

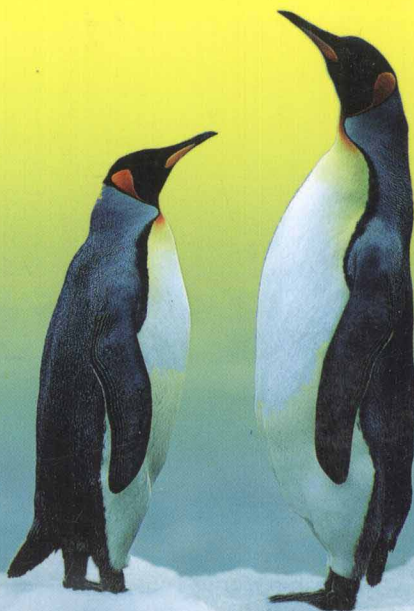
LINUX



- ❖ 全面的基础知识，帮助读者完成 Linux 系统下内存、CPU、磁盘、网络及音频设备的访问及管理
- ❖ 案例为指导，每一个知识点中都实现了一个应用案例，读者可以针对每一个知识点进行实例编程演练
- ❖ 紧扣技术前沿，所采用的开发平台为新的 Linux 内核，开发工具为新的 GCC

高级程序设计 (第三版)

◆ 杨宗德 吕光宏 刘雍 编著



Linux



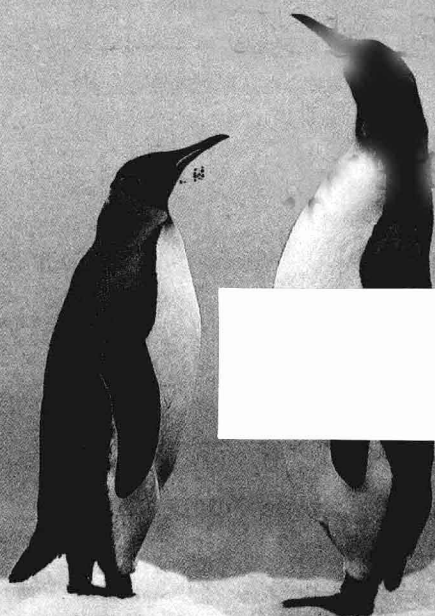
人民邮电出版社
POSTS & TELECOM PRESS

UNIX

高级程序设计 (第三版)

◆ 杨宗德 吕光宏 刘雍 编著

Linux



人民邮电出版社
北京

图书在版编目 (C I P) 数据

Linux高级程序设计 : 第3版 / 杨宗德, 吕光宏, 刘
雍编著. -- 3版. -- 北京 : 人民邮电出版社, 2012.11
ISBN 978-7-115-29290-2

I. ①L… II. ①杨… ②吕… ③刘… III. ①
Linux操作系统—程序设计 IV. ①TP316.89

中国版本图书馆CIP数据核字(2012)第207497号

内 容 提 要

本书围绕 Linux 操作系统“一切都是文件”的特点, 讲述了 Linux 操作系统下应用层“一段执行单元(进程)对系统资源(CPU 资源、各类文件资源)的管理”。详细介绍了 Linux 系统编程环境及编程工具(GCC/Makefile/GDB)、文件管理(文件属性控制、ANSI 以及 POSIX 标准下文件读写操作、终端编程)、进程管理(创建、退出、执行、等待、属性控制)、进程间通信(管道、消息队列、共享内存)、进程间同步机制(信号量)、进程间异步机制(信号)、线程管理(创建、退出、取消等以及属性控制)、线程间同步(互斥锁、读写锁、条件变量)、线程与信号以及 BSD socket 编程中的 TCP、UDP、原始套接口、网络服务器应用开发等内容, 并对 Linux 系统下的音频应用程序开发做了讲解。

本书内容丰富、紧扣应用, 适合从事 Linux 下 C 应用编程的人员阅读, 也适合从事嵌入式 Linux 开发的人员阅读。

Linux 高级程序设计 (第三版)

- ◆ 编 著 杨宗德 吕光宏 刘 雍
责任编辑 张 涛
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
中国铁道出版社印刷厂印刷
- ◆ 开本: 787×1092 1/16
印张: 31
字数: 953 千字 2012 年 11 月第 3 版
印数: 10 801—14 300 册 2012 年 11 月北京第 1 次印刷

