

SAMS

GNU/Linux 编程指南

(第二版)

入门·应用·精通

[美] Kurt Wall 等著

张 辉 译

清华大学出版社

(京)新登字 158 号

著作权合同登记号: 01-2001-2327

内 容 提 要

本书全面而深入地介绍了 GNU/Linux 编程。首先介绍了在 Linux 上编程必备的编程工具, 然后在库函数、系统调用以及内核上阐述 Linux 编程知识, 并专门讲述了包括 TCP/IP、UDP 以及多播套接口在内的网络编程知识; 图形界面也是本书的重点内容, 本书着重讲述了文本形式的图形界面库 ncurses; 还分别讲解了真正图形化的流行系统 X Window、Qt、GNOME 以及 OpenGL 的基本编程方法; 最后, 介绍了 Bash 编程和设备驱动编程。

本书包含大量实用实例, 读者可以通过实例代码深入理解编程思想和技巧。本书另一优点是讲述了其他编程书籍通常没有提及的 RPM 包管理工具、文档编写以及发布许可证选择等内容, 这是任何准备投身于 GNU 开发工作的程序员所必须具备的知识。

本书对于所有 Linux 编程人员——无论是初学者还是高级用户——都是一本不可多得的参考资料。

Linux Programming Unleashed. Second Edition

Copyright © 2001 by Sams Publishing.

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher.

本书中文简体字版由美国 Sams 公司授权清华大学出版社和北京科海培训中心出版。未经出版者书面允许不得以任何方式复制或抄袭本书内容。

版权所有, 盗版必究。

本书封面贴有清华大学出版社激光防伪标签, 无标签者不得销售。

书 名: GNU/Linux 编程指南 (第二版)
作 者: Kurt Wall, et al.
译 者: 张辉
出 版 者: 清华大学出版社 (北京清华大学校内, 邮编 100084)
印 刷 者: 北京市耀华印刷有限公司
发 行 者: 新华书店总店北京科技发行所
开 本: 787×1092 1/16 印张: 42.125 字数: 1024 千字
版 次: 2002 年 6 月第 1 版 2002 年 6 月第 1 次印刷
书 号: ISBN 7-302-05550-5/TP·3273
印 数: 0001~5000
定 价: 68.00 元

引 言

很难夸大 Linux 所取得的不可思议的成功和流行。这种成功引发了对开发 Linux 应用的程序员的巨大需求，而且从长远来看，这一需求只会增加。但是，知道从哪里开始，如何开始学习为 Linux 编写程序则很困难。要对新的 Linux 程序员说的是，欢迎加入这场革命！读者将会发现本书在大多数方面是 Linux 编程的优秀指导和教材。如果读者是一位有经验的 Linux 程序员，本书也同样适用，因为它传达了一系列 Linux 编程的主题，其中也包括读者可能尚未探索过的主题。

本书的目的

本书打算向读者展示如何用 Linux 编程，如何在 Linux 上编程以及如何为 Linux 编程。它把注意力几乎全部放在了 C 语言上，因为 C 语言仍是 Linux 的“万国通用语言”。在向读者介绍一些基本开发工具之后，本书立即投入到使用 Linux I/O 模型进行编程的内容当中。本书的第 3 部分介绍了进程和同步的问题，包括线程、内存管理以及进程间通信。本书的第 5 部分专门讨论 Linux 的用户界面，它既用到了基于文本的工具，也用到了基于图形的工具（X Window 系统）。其他部分则涉及到包括 shell 编程和编写设备驱动程序在内的各种各样的话题。本书结尾的 3 章正常情况下会被一般编程的书籍所忽略：它向用户交付应用。这几章向读者展示了如何使用诸如 RPM 这样的包管理工具，以及如何创建有用的文档，还介绍了许可证的问题及其选择。读完本书，读者们就能为投身于称为“Linux”的伟大的社会性以及技术性现象做好了准备。

本书的读者

熟悉其他操作系统但是刚刚接触 Linux 的程序员会得到有关 Linux 编程的详细介绍。读者会接触到将要用到的工具以及将要开展工作的环境。

有经验的 UNIX 程序员会发现 Linux 的编程习惯用法非常熟悉。对于这类读者，本书突出了 Linux 和 UNIX 专有版本的不同之处。最大的可移植性将会是一个重要的话题，因为 Linux 运行在一种尚在变化的平台上：Intel i386, Sun Sparc 和 UltraSparc, Digital Alpha, MIPS 处理器, Power PC 以及基于 Motorola 68000 的 Macintosh 计算机。

中等水平的 C 程序员也能从本书获益。总体而言，在 Linux 编程类似于在其他类 UNIX 系统上编程，所以读者将会很快入门，成为高效的 UNIX 程序员并且理解 Linux/UNIX 深入的特质。

目 录

第 1 部分 Linux 编程工具包

第 1 章 Linux 及 Linux 编程综述.....1

1.1 Linux 变得成熟了.....1	
1.1.1 Linux 的昨天.....1	
1.1.2 Linux 的今天.....3	
1.1.3 Linux 的明天.....3	
1.2 为何选择 Linux 编程.....3	
1.3 每章内容介绍.....5	
1.3.1 Linux 编程工具包.....5	
1.3.2 输入、输出、文件和目录.....5	
1.3.3 进程和同步.....6	
1.3.4 网络编程.....6	
1.3.5 用户界面编程.....7	
1.3.6 特殊编程技术.....8	
1.3.7 补充内容.....8	
1.4 小结.....8	

