …… 341
11.5.4 编写 timeserv.c：时间服务器 …… 342

11.5.5 测试 timeserv.c …… 347
11.5.6 编写 timeclnt.c：时间服务客户端 …… 347
11.5.7 测试 timeclnt.c …… 350
11.5.8 另一种服务器：远程的 ls …… 351
11.6 软件精灵 …… 356
小结 …… 356

第 12 章 连接和协议：编写 Web 服务器 …… 360
12.1 服务器设计重点 …… 360
12.2 三个主要操作 …… 360
12.3 操作 1 和操作 2：建立连接 …… 361
12.3.1 操作 1：建立服务器端 socket …… 361
12.3.2 操作 2：建立到服务器的连接 …… 362
12.3.3 socklib.c …… 362
12.4 操作 3：客户/服务器的会话 …… 364
12.4.1 使用 socklib.c 的 timeserv/timeclnt …… 365
12.4.2 第 2 版的服务器：使用 fork …… 366
12.4.3 服务器的设计问题：DIY 或代理 …… 367
12.5 编写 Web 服务器 …… 369
12.5.1 Web 服务器功能 …… 369
12.5.2 设计 Web 服务器 …… 370
12.5.3 Web 服务器协议 …… 370
12.5.4 编写 Web 服务器 …… 372
12.5.5 运行 Web 服务器 …… 374
12.5.6 Webserv 的源程序 …… 374
12.5.7 比较 Web 服务器 …… 379
小结 …… 379

第 13 章 基于数据报(Datagram)的编程：编写许可证服务器[①] …… 381
13.1 软件控制 …… 381
13.2 许可证控制简史 …… 382
13.3 一个非计算机系统实例：轿车管理系统 …… 383
13.3.1 轿车钥匙管理描述 …… 383
13.3.2 用客户/服务器方式管理轿车 …… 383
13.4 许可证管理 …… 384
13.4.1 许可证服务系统：它做些什么 …… 384
13.4.2 许可证服务系统：如何工作 …… 385
13.4.3 一个通信系统的例子 …… 386
13.5 数据报 socket …… 386