ISBN 978-7-115-29290-2

定价: 59.00 元

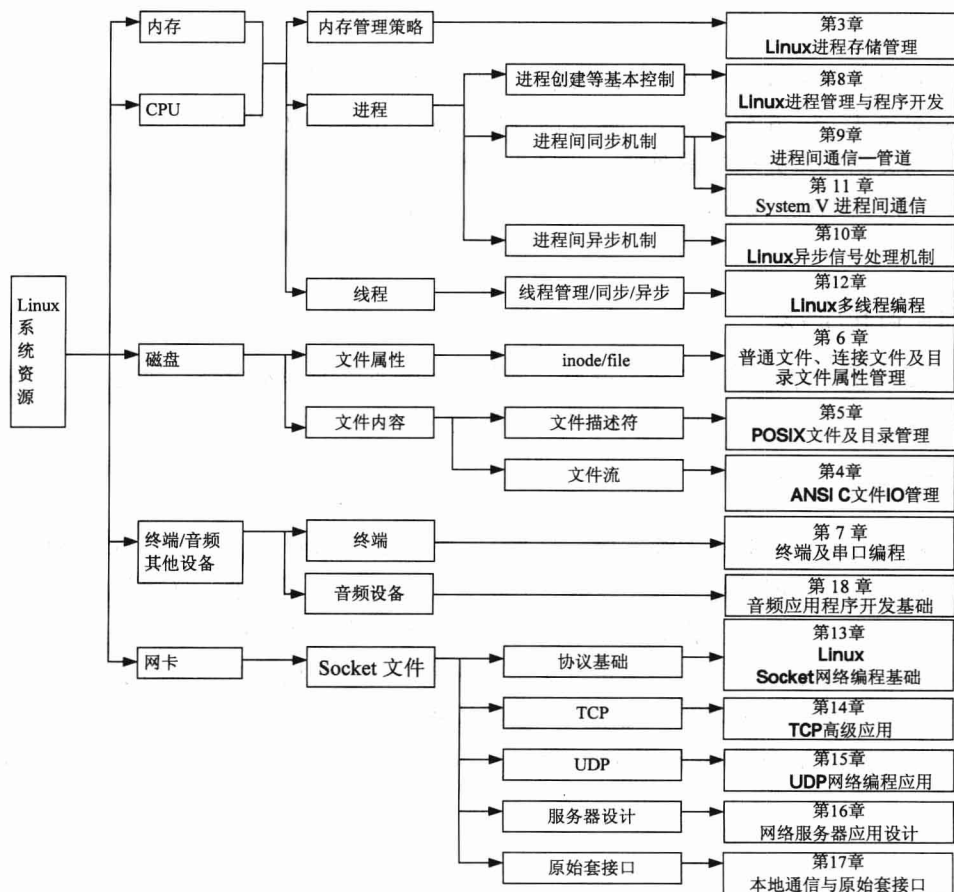
读者服务热线: (010)67132692 印装质量热线: (010)67129223
反盗版热线: (010)67171154

前 言

Linux 应用开发是目前最为广泛的软件开发内容之一，同时也是从事 Linux 内核及驱动开发的基础。《Linux 高级程序设计》一书经过两次出版，收到了大量的读者来信，对本书提出了各种意见和建议，同时，随着技术的更新，新技术、新应用不断涌现，综合各方面的考虑，笔者做了大量的修订工作，推出了第三版。

本书主要内容

本书围绕 Linux 操作系统“一切都是文件”的特点，讲述了 Linux 操作系统下应用层“一段执行单元（进程）对系统资源（CPU、内存、磁盘、外设、网络设备）的管理”，重点介绍了进程、线程、文件属性访问、文件内容访问、Socket、终端和音频设备编程等重要概念。整个内容安排如下图所示，完全涵盖了应用开发中涉及的所有内容。





本书主要特点

(1) 内容丰富。本书是作者多年计算机教学及工程经验总结,整合了 Linux 应用编程的绝大多数知识点,几乎涵盖了 Linux 操作系统下 C 应用编程的所有内容,包括工具使用及环境设置、文件及文件管理、进程及进程管理、进程间通信、线程及线程管理、线程通信、网络及网络应用编程等知识点。

(2) 循序渐进。本书在写作思路避开了大量理论的介绍,按知识体系介绍→应用函数分析→应用案例开发的写作顺序,让读者在掌握具体知识点的同时可以掌握实例的具体实现。

(3) 案例指导。本书中所有调用函数及引用都标出具体的出处(在 Linux 操作系统中的文件位置),读者可以一目了然地知道对应函数及类型的定义过程。另外,本书遵循案例教学思想,每一个知识点都讲解一个应用程序,且所有代码都在教学实践过程中调试通过,读者可以直接使用。

(4) 紧扣应用。本书所列代码和实例都来源于具体的应用程序。

本书修订说明

这次修订在第二版的基础上增加了大量的应用案例及新的知识体系。

(1) 新增加第 7 章终端编程,第 17 章本地通信及原始套接口内容以及第 18 章音频编程内容。

(2) 对第 2 章编程工具及第 12 章线程编程章节做了适当的合并和删简。

(3) 对第 4、5、6 章磁盘文件管理内容新增加了 `tree` 等应用案例。

(4) 为突出异步处理的重要性,专门使用第 10 章介绍进程的异步信息处理。

(5) 对网络编程知识体系进行了重新整编,根据应用协议更新了第 13、14、15、16、17 章内容。

对读者的假定

本书要求读者有较好的 C 语言基础,熟悉 Linux 系统的基本命令。如果读者对操作系统或 Linux 内核有一定了解,则更容易学习本书。

本书编写工作

本书所有内容由杨宗德主编完成,吕光宏、刘雍、邓玉春、曾庆华参与本书相关代码的编写及审稿工作。同时感谢何伟、张兵、刘兆宏、季建华、刘福刚、赵文革、黄弦等老师对本书的编写提出了大量宝贵指导意见,另外感谢石昀、朱元斌、钱文杰、陈功杰、汪洪、刘超、钟晓媛、刘梨平、石霞等同学试读了本书初稿,为本书相关内容提出了宝贵意见。由于时间仓促,本书难免有疏忽和不足之处,恳请读者批评赐教。