第 2 章 设置开发系统.....9

2.1 一般性考虑.....9	
2.2 主板和 CPU.....10	
2.2.1 板上 I/O.....12	
2.2.2 处理器.....12	
2.2.3 BIOS.....13	
2.2.4 内存.....13	
2.2.5 机箱和电源.....13	
2.3 用户交互硬件：视频、声音、键盘 及鼠标.....14	
2.3.1 显卡.....14	
2.3.2 显示器.....15	
2.3.3 声卡.....16	
2.3.4 键盘及鼠标.....16	
2.4 通信设备、端口及总线.....17	

2.4.1 调制解调器.....17	
2.4.2 网络接口卡.....18	
2.4.3 SCSI.....18	
2.4.4 USB 和火线（IEEE1394）.....18	
2.4.5 串行卡.....18	
2.4.6 IRDA.....19	
2.4.7 PCMCIA 卡.....19	
2.4.8 ISA 即插即用设备.....19	
2.5 存储设备.....19	
2.5.1 硬盘.....20	
2.5.2 可移动磁盘设备.....20	
2.5.3 CD-ROM/DVD.....20	
2.5.4 磁带备份设备.....20	
2.6 外围设备.....21	
2.6.1 打印机.....21	
2.6.2 扫描仪.....21	
2.6.3 数码相机.....22	
2.6.4 家居自动控制设备.....22	
2.7 完备型系统.....22	
2.8 便携系统.....22	
2.9 开发工具软件.....23	
2.9.1 关键库和头文件.....23	
2.9.2 调试器.....23	
2.9.3 编程工具.....23	
2.9.4 文本编辑器.....24	
2.10 小结.....24	

第 3 章 使用 GNU CC.....25

3.1 GNU CC 特性.....25	
3.2 教学示例.....26	
3.3 常用命令行选项.....29	

3.3.1 函数库和包含文件.....	30	5.3.5 类型定义测试.....	64
3.3.2 警告和出错消息选项.....	31	5.3.6 编译器行为测试.....	65
3.4 优化选项.....	36	5.3.7 系统服务测试.....	65
3.5 调试选项.....	38	5.3.8 UNIX 变体测试.....	66
3.6 特定体系结构的选项.....	40	5.4 普通宏.....	66
3.7 GNU C 扩展.....	41	5.5 一个带注释的 autoconf 脚本.....	68
3.7.1 关于可移植性.....	41	5.6 小结.....	74
3.7.2 GNU 扩展.....	42		
3.8 gcc: 奔腾处理器的编译器.....	45	第 6 章 比较和合并源代码文件.....	75
3.9 小结.....	45	6.1 使用 diff 命令比较文件.....	75
第 4 章 使用 GNU make 管理项目.....	46	6.2 理解 diff3 命令.....	83
4.1 为何使用 make.....	46	6.3 准备源代码补丁.....	86
4.2 编写 makefile.....	46	6.3.1 patch 的命令行选项.....	86
4.3 编写 makefile 的规则.....	47	6.3.2 创建补丁.....	87
4.3.1 伪目标.....	49	6.3.3 应用补丁.....	88
4.3.2 变量.....	50	6.4 小结.....	88
4.3.3 隐式规则.....	52		
4.3.4 模式规则.....	53	第 7 章 使用 RCS 和 CVS 控制版本... ..	89
4.3.5 注释.....	53	7.1 基本术语.....	89
4.4 命令行选项和参数.....	53	7.2 使用修订控制系统 (RCS).....	90
4.5 调试 make.....	54	7.2.1 RCS 基本用法.....	90
4.6 常见的 make 出错信息.....	54	7.2.2 找出 RCS 文件间的不同.....	95
4.7 有用的 makefile 目标.....	54	7.2.3 其他 RCS 命令.....	98
4.8 小结.....	55	7.3 使用并发版本系统 (CVS).....	100
第 5 章 创建可移植的自配置软件.....	56	7.3.1 同 RCS 相比的优点.....	100
5.1 考虑可移植性.....	56	7.3.2 设置 CVS.....	100
5.1.1 什么是程序的可移植性.....	56	7.3.3 检出源代码文件.....	102
5.1.2 移植性的线索和技巧.....	57	7.3.4 将改动合并进源代码库.....	103
5.2 理解 autoconf.....	58	7.3.5 检查改动.....	103
5.2.1 创建 configure.in.....	58	7.3.6 添加和删除文件.....	104
5.2.2 构造文件.....	58	7.3.7 解决文件冲突.....	105
5.2.3 有用的 autoconf 工具.....	59	7.3.8 CVS 命令.....	106
5.3 内置宏.....	62	7.3.9 CVS 选项.....	106
5.3.1 候选程序测试.....	62	7.4 小结.....	107
5.3.2 库函数测试.....	63		
5.3.3 头文件测试.....	64	第 8 章 调试.....	108
5.3.4 结构测试.....	64	8.1 为使用 GDB 进行编译.....	108
		8.2 使用基本的 GDB 命令.....	109
		8.2.1 启动 GDB.....	109