源程序下载地址为: www.ptpress.com.cn。联系邮箱为: zhangtao@ptpress.com.cn。

编 者

目 录

第 1 章 Linux 下 C 语言开发环境1	第 3 章 Linux 进程存储管理35
1.1 Linux 操作系统简介.....2	3.1 Linux 程序存储结构与进程结构.....36
1.1.1 Linux 操作系统简介.....2	3.1.1 Linux 可执行文件结构.....36
1.1.2 GNU/Linux 简介.....3	3.1.2 Linux 进程结构.....37
1.1.3 相关术语介绍.....3	3.1.3 C 变量及函数存储类型.....39
1.2 Linux 开发初步.....5	3.1.4 栈和堆的区别.....44
1.2.1 Linux 下 C 程序标准.....5	3.1.5 示例：查看代码中各数据存储位置.....45
1.2.2 库函数和系统调用.....7	3.1.6 常见内存错误示例分析.....48
1.2.3 在线文档介绍.....8	3.2 ANSI C 动态内存管理.....50
1.2.4 获取错误信息.....9	3.2.1 内存分配的基本方式.....50
1.3 部分常用工具简介.....10	3.2.2 示例：为程序申请动态内存空间.....50
1.3.1 tar 打包器.....10	3.2.3 内存数据管理函数.....54
1.3.2 Linux 常用命令及工具.....11	3.3 Valgrind 及 valkyrie 内存管理工具.....56
1.4 Linux 下编码风格.....15	3.3.1 Valgrind 介绍.....57
1.4.1 GNU 编码规范.....16	3.3.2 Valgrind 安装与使用.....59
1.4.2 Linux 内核编码规范.....17	3.3.3 valgrind 图形化工具 Valkyrie.....61
第 2 章 Linux 下 C 语言开发工具19	3.3.4 内存检测示例.....62
2.1 常用编辑工具.....20	3.4 Linux 进程环境及系统限制.....64
2.1.1 VIM 编辑器.....20	3.4.1 进程与命令行选项及参数.....64
2.1.2 Emacs 编辑器.....22	3.4.2 进程与环境变量.....69
2.1.3 Source Insight 工具.....23	3.4.3 Linux 系统限制.....70
2.2 GCC/GDB 编译调试工具基础.....27	3.4.4 Linux 时间管理.....72
2.2.1 GCC/G++ 简单介绍.....28	第 4 章 ANSI C 文件 IO 管理75
2.2.2 GDB 调试工具简介.....30	4.1 文件及文件流.....77
2.2.3 使用 GCC 编译 C 程序示例.....31	
2.2.4 使用 g++ 编译 C++ 程序示例.....32	
2.2.5 GDB 演示示例.....33	



4.1.1	文件与流的基本概念	77	5.3.5	当前工作路径操作	122
4.1.2	标准流及流主要功能	78	5.3.6	文件流、目录流、文件描述符总结	123
4.1.3	文件流指针	79	5.4	应用案例: 递归文件目录复制操作	123
4.1.4	缓冲区类型	81	5.4.1	应用需求及流程图	123
4.1.5	指定流缓冲区	82	5.4.2	示例代码	125
4.2	ANSI C 文件 I/O 操作	85	第 6 章	普通文件、连接文件及目录文件属性管理	128
4.2.1	打开关闭文件	85	6.1	Linux 文件系统管理	129
4.2.2	读/写文件流	86	6.1.1	Linux 下 VFS 虚拟文件系统	129
4.2.3	文件流定位	91	6.1.2	ext2 文件系统结构	130
4.2.4	实现文件复制操作示例	92	6.1.3	目录文件及常规文件存储方法	132
4.3	流的格式化输入/输出操作	94	6.2	Linux 系统下文件类型及属性	132
4.3.1	printf/scanf 函数分析	94	6.2.1	Linux 文件类型及权限	132
4.3.2	fprintf/fscanf 函数分析	95	6.2.2	Linux 文件类型	133
4.3.3	sprintf 函数分析	96	6.2.3	文件权限修饰位	136
4.3.4	sscanf 函数分析	97	6.2.4	文件访问权限位	137
第 5 章	POSIX 文件及目录管理	99	6.3	Linux 文件属性管理	138
5.1	文件描述符与内核文件表项	100	6.3.1	读取文件属性	138
5.1.1	文件流与文件描述符的区别	100	6.3.2	修改文件权限操作	141
5.1.2	文件表结构图	101	6.3.3	修改系统 umask 值	142
5.1.3	文件描述符与文件流的转换操作	101	6.3.4	修改文件的拥有者及组	143
5.2	POSIX 标准下文件 IO 管理	103	6.3.5	用户名/组名与 UID/GID 的转换	144
5.2.1	创建/打开/关闭文件	104	6.3.6	创建/删除硬连接	145
5.2.2	文件控制 fcntl	107	6.3.7	符号连接文件特殊操作	146
5.2.3	读/写文件内容	110	6.3.8	文件时间属性修改与时间处理	147
5.2.4	使用 POSIX IO 实现大于 2G 文件复制	111	6.4	示例: ls -l 以排序方式列出目录信息	149
5.2.5	文件定位	112	6.4.1	需求及知识点涵盖	149
5.2.6	同步内核缓冲区	113	6.4.2	流程及源代码实现	149
5.2.7	映射文件到内存	114			
5.2.8	锁定/解锁文件	116			
5.3	目录流基本操作	118			
5.3.1	打开/关闭目录文件	118			
5.3.2	读/写目录内容	119			
5.3.3	定位目录位置	121			
5.3.4	添加和删除目录	121			

6.5 示例: 实现 tree 系统命令	152	8.2.2 在进程中运行新代码	197
第 7 章 终端及串口编程	156	8.2.3 回收进程用户空间资源	201
7.1 终端设备类型	157	8.2.4 回收内核空间资源	203
7.1.1 实际的物理串口	157	8.2.5 孤儿进程与僵死进程	205
7.1.2 控制台终端	158	8.2.6 修改进程用户相关信息	206
7.1.3 虚拟终端	159	8.3 Linux 特殊进程	210
7.1.4 当前终端	159	8.3.1 守候进程及其创建过程	210
7.2 终端属性控制	160	8.3.2 日志信息及其管理	211
7.2.1 读取/设置终端属性信息	160	8.3.3 守候进程应用示例	214
7.2.2 c_cflag 终端控制选项	161	第 9 章 进程间通信——管道	216
7.2.3 c_lflag 终端本地选项	163	9.1 进程间通信——PIPE	218
7.2.4 c_iflag 终端输入选项	165	9.1.1 无名管道概念	218
7.2.5 c_oflag 终端输出选项	166	9.1.2 无名管道文件操作的特殊性	218
7.2.6 c_cc[NCCS]终端控制字符	166	9.1.3 文件描述符重定向	221
7.2.7 IOCTLs 控制终端	167	9.1.4 实现 who sort	225
7.2.8 进程与终端	168	9.1.5 流重定向	226
7.3 串口编程	169	9.2 进程间通信——FIFO	228
7.3.1 串口物理设备	169	9.2.1 有名管道概念	228
7.3.2 串口终端基本操作	170	9.2.2 有名管道管理及其特殊性	228
7.3.3 串口编程示例	171	9.2.3 管道基本特点总结	232
7.4 控制台终端应用基础	175	第 10 章 Linux 异步信号处理机制	233
7.4.1 终端属性设置	175	10.1 Linux 常见信号与处理	234
7.4.2 控制命令基本格式	176	10.1.1 信号与中断	234
7.4.3 从控制台终端获取信息不回显	178	10.1.2 信号基本概念	236
第 8 章 Linux 进程管理与程序开发	180	10.1.3 信号的生命周期	236
8.1 进程环境及进程属性	181	10.1.4 发送信号	237
8.1.1 程序、进程与进程资源	181	10.2 安装信号与捕获信号	242
8.1.2 进程状态	182	10.2.1 信号处理办法	242
8.1.3 进程基本属性	183	10.2.2 signal 安装信号	243
8.1.4 进程用户属性	187	10.2.3 sigaction 安装信号	244
8.2 进程管理及控制	190	10.2.4 signal 的系统漏洞	248
8.2.1 创建进程	190		



10.3 安装信号与捕获信号.....250	12.1.2 创建线程.....295
10.3.1 设置进程屏蔽信号集...250	12.1.3 线程退出与等待.....297
10.3.2 获取当前未决的信号...251	12.1.4 取消线程.....299
10.3.3 信号集合操作.....251	12.1.5 线程与私有数据.....302
10.3.4 信号集合操作应用 示例.....252	12.2 线程同步机制.....305
10.4 等待信号.....256	12.2.1 互斥锁通信机制.....305
10.4.1 pause 函数.....256	12.2.2 条件变量通信机制.....308
10.4.2 sigsuspend 函数.....256	12.2.3 读写锁通信机制.....314
10.5 信号应用实例.....258	12.3 多线程异步管理——信号.....319
第 11 章 System V 进程间通信.....261	12.3.1 线程信号管理.....319
11.1 System V IPC 基础.....263	12.3.2 线程信号应用实例.....320
11.1.1 key 值和 ID 值.....263	12.4 线程属性控制.....322
11.1.2 拥有者及权限.....265	12.4.1 获取线程 ID.....323
11.2 消息队列.....265	12.4.2 初始化线程属性对象...324
11.2.1 消息队列 IPC 原理.....265	12.4.3 获取/设置线程 detachstate 属性.....325
11.2.2 Linux 消息队列管理.....267	12.4.4 获取/设置线程栈相关 属性.....326
11.2.3 消息队列应用实例.....269	第 13 章 Linux Socket 网络编程基础...328
11.3 信号量通信机制.....273	13.1 网络通信基础.....329
11.3.1 信号量 IPC 原理.....273	13.1.1 TCP/IP 协议簇基础.....329
11.3.2 Linux 信号量管理 操作.....274	13.1.2 IPv4 协议基础.....330
11.3.3 SEM_UNDO 参数的 应用.....279	13.1.3 点分十进制 IP 地址与二 进制 IP 地址转换.....333
11.3.4 使用信号量实现生产 消费问题.....282	13.1.4 网络数据包封包与拆包 过程.....335
11.4 共享内存.....285	13.1.5 字节顺序与大小端 问题.....340
11.4.1 共享内存 IPC 原理.....285	13.2 BSD Socket 网络通信编程.....344
11.4.2 Linux 共享内存管理.....286	13.2.1 BSD TCP 通信编程 流程.....344
11.4.3 共享内存的权限管理 示例.....287	13.2.2 BSD Socket 网络编程 API.....346
11.4.4 共享内存处理应用 示例.....288	13.3 使用 TCP 实现简单聊天 程序.....351
第 12 章 Linux 多线程编程.....293	13.3.1 服务器端代码分析.....352
12.1 线程基本概念与线程操作.....294	13.3.2 客户器端代码分析.....354
12.1.1 线程与进程的对比.....294	