8.2.2 在调试器中查看代码.....	111	9.4.2 系统日志函数.....	135
8.2.3 检查数据.....	111	9.4.3 用户程序.....	138
8.2.4 设置断点.....	113	9.5 小结.....	139
8.2.5 检查并更改运行中的代码.....	114		
8.3 高级 GDB 概念和命令.....	116	第 10 章 使用库.....	140
8.3.1 变量的作用域和上下文.....	116	10.1 使用编程库.....	140
8.3.2 遍历函数堆栈.....	117	10.1.1 库兼容性.....	140
8.3.3 操纵源代码文件.....	119	10.1.2 命名和编号约定.....	141
8.3.4 与 Shell 进行通信.....	119	10.1.3 经常使用的库.....	142
8.3.5 附加到某个运行中的程序.....	119	10.2 库操作工具.....	143
8.4 小结.....	121	10.2.1 理解 nm 命令.....	143
第 9 章 出错处理.....	122	10.2.2 理解 ar 命令.....	144
9.1 出错处理简述.....	122	10.2.3 理解 ldd 命令.....	144
9.2 出错处理选项.....	122	10.2.4 理解 ldconfig.....	145
9.3 C 语言机制.....	123	10.2.5 环境变量和配置文件.....	145
9.3.1 assert 宏.....	123	10.3 编写并使用静态库.....	146
9.3.2 使用预编译.....	125	10.4 编写并使用共享库.....	151
9.3.3 标准库函数.....	127	10.5 使用动态加载的共享对象.....	152
9.4 使用系统日志.....	133	10.5.1 理解 dl 接口.....	152
9.4.1 系统日志选项.....	134	10.5.2 使用 dl 接口.....	154
		10.6 小结.....	155

第 2 部分 输入、输出、文件和目录

第 11 章 输入和输出.....	156	11.3.7 使用 fchown 改变文件所有权.....	172
11.1 基本特点和概念.....	156	11.3.8 使用 fchmod 改变文件读写权.....	172
11.2 理解文件描述符.....	160	11.3.9 使用 flock 和 fcntl 给文件上锁.....	172
11.2.1 文件描述符的概念.....	160	11.3.10 使用 dup 和 dup2 调用.....	177
11.2.2 文件描述符的优缺点.....	161	11.3.11 使用 select 同时读写多个文件.....	179
11.3 使用文件描述符.....	162	11.3.12 使用 ioctl.....	182
11.3.1 打开关闭文件描述符.....	162	11.4 小结.....	182
11.3.2 读写文件描述符.....	164	第 12 章 文件和目录操作.....	183
11.3.3 使用 ftruncate 缩短文件.....	166	12.1 标准文件函数.....	183
11.3.4 使用 lseek 定位文件指针.....	166		
11.3.5 使用 fsync 同步到硬盘.....	167		
11.3.6 使用 fstat 获得文件信息.....	167		

12.1.1 打开和关闭文件.....	183	12.2.7 删除和改名.....	189
12.1.2 读写文件.....	184	12.2.8 使用临时文件.....	190
12.1.3 获得文件状态.....	185	12.3 目录操作.....	191
12.2 输入输出调用.....	186	12.3.1 找到当前目录.....	191
12.2.1 格式化输出.....	186	12.3.2 改变目录.....	191
12.2.2 格式化输入.....	187	12.3.3 创建和删除目录.....	191
12.2.3 字符输入输出.....	187	12.3.4 获得目录列表.....	192
12.2.4 行输入输出.....	188	12.4 特殊的 ext2 文件系统属性.....	192
12.2.5 文件定位.....	189	12.5 小结.....	195
12.2.6 缓冲区控制.....	189		

第 3 部分 进程和同步

第 13 章 进程控制.....	196	第 14 章 线程概述.....	229
13.1 Linux 进程模型.....	196	14.1 什么是线程.....	229
13.2 进程属性.....	196	14.2 __clone 函数调用.....	229
13.2.1 进程标识号.....	197	14.3 pthread 接口.....	230
13.2.2 Real 和 Effective 标识号.....	198	14.3.1 pthread 是什么.....	230
13.2.3 SetUID 和 SetGID 程序.....	198	14.3.2 何时使用 Pthread.....	231
13.2.4 用户和用户组信息.....	200	14.3.3 pthread_create 函数.....	231
13.2.5 附加的进程信息.....	201	14.3.4 pthread_exit 函数.....	231
13.3 创建进程.....	208	14.3.5 pthread_join 函数.....	232
13.3.1 使用 system 函数.....	208	14.3.6 pthread_atfork 函数.....	234
13.3.2 fork 系统调用.....	209	14.3.7 取消线程.....	234
13.3.3 exec 函数族.....	211	14.3.8 pthread cleanup 宏.....	237
13.3.4 使用 popen 函数.....	213	14.3.9 Pthread 条件.....	238
13.4 控制进程.....	214	14.3.10 pthread_equal 函数.....	238
13.4.1 等待进程——wait 函数族.....	214	14.3.11 线程属性.....	239
13.4.2 杀死程序.....	216	14.3.12 互斥.....	240
13.5 信号.....	218	14.4 小结.....	244
13.5.1 什么是信号.....	218	第 15 章 访问系统信息.....	245
13.5.2 发送信号.....	219	15.1 进程信息.....	246
13.5.3 捕获信号.....	221	15.1.1 cmdline 文件.....	246
13.5.4 检测信号.....	226	15.1.2 environ 文件.....	246
13.6 进程调度.....	227	15.1.3 fd 目录.....	247
13.7 小结.....	228	15.1.4 mem 文件.....	247