13.4 网络调试工具	356	数据	400
13.4.1 tcpdump 的使用	356	15.5 域名与 IP 信息解析	403
13.4.2 netstat 工具使用	359	15.5.1 Linux 下域名解析	
13.4.3 lsof 工具使用	360	过程	403
第 14 章 TCP 高级应用	362	15.5.2 通过域名返回主机	
14.1 文件 I/O 方式比较	363	信息	404
14.2 I/O 阻塞与非阻塞操作	364	15.5.3 通过域名和 IP 返回主机	
14.2.1 阻塞与非阻塞基本		信息	405
概念	364	15.5.4 getaddrinfo 获取主机	
14.2.2 非阻塞应用示例	365	信息	406
14.3 socket 多路复用应用	368	第 16 章 网络服务器应用设计	410
14.3.1 select()与 pselect 函数		16.1 迭代服务器设计	411
介绍	368	16.1.1 xinetd 服务介绍	411
14.3.2 poll 与 ppoll 函数	370	16.1.2 时间服务器应用	412
14.3.3 多路复用应用示例	371	16.2 多进程/多线程并发服务器	
14.4 控制 socket 文件描述符		设计	414
属性	376	16.2.1 多进程实现多客户端	414
14.4.1 set/getsockopt()修改		16.2.2 多线程实现多客户端	418
socket 属性	376	16.2.3 基于 HTTP 的多进程	
14.4.2 fcntl 控制 socket	379	并发文件服务器	418
14.4.3 ioctl 控制文件描述符	379	16.3 进程池/线程池服务器设计	428
第 15 章 UDP 网络编程应用	383	16.3.1 进程池/线程池服务器	
15.1 UDP 网络编程基础	384	模型	428
15.1.1 UDP 网络通信流程	384	16.3.2 线程池文件服务器	
15.1.2 使用 AF_INET 实现 UDP		示例	431
点对点通信示例	385	第 17 章 本地通信与原始套接口	440
15.2 UDP 广播通信	388	17.1 sock 实现本地进程间通信	441
15.2.1 广播地址与广播通信	388	17.1.1 使用 socket 实现本地进	
15.2.2 UDP 广播通信示例	390	程通信	441
15.3 UDP 组播通信	393	17.1.2 使用 AF_UNIX 实现本	
15.3.1 组播地址与组播通信	393	机数据流	442
15.3.2 UDP 组播应用示例	394	17.2 本地 socket 传递文件描述符	445
15.4 socket 信号驱动	399	17.2.1 sendmsg/recvmmsg 函数	446
15.4.1 异步信号处理机制		17.2.2 传递文件描述符示例	446
流程	399	17.3 原始套应用程序开发	450
15.4.2 信号驱动方式处理 UDP		17.3.1 原始套接口基本原理	450



17.3.2	原始套接口实现 ping 应用程序.....	450	18.4.4	MP3 文件格式	464	
17.3.3	原始套实现 DOS 攻击	456	18.2	OSS 音频设备编程	467	
第 18 章 音频应用程序开发基础			459	18.2.1	OSS 音频设备基本 架构	467
18.1	WAV 音频文件格式分析	460	18.2.2	OSS 音频编程应用 示例	469	
18.1.1	数字音频基本参数	460	18.3	ALSA 音频设备编程	474	
18.1.2	WAV 音频文件结构	460	18.3.1	ALSA 基本架构	474	
18.1.3	读出 WAV 格式文件头 信息	463	18.3.2	alsa-libs 基本应用	476	
			18.3.3	ALSA 音频编程示例	481	

LINUX

第1章

Linux 下 C 语言开发环境

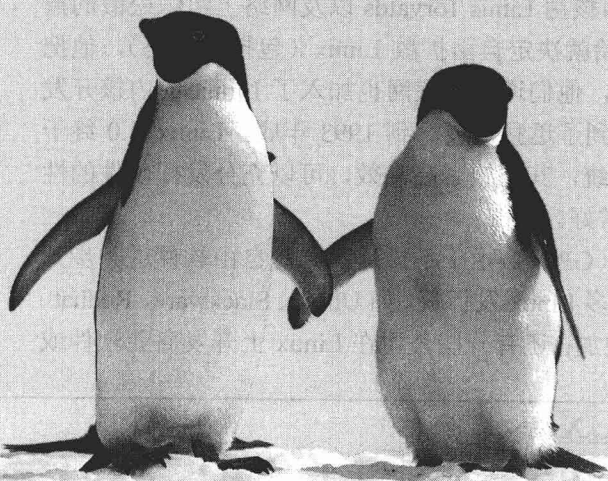
Linux 应用程序开发平台有别于 Windows 应用程序开发平台，因此在介绍具体编程内容之前，本书第 1、2 章主要介绍 Linux 操作系统下 C 语言程序的开发环境和开发工具。

本章主要介绍 Linux 下 C 语言开发环境，包括一些基本概念和基本编程环境。本章第 1 节主要对 Linux 操作系统及其相关术语进行了简要介绍。

本章第 2 节主要介绍 Linux 操作系统下编程基本概念以及如何获得 Linux 下的帮助文件，包括 Linux 操作系统下 C 语言库文件标准以及系统调用的基本概念。

本章第 3 节主要介绍 Linux 部分常用工具，包括文件打包工具、查找搜索工具，熟练使用这些命令或工具在编程时能够很好地提高效率。

本章第 4 节为读者展示了 GNU 编码规范和 Linux 内核编码规范。读者在学习 Linux 编程之前应养成良好的编码规范，这样不仅能增强代码的可读性，还能减少代码维护的工作量，提高代码的可扩展性。





1.1 Linux 操作系统简介

1.1.1 Linux 操作系统简介

UNIX 操作系统于 1969 年由 Ken Thompson 在 AT&T 贝尔实验室的一台 DEC PDP-7 计算机上实现。后来 Ken Thompson 和 Dennis Ritchie 使用 C 语言对整个系统进行了再加工和编写,使得 UNIX 能够很容易地移植到其他硬件的计算机上。由于此时 AT&T 还没有把 UNIX 作为它的正式商品,因此研究人员只是在实验室内部使用并完善它。正是由于 UNIX 是被作为研究项目,其他科研机构 and 大学的计算机研究人员也希望能得到这个系统,以便进行自己的研究。AT&T 采用分发许可证的方法,大学和研究机构仅仅需要很少的费用就能获得 UNIX 的源代码以进行研究。UNIX 的源代码被散发到各个大学,一方面使得科研人员能够根据需要改进系统,或者将其移植到其他的硬件环境中去;另一方面培养了大量懂得 UNIX 使用和编程的学生,这使 UNIX 的使用更为普及。

到了 20 世纪 70 年代末,在 UNIX 发展到版本 6 之后,AT&T 认识到了 UNIX 的价值,并成立了 UNIX 系统实验室 (UNIX System Lab, USL) 来继续发展 UNIX。因此一方面 AT&T 继续发展内部使用的 UNIX 版本 7,一方面由 USL 开发对外正式发行的 UNIX 版本,同时 AT&T 也宣布对 UNIX 产品拥有所有权。几乎在同时,加州大学伯克利分校计算机系统研究小组 (CSRG) 借助 UNIX 对操作系统进行了研究,他们对 UNIX 进行的改进相当多,增加了很多当时非常先进的特性,包括更好的内存管理、快速且健壮的文件系统等,大部分原有的源代码都被重写,很多其他的 UNIX 使用者,包括其他大学和商业机构,都希望能得到经 CSRG 改进的 UNIX 系统。因此 CSRG 的研究人员把他们的 UNIX 组成一个完整的 UNIX 系统——BSD UNIX (Berkeley Software Distribution) 向外发行。

与此同时,AT&T 的 UNIX 系统实验室也在不断改进他们的商用 UNIX 版本,直到他们吸收了 BSD UNIX 中已有的各种先进特性,并结合其本身的特点,推出了 UNIX System V 版本。从此以后,BSD UNIX 和 UNIX System V 形成了当今 UNIX 的两大主流,现代的 UNIX 版本大部分都是这两个版本的衍生产品:IBM 的 AIX4.0、HP/UX11 和 SCO 的 UNIXWare 等属于 System V,而 Minix、freeBSD、NetBSD、OpenBSD 等属于 BSD UNIX。

Linux 由 UNIX 操作系统发展而来,它的内核由 Linus Torvalds 以及网络上组织松散的黑客队伍一起从零开始编写而成。Linux 从一开始就决定自由扩散 Linux (包括源代码),他把源代码发布在网上,随即就引起爱好者的注意,他们通过互联网也加入了 Linux 的内核开发工作。一大批高水平程序员的加入使 Linux 得到了迅猛发展。到 1993 年底,Linux 1.0 终于诞生。Linux 1.0 已经是一个功能完备的操作系统,其内核紧凑高效,可以充分发挥硬件的性能,在 4MB 内存的 80386 机器上也表现得非常好。