15.1.5	stat.....	247	16.1.2	calloc 函数的使用	257
15.1.6	status 文件	249	16.1.3	realloc 函数的使用	257
15.1.7	cwd 符号链接	249	16.1.4	free 函数的使用	257
15.1.8	exe 符号链接	249	16.1.5	alloca 函数的使用	258
15.1.9	maps 文件	249	16.2	内存映像文件	258
15.1.10	root 符号链接	249	16.2.1	mmap 函数的使用	259
15.1.11	statm 文件	249	16.2.2	munmap 函数的使用	260
15.2	一般系统信息	249	16.2.3	msync 函数的使用	260
15.2.1	/proc/cmdline 文件	249	16.2.4	mprotect 函数的使用	260
15.2.2	/proc/cpuinfo 文件	250	16.2.5	锁定内存	261
15.2.3	/proc/devices 文件	250	16.2.6	mremap 函数的使用	261
15.2.4	/proc/dma 文件	250	16.2.7	用内存映像实现 cat 命令	261
15.2.5	/proc/file systems 文件	250	16.3	发现并修改内存问题	263
15.2.6	/proc/interrupts 文件	250	16.3.1	一个有问题的程序	263
15.2.7	/proc/ioports 文件	250	16.3.2	Electric Fence	265
15.2.8	/proc/kcore 文件	250	16.4	小结	267
15.2.9	/proc/kmsg 文件	250	第 17 章	进程间通信	268
15.2.10	/proc/ksyms 文件	251	17.1	管道	268
15.2.11	/proc/loadavg 文件	251	17.1.1	打开和关闭管道	269
15.2.12	/proc/locks 文件	251	17.1.2	读写管道	271
15.2.13	/proc/mdstat 文件	251	17.1.3	更简单的方法	274
15.2.14	/proc/meminfo 文件	251	17.2	FIFO	276
15.2.15	/proc/misc 文件	251	17.2.1	理解 FIFO	276
15.2.16	/proc/modules 文件	251	17.2.2	创建 FIFO	278
15.2.17	/proc/mounts 文件	251	17.2.3	打开和关闭 FIFO	279
15.2.18	/proc/pci 文件	251	17.2.4	读写 FIFO	279
15.2.19	/proc/rtc 文件	252	17.3	System V IPC 概述	282
15.2.20	/proc/stat 文件	252	17.3.1	System V IPC 的主要概念	282
15.2.21	/proc/uptime 文件	252	17.3.2	System V IPC 的问题	284
15.2.22	/proc/version 文件	252	17.3.3	Linux 和 System V IPC	285
15.2.23	/proc/net 子目录	252	17.4	共享内存	285
15.2.24	/proc/scsi 子目录	253	17.4.1	创建共享内存区	286
15.2.25	/proc/sys 子目录	253	17.4.2	附加共享内存区	288
15.3	未来内核中/proc 的变化	255	17.5	消息队列	291
15.4	小结	255	17.5.1	创建和打开消息队列	291
第 16 章	内存管理	256	17.5.2	向队列中写入消息	293
16.1	C 内存管理回顾	256	17.5.3	读取队列中的消息	295
16.1.1	malloc 函数的使用	256	17.5.4	删除消息队列	297

17.6 信号灯	299	18.2.1 函数调用	305
17.6.1 创建信号灯	299	18.2.2 出错处理	308
17.6.2 控制和删除信号灯	301	18.3 和守护进程通信	311
17.7 小结	303	18.3.1 读取配置文件	311
第 18 章 守护进程	304	18.3.2 向守护进程加入信号处理功能	314
18.1 理解守护进程	304	18.4 小结	319
18.2 创建守护进程	304		

第 4 部分 网络编程

第 19 章 TCP/IP 和套接口编程	320	19.6 通过其他编程语言使用套接口	339
19.1 套接口的定义	320	19.7 UNIX 域套接口的 Perl 编程	339
19.2 通信域	321	19.8 监视套接口活动的工具	341
19.3 套接口编程基础	321	19.9 小结	342
19.3.1 分配套接口和初始化	321	第 20 章 UDP: 用户数据报协议	343
19.3.2 完成连接的系统调用	323	20.1 UDP 概述	343
19.3.3 传送数据	324	20.1.1 UDP 和 TCP 的对比	343
19.3.4 关闭	325	20.1.2 TCP 的优缺点	343
19.4 使用套接口的客户机/服务器例子	325	20.1.3 UDP 的优缺点	344
19.4.1 服务器的例子程序	326	20.1.4 选择使用哪一种协议	344
19.4.2 客户机的例子程序	328	20.2 实现一个基于 UDP 的应用	345
19.4.3 运行客户机和服务器的例子	331	20.2.1 使用 UDP 发送数据	345
19.4.4 使用 Web 浏览器作为客户机	331	20.2.2 接收 UDP 数据	348
19.5 一个简单的 Web 服务器和 Web 客	331	20.2.3 最少的出错检查	350
19.5.1 实现一个简单的 Web 服务器	332	20.2.4 非阻塞 I/O	355
19.5.2 实现一个简单的 Web 客户机	336	20.3 小结	361
19.5.3 测试 Web 服务器和 Web 客户机	338	第 21 章 多播套接口和非阻塞 I/O	362
19.5.4 使用 Netscape Navigator 作为客户	339	21.1 配置 Linux 支持多播 IP	362
		21.2 为支持多播 IP 重新编译 Linux 内核	363
		21.3 多播 IP 广播的示例程序	363
		21.3.1 使用多播 IP 广播数据	364
		21.3.2 创建客户程序监听多播 IP 广播	366