Linux 加入 GNU 并遵循通用公共许可证 (GPL),由于不排斥商家对自由软件进一步开发,故而使 Linux 开始了又一次飞跃,出现了很多 Linux 发行版,如 Ubuntu、Slackware、Redhat、TurboLinux、OpenLinux 等十多种,而且还在增加;还有一些公司在 Linux 上开发商业软件或

把其他 UNIX 平台的软件移植到 Linux 上来。如今很多 IT 界的大腕如 IBM、Intel、Oracle、Infomix、Sysbase、Netscape、Novell 等都宣布支持 Linux。商家的加盟弥补了纯自由软件的不足，扫清了发展障碍，Linux 得以迅速普及。

Linux 操作系统具有以下特点。

- Linux 具备现代一切功能完整的 UNIX 系统所具备的全部特征，其中包括真正的多任务、虚拟内存、共享库、需求装载、优秀的内存管理以及 TCP/IP 网络支持等。
- Linux 的发行遵守 GNU 的通用公共许可证（GPL）。
- 在原代码级上兼容绝大部分的 UNIX 标准（如 IEEE POSIX，System V，BSD），遵从 POSIX 规范。读者可以在网络上获得关于这一内容的更多信息。

1.1.2 GNU/Linux 简介

GNU 工程（GNU 是“GNU's Not UNIX”首字母缩写语）开始于 1984 年，旨在发展一款类 UNIX 且为自由软件的完整操作系统：GNU 系统。更精确地说，各种使用 Linux 作为内核的 GNU 操作系统应该被称为 GNU/Linux 系统。

GNU 工程开发了大量用于 UNIX 的自由软件工具和类 UNIX 操作系统，例如 Linux。虽然有许多组织和个人都对 Linux 的发展作出了帮助，但自由软件基金会依然是最大的单个贡献者。它不仅仅创造了绝大部分在 Linux 中使用的工具，还为 Linux 的存在提供了理论和社会基础。

为保证 GNU 软件可以自由地“使用、复制、修改和发布”，所有 GNU 软件都遵循无条件授权所有权利给任何人的协议条款——GNU 通用公共许可证（GNU General Public License，GPL）。

1985 年 Richard Stallman 创立的自由软件基金会（FSF，Free Software Foundation）为 GNU 计划提供了技术、法律以及财政支持。尽管 GNU 计划大部分时候是由个人自愿无偿贡献，但 FSF 有时还是会聘请程序员帮助编写。当 GNU 计划开始逐渐获得成功时，一些商业公司开始介入开发和技术支持。

到了 1990 年，GNU 计划已经开发出的软件包括了一个功能强大的文字编辑器 Emacs、C 语言编译器 GCC 以及大部分 UNIX 系统的程序库和工具。唯一依然没有完成的重要组件就是操作系统的内核（称为 HURD）。

1.1.3 相关术语介绍

1. POSIX 及其重要地位

POSIX 表示可移植操作系统接口（Portable Operating System Interface，缩写为 POSIX 是为了读音更像 UNIX）。它由电气和电子工程师协会（Institute of Electrical and Electronics Engineers，简称为 IEEE）开发，可以提高类 UNIX 环境下应用程序的可移植性。然而，POSIX 并不局限于 UNIX，许多其他的操作系统，例如 DEC OpenVMS 和 Microsoft Windows NT，都支持 POSIX 标准，尤其是 IEEE STD.1003.1-1990（1995 年修订）和 POSIX.1。POSIX.1 给操作系统提供了源代码级别的 C 语言应用编程接口（API），例如读写文件 read/write。POSIX.1 已经被国际标准化组织（International Standards Organization，缩写为 ISO）所接受，被命名



为 ISO/IEC9945-1:1990 标准。虽然某些部分还处在开发过程中,但是 POSIX 现在已经发展成为一个庞大的标准族。

2. GNU 和 Linux 的关系

GNU 项目已经开发了许多高质量的编程工具,包括 emacs 编辑器、著名的 GNU C 和 C++ 编译器 (gcc 和 g++),这些编译器可以在任何计算机系统上运行。所有的 GNU 软件和派生工作均使用 GNU 通用公共许可证 (GPL)。GPL 允许软件作者拥有软件版权,但要授予其他任何人以合法复制、发行和修改软件的权利。

Linux 中使用了许多 GNU 工具,用于实现 POSIX.2 标准的工具几乎都是 GNU 项目开发的。Linux 内核、GNU 工具以及其他的一些自由软件组成了人们常说的 Linux,包括 C 语言编译器和其他开发工具及函数库、X Window 窗口系统、各种应用软件 (包括字处理软件、图像处理软件等)、其他各种 Internet 软件 (包括 FTP 服务器、WWW 服务器)、关系数据库管理系统等。

3. 通用公共许可证 (General Public License, GPL)

GPL 的文本保存在 Linux 系统的不同目录中的 COPYING 文件里。例如,键入 “cd/usr/doc/ghostscript*”,然后再键入 “more COPYING” 可查看 GPL 的内容。GPL 和软件是否免费无关,它的主要目标是保证软件对所有用户来说是自由的。GPL 通过如下途径实现这一目标。