21.3.3 运行多播 IP 示例程序	370
---------------------------	-----

21.4 小结	371
---------------	-----

第 5 部分 用户界面编程

第 22 章 底层终端控制

372

22.1 终端接口	372
22.2 控制终端	373
22.2.1 属性控制函数	374
22.2.2 速度控制函数	375
22.2.3 行控制函数	376
22.2.4 进程控制函数	377
22.3 使用终端接口	378
22.4 改变终端模式	380
22.5 使用 terminfo	382
22.5.1 terminfo 能力	382
22.5.2 terminfo 编程	383
22.5.3 发挥 terminfo 能力	387
22.6 小结	390

第 23 章 ncurses 入门

391

23.1 ncurses 简史	391
23.2 使用 ncurses 编译程序	392
23.3 调试 ncurses 程序	392
23.4 关于窗口	392
23.4.1 ncurses 窗口设计	393
23.4.2 ncurses 函数命名规则	394
23.5 初始化和终止	395
23.5.1 ncurses 初始化结构	395
23.5.2 ncurses 终止	395
23.5.3 说明 ncurses 初始化和终止	396
23.6 输入和输出	398
23.6.1 输出例程	398
23.6.2 输入例程	406
23.7 色彩例程	409
23.8 窗口管理	411
23.9 其他各种工具函数	412

23.10 小结	415
----------------	-----

第 24 章 ncurses 高级编程

416

24.1 其他 ncurses 功能	416
24.1.1 鼠标支持	416
24.1.2 菜单支持	416
24.1.3 窗体支持	416
24.2 和鼠标交互	417
24.2.1 鼠标 API 概述	417
24.2.2 鼠标控制例程	418
24.2.3 示例程序	419
24.3 使用菜单	422
24.3.1 菜单 API 概述	422
24.3.2 菜单控制例程	423
24.3.3 示例程序	428
24.4 ncurses 窗体	430
24.4.1 窗体 API 概述	430
24.4.2 窗体管理例程	431
24.4.3 示例程序	440
24.5 小结	441

第 25 章 X Windows 编程

442

25.1 X 的概念	443
25.2 Xlib API	444
25.2.1 XOpenDisplay	445
25.2.2 XCreateSimpleWindow 和 XCreateWindow	445
25.2.3 映射窗口和撤销映射窗口	446
25.2.4 撤销窗口	447
25.2.5 事件处理	447
25.2.6 初始化图形设备上下文和字体	448
25.2.7 在 X 窗口中绘图	449

25.2.8 一个 Xlib 的示例程序.....	449	26.5.4 Button 类.....	491
25.3 X Toolkit API.....	455	26.5.5 Text 类.....	492
25.3.1 X Toolkit 使用入门.....	455	26.6 小结.....	494
25.3.2 使用 X 工具包设置窗口部件参数.....	456	第 27 章 使用 GTK+ 进行 GUI 编程... 495	
25.4 XFree86.....	458	27.1 GTK+ 简介.....	496
25.4.1 DPMS——显示器电源管理信令.....	458	27.1.1 在 GTK+ 中处理事件.....	497
25.4.2 DRI——直接显示接口.....	458	27.1.2 使用 GTK+ 的简短示例程序....	498
25.4.3 DGA——直接图形体系结构..	458	27.1.3 各种 GTK 窗口部件.....	500
25.4.4 XV——X 视频.....	459	27.1.4 GTK 容器窗口部件.....	501
25.5 小结.....	459	27.2 一个用于显示 XML 文件的 GTK+ 程序.....	502
第 26 章 Athena、Motif 和 LessTif 窗口部件..... 460		27.2.1 XML 简介.....	502
26.1 使用 Athena 的窗口部件.....	460	27.2.2 James Clark 的 XML 分析器 expat.....	503
26.1.1 Athena 的标签窗口部件.....	460	27.2.3 实现 GTK+ 的 XML 显示程序.....	504
26.1.2 Athena 的命令按钮窗口部件..	461	27.2.4 运行 GTK+ 的 XML 显示程序.....	510
26.1.3 Athena 的列表窗口部件.....	464	27.3 一个使用 Notebook 窗口部件的 GUI 程序.....	510
26.1.4 Athena 的文本窗口部件.....	465	27.3.1 Notebook 窗口部件示例程序的实现.....	510
26.1.5 Athena 的简单菜单窗口部件..	468	27.3.2 实现 Drawing 窗口部件.....	512
26.2 使用 Motif 的窗口部件.....	470	27.3.3 运行 GTK Notebook 窗口部件的示例程序.....	515
26.2.1 Motif 的标签窗口部件.....	471	27.4 通过其他编程语言使用 GTK+.....	515
26.2.2 Motif 的列表窗口部件.....	472	27.4.1 通过 C++ 使用 GTK+.....	516
26.2.3 Motif 的文本窗口部件.....	474	27.4.2 通过 Perl 使用 GTK+.....	517
26.3 编写一个定制的 Athena 窗口部件.....	477	27.4.3 通过 Python 使用 GTK+.....	518
26.3.1 使用 fetch_url.c 文件.....	477	27.5 GTK+ 的 RAD 工具.....	518
26.3.2 使用 URL.h 文件.....	479	27.6 小结.....	519
26.3.3 使用 URLP.h 文件.....	480	第 28 章 使用 Qt 进行 GUI 编程..... 520	
26.3.4 使用 URL.c 文件.....	481	28.1 通过重载 QWidget 类方法处理事件.....	521
26.3.5 测试 URLWidget.....	484	28.1.1 QWidget 类概述.....	521
26.4 在 C++ 程序中使用 Athena 和 Motif.....	485	28.1.2 实现 DrawWidget 类.....	523
26.5 使用封装 Athena 窗口部件的一个 C++ 类库.....	486	28.1.3 测试 DrawWidget.....	525
26.5.1 Component 类.....	487		
26.5.2 PaneWindow 类.....	488		
26.5.3 Label 类.....	490		

28.2 使用 Qt 槽和信号处理事件	526	29.3.3 使用 GLUT 创建简单的 3D 对象	541
28.2.1 派生 StateLCDWidget 类	526	29.3.4 使用 x-y-z 坐标在 3D 空间中放置对象	542
28.2.2 使用信号和槽	529	29.3.5 沿着 x-、y-、z-中任一坐标轴或所有坐标轴旋转对象	543
28.2.3 运行信号/槽示例程序	531	29.3.6 启用 Material 属性	544
28.3 用 Qt 实现 XMLview 的程序	531	29.3.7 启用深度测试	545
28.3.1 SAX2: 一个用于 XML 的简单 API	532	29.3.8 处理键盘事件	545
28.3.2 DOM: 文档目标对象	533	29.3.9 为获得动画效果更新 OpenGL 图形	545
28.4 小结	537	29.3.10 Orbits 程序清单	546
第 29 章 使用 OpenGL 和 Mesa 进行 3D 图形编程	538	29.4 纹理映像	548
29.1 需要为本章准备什么	538	29.4.1 用纹理面产生立方体	548
29.2 使用 OpenGL	539	29.4.2 创建纹理映像	549
29.3 3D 图形编程	539	29.4.3 立方体程序清单	550
29.3.1 orbits.c	539	29.5 小结	554
29.3.2 为 OpenGL 图形创建窗口并初始化 OpenGL	540		

第 6 部分 特殊编程技术

第 30 章 使用 GNU Bash 进行 Shell 编程	555	30.5.3 不确定性循环: while 和 until	569
30.1 为何使用 bash	555	30.5.4 选择结构: case 和 select	569
30.2 bash 基础知识	555	30.6 shell 函数	572
30.2.1 通配符	556	30.7 输入与输出	573
30.2.2 花括号展开式	556	30.7.1 I/O 重定向	573
30.2.3 特殊字符	557	30.7.2 字符串 I/O	574
30.3 使用 bash 变量	558	30.8 命令行处理	576
30.4 使用 bash 操作符	561	30.9 进程和作业控制	578
30.4.1 字符串操作符	561	30.9.1 Shell 的信号处理	578
30.4.2 模式匹配操作符	562	30.9.2 使用 trap	578
30.5 流控制	564	30.10 小结	580
30.5.1 条件执行: if	564	第 31 章 设备驱动程序	581
30.5.2 确定性循环: for	568	31.1 驱动程序的类型	581

31.1.1 静态链接的内核设备驱动程序	581	31.5.1 低层端口的 I/O.....	589
31.1.2 可加载的内核模块.....	582	31.5.2 使用 DMA 访问内存.....	591
31.1.3 共享库	582	31.5.3 引发使用设备驱动程序的中断	591
31.1.4 无特权用户模式程序.....	582	31.5.4 设备驱动程序分层	592
31.1.5 特权用户模式程序.....	583	31.6 简单的用户模式测试驱动程序.....	593
31.1.6 守护进程	583	31.7 创建内核驱动程序	594
31.1.7 字符设备与块设备的对比.....	583	31.7.1 查看源代码.....	594
31.2 怎样构造硬件.....	583	31.7.2 编译驱动程序	619
31.2.1 理解步进电机的工作原理.....	584	31.7.3 使用内核驱动程序	620
31.2.2 标准的或双向的并口.....	586	31.7.4 未来发展方向	621
31.3 建立开发环境.....	588	31.8 其他信息资源.....	621
31.4 调试内核级驱动程序.....	589	31.9 小结.....	622
31.5 设备驱动程序内幕.....	589		

第 7 部分 补充内容

第 32 章 软件包管理623

32.1 理解 tar 文件	623
32.1.1 创建 tar 文件	624
32.1.2 更新 tar 文件	625
32.1.3 列出 tar 文件的内容.....	626
32.1.4 从一个存档文件解出文件.....	627
32.2 理解 install 命令.....	628
32.3 理解 Red Hat 包管理器 (RPM) ...	630
32.3.1 RPM 是什么	630
32.3.2 最小要求	631
32.3.3 配置 RPM	631
32.3.4 控制构造过程: 使用 spec 文件	633
32.3.5 分析一个 spec 文件.....	634
32.3.6 构造软件包.....	637
32.4 文件层次结构标准.....	637
32.5 小结	639

第 33 章 建档..... 640

33.1 编写手册页面.....	640
33.1.1 手册页面的组成.....	640
33.1.2 手册页面的例子.....	641
33.1.3 使用 groff 命令.....	643
33.1.4 Linux 约定	644
33.2 使用 DocBook	645
33.2.1 DocBook 是什么.....	646
33.2.2 DocBook 标记	646
33.2.3 DocBook 文档示例.....	647
33.2.4 生成输出.....	653
33.3 小结.....	653

第 34 章 许可证的发放 654

34.1 介绍和弃权.....	654
34.2 MIT/X 风格的许可证.....	654
34.3 BSD 风格的许可证	655

34.4 Artistic 的许可证.....	656	34.5.2 GNU 库通用公共许可证 (LGPL)		658
34.5 GNU 通用公共许可证	656			
34.5.1 GNU 通用公共许可证 (GPL)		34.6 开发源代码的定义		658
.....	657	34.7 小结.....		660

第 1 部分 Linux 编程工具包

第 1 章 Linux 及 Linux 编程综述

Linux 不再是爱好者的玩具了。它已经成为几乎每一种计算基础设施，尤其是因特网必不可少的组成部分。如果说编写本书第一版的 1998 年是 Linux 最终出现在美国的雷达屏幕上的一年，那么到了 1999 年 Linux 已经是一名美国正式公民了。再向前看，在 2000 年里，Linux 肯定会把自己稳固树立为因特网计算基础设施的一个基本组成部分以及新经济的一名正式成员。

1.1 Linux 变得成熟了

熟悉 Linux 的程序员尤其是应用程序开发人员都知道 Linux 市场正在爆炸性地增长。本节简要说明了这一情况发生的原因。如果你熟悉 Linux 的历史，可以直接跳到下一节。

1.1.1 Linux 的昨天

Linux 的发布历史始于 1991 年 8 月发表在 Usenet 新闻组 comp.os.minix 上的如下一篇文章，作者是一名芬兰的大学生：

```
Hello everybody out there using minix-  
I'm doing a (free) operating system (just a hobby, won't be big and  
professional like gnu) for 386 (486) AT clones. This has been brewing  
since april, and is starting to get ready. I'd like any feedback on  
things people like/dislike in minix, as my OS resembles it somewhat  
(same physical layout of the file-system (due to practical reasons)  
among other things).
```

这个学生就是 Linus Torvalds，而他所编写的“业余爱好”Linux 已经发展成为现在的 Linux。Linux 1.0 版内核发布于 1994 年 3 月 14 日。1996 年 6 月发布了 2.0 版，在 1.0 版和 2.0 版内核之间，大量的开发工作极大地提高了基本的内核功能、增加了设备支持，最重要的是扩大了可以使用的应用程序的范围。到了发布 2.0 版内核的时候，几乎没有用 Linux 执行不了的任务，只是界面尚不理想（理想的意思是“类似于 Windows”）。

注意：实际上，Linus 最初关于 Linux 的文章出现在 1991 年 7 月 3 日，但当时并没有特别提到 Linux。Linus 本人讲述的 Linux 早期开发的一段有趣的历史可以从网上得到：<http://www.li.org/li/linuxhistory.shtml>。

目前稳定的 2.2 版内核正式发布于 1999 年 1 月 25 日。编写本书的时候, 2.3 版的开发版内核处于代码冻结状态。此时内核不再增加新特性, 按照发布 2.4 版内核的目标, 代码编写工作转向排错并稳定内核的基础代码。该版本的目标是包括一个日志型文件系统、使对称多处理子系统有更好的粒度, 以及对 POSIX 标准更完备的支持。上面的几点介绍没有涵盖增强和排错工作的所有方面。

1998 年 3 月, 当 Netscape 承诺在 GNU 计划的 GPL (General Public License, 通用公共许可证) 的一个修订版本的基础上公开 Netscape Communicator Internet 套件的源代码时, Linux 一下子进入了大众视野。同年 7 月, 世界上最大的两家关系数据库厂商 Informix 和 Oracle 宣布把他们的数据库产品移植到了 Linux 上。1998 年 8 月, Intel 和 Netscape 公司购买了 Linux 发布商中的领头羊 Red Hat 公司的少量股票, 专项资金投入了 Linux 世界。同时, IBM 开始了它的旗舰数据库产品 DB/2 在 Linux 上移植版本的 beta 测试版。这一年中还有许多类似的开发工作在继续实施。

1999 年 Linux 成为主流 IT 市场中的年青一分子。主要的软硬件厂商继续向 Linux 公司进行投资, 随后很快又投入了风险资金。不出所料, 随着 Red Hat 和 VA Linux 首先成为上市的 Linux 公司, 从大型投资商注入的资金取得了成功的股票销售行情。

2000 年的头几个月, IBM 宣布将在它的服务器、台式机以及便携机产品线上预装 Linux。实际上, 截止到 2000 年 4 月, 大多数主要的硬件厂商都把 Linux 作为其所售计算机系统上的一种候选软件。公众对于 Linux 的兴趣与日俱增, 计算机业内几乎每周都有与 Linux 相关的消息发布。更令人兴奋的是大众媒体对于 Linux 的关注。实际上, Linux 已经不再仅仅是爱好者们的一个玩具了。

与此同时, 那个位于雷蒙德^①市的小^②软件公司微软再也坐不住了。在微软著名(也可以说臭名昭著)的万圣节文件泄漏之后, 迫使该公司对这种迅速发迹的操作系统予以回应。万圣节文档是微软内部的备忘录, 该文档详细地分析了微软对于 Linux 对其市场霸权尤其是服务器操作系统 Windows NT 的威胁, 并且讨论了用以对付 Linux 挑战的策略。根据万圣节文档制订的策略, 微软发起了一个“恐惧、不可靠、怀疑”(FUD, 三个字母分别代表 Fear、Uncertainty 和 Doubt) 运动, 其后果是很快否认 Mindcraft 公司有关服务器性能的评测报告, 并发表违背常识的带有偏见的白皮书“Linux 神话”。Linux 社群中的大多数人都嘲笑微软的这些伎俩, 而在更大范围的计算机世界里, 特别是鉴于微软最初的结论曾经还是开放源代码的开发方式要比传统的软件开发模型更优越这一事实, 通常很少有人会严肃看待微软关于 Linux 或开放源代码软件所发表的声明。然而, 微软被迫用 FUD 的策略来对付 Linux 的事实表明它对于自己的产品没有信心。

注意: Mindcraft 公司最初的研究报告载于 <http://www.mindcraft.com/whitepapers/nts4rhlinux.html>。“Linux 神话”一文可通过 <http://www.microsoft.com/NTServer/nts/news/msnw/LinuxMyths.asp> 在线阅览。

① 译者注: 雷蒙德(Redmond)是位于美国华盛顿州西雅图东北的一个小城, 微软总部所在地。

② 译者注: 众所周知, 微软是世界上最大的软件公司, 作者在这里用“小”软件公司来形容它, 一方面表示作者对微软的鄙视, 另一方面也隐含了微软公司名称中的“小”(Micro)。

1.1.2 Linux 的今天

作为一种服务器级的操作系统, Linux 已经成熟了。提供 Web 服务器的 Linux 系统遍布全球, 而且越来越多的商业用户使用 Linux 系统提供文件和打印服务。它既被当作邮件服务器的一种候选平台, 也被当作一种强壮而安全的防火墙。业界认为 Linux 在因特网服务器市场将会取得更大份额(绝大多数是原 Windows NT 和 Windows 2000 的市场份额), 并且预计 Linux 还会在嵌入式设备市场、因特网设备市场和专门性服务器市场占据主导地位。

Linux 的企业级特性, 比如支持多处理器、支持大型文件系统、日志型文件系统以及密集型计算和高可用性集群技术, 也会逐步成熟。2.2 版内核最多支持 16 个 CPU, 比 2.0 版最多支持 4 个 CPU 有了提高。密集型计算集群技术让 Linux 用户可以创建包含数十乃至数百台廉价的日用型个人计算机的系统, 从而达到超级计算机水平的处理速度, 而价格同 Cray、SGI 或 SUN 的超级计算机相比极为低廉。高可用性集群技术让 Linux 系统在一个或多个要害部件(比如电源或硬盘)出现故障时, 仍然能够继续正常工作。

桌面上的 Linux 也在继续完善。KDE 桌面提供的图形用户界面(GUI, Graphical User Interface)在易用性和可配置方面都能和微软的 Windows 相媲美。但与 Windows 不同的是, KDE 只是运行在健壮而灵活的操作系统之上为改善视觉效果而加入的一层软件。KDE 强大的命令行界面是用户垂手可得的强大工具。Linux 有不少于四种以上办公软件套件: Applixware、StarOffice 以及作为 KDE 计划一部分的 KOffice 都已投入实际使用。Corel 发行了它的 Linux 办公套件 WordPerfect Office 2000, 以及它自己的 Linux 发布版本 Corel LINUX OS。在大量 Linux 应用程序和工具软件的基础上, 与微软 Office 类似的办公软件的出现使得 Linux 成为 Windows 在桌面系统上的一个竞争对手。

1.1.3 Linux 的明天

Linux 将会何去何从? Linux 因其强壮、稳定、功能强大的特点而在服务器领域极为流行, 但它因为在实用性方面持续遭遇挑战而仍然没有在桌面上超越 Windows。在渐露头角的因特网设备, 比如防火墙和路由器市场上, Linux 是充满希望的, 但还没有在数量上得到完全验证。缩微版本的 Linux 在嵌入式设备, 比如电话和机器控制器市场上逐渐流行起来。随着 Linux 厂商, 比如 Caldera、Red Hat、VA Linux 以及 Cobalt 作为上市公司逐步成熟, 它们必然会推进自己的商业计划和策略, 并指导 Linux 的未来。

1.2 为何选择 Linux 编程

人们为什么会用 Linux 编程, 又为什么会为 Linux 编程呢? 这个问题的答案可能会和从事 Linux 编程的人数一样多。但我认为, 大多数答案可以归结为以下几类。

首先, Linux 编程很有意思——这就是我做这件事的原因。其次, Linux 是自由软件。第三, Linux 是开放的。它没有隐藏的接口, 也没有未公开的功能或 API (Application Programming Interface, 应用程序编程接口)——在了解操作系统的功能方面, 没有人或组