(1) 要求软件以源代码的形式发布,并规定任何用户都能够以源代码的形式将软件复制或发布给其他用户。

(2) 提醒每个用户,对于该软件不提供任何形式的担保。

(3) 如果用户的软件使用了受 GPL 保护的软件的任何一部分,该软件都会成为 GPL 软件,也就是说必须随应用程序一起发布源代码。

(4) GPL 并不排斥对自由软件进行商业性质的包装和发行,也不限制在自由软件的基础上打包发行其他非自由软件。

遵照 GPL 的软件并不是可以任意传播的,这些软件通常都有正式的版权,GPL 在发布软件或者复制软件时都会声明限制条件。但是,从用户的角度考虑,这些根本不能算是限制条件,相反,用户只会从中受益,因为它可以确保用户获得源代码。

尽管 Linux 内核也属于 GPL 范畴,但 GPL 并不适用于通过系统调用而使用内核服务的应用程序,通常把这种应用程序看作是内核的正常使用。假如准备以二进制的形式发布应用程序 (像大多数商业软件那样),则必须确保自己的程序未使用 GPL 保护的任何软件。如果软件通过库函数调用而且使用了其他软件,则不必受此限制。大多数函数库受另一种 GNU 公共许可证,即 LGPL 的保护,下面将会介绍。

4. LGPL (Libraray General Public License)

GNU LGPL (GNU 程序库通用公共许可证) 的内容包括在 COPYING.LIB 文件中。如果安装了内核源程序,在任意一个源程序的目录下都可以找到 COPYING.LIB 文件的一个复制。

LGPL 允许在自己的应用程序中使用程序库,即使不公开自己的源代码。但是,LGPL 还规定,用户必须能够获得在应用程序中使用的程序库的源代码,并且允许用户对这些程序库进行修改。

大多数 Linux 程序库，包括 C 程序库（libc.a）都属于 LGPL 范畴。因此，如果在 Linux 环境下，使用 GCC 编译器建立自己的应用程序，程序所连接的多数程序库是受 LGPL 保护的。如果想以二进制的形式发布自己的应用程序，则必须注意遵循 LGPL 有关规定。

遵循 LGPL 的一种方法是，随应用程序一起发布目标代码，并发布这些目标程序和受 LGPL 保护的、更新的 Linux 程序库连接起来的 makefile 文件。

遵循 LGPL 的另一种方法是使用动态连接。使用动态连接时，即使程序在运行中调用函数库中的函数，应用程序本身和函数库也是不同的实体。通过动态连接，用户可以直接使用更新后的函数库，而不用对应用程序重新连接。

1.2 Linux 开发初步

1.2.1 Linux 下 C 程序标准

在 Linux 操作系统下进行 C 程序开发的标准主要有两个：ANSI C 标准和 POSIX 标准。

ANSI C 标准是 ANSI（美国国家标准局）于 1989 年制定的 C 语言标准，后来被 ISO（国际标准化组织）接受为标准，因此也称为 ISO C。

POSIX 标准是最初由 IEEE 开发的标准族，部分已经被 ISO 接受为国际标准。

1. ANSI C

ANSI C 的目标是为各种操作系统上的 C 程序提供可移植性保证（例如 Linux 与 Windows 之间），而不仅仅限于类 UNIX 系统。该标准不仅定义了 C 编程语言的语法和语义，而且还定义了一个标准库。ISO C 标准定义的头文件如表 1-1 所示。

表 1-1 ISO C 标准定义的头文件

头 文 件	说 明	头 文 件	说 明
<assert.h>	验证程序断言	<signal.h>	信号
<complex.h>	支持复数算术运算	<stdarg.h>	可变参数表
<ctype.h>	字符类型	<stdbool.h>	布尔类型和值
<errno.h>	出错码	<stddef.h>	标准定义
<fenv.h>	浮点环境	<stdint.h>	整型
<float.h>	浮点常量	<stdio.h>	标准 I/O 库
<inttypes.h>	整型格式转换	<stdlib.h>	实用程序库函数
<iso646.h>	替代关系操作符宏	<string.h>	字符串操作
<limits.h>	实现常量	<tgmath.h>	通用类型数学宏
<locale.h>	局部类别	<time.h>	时间和日期
<math.h>	数学常量	<wchar.h>	扩展的多字节和宽字符支持
<setjmp.h>	非局部 goto	<wctype.h>	宽字符分类和映射支持



2. POSIX 标准

POSIX.1 和 POSIX.2 分别定义了兼容操作系统的 C 语言系统接口以及工具标准。Posix 是类 UNIX 系统都遵循的标准，例如 Ubuntu 下和 RedHat 下没有代码不需要移植可直接编译后执行，而在 Linux 下基于 Posix 标准完成的代码就无法在 Windows 下直接编译并执行。表 1-2 所示为 26 项 POSIX 标准定义的头文件，表 1-3 所示为 26 项 POSIX 标准定义的 XSI 扩展头文件，表 1-4 所示为 8 项 POSIX 标准定义的可选头文件。

表 1-2 26 项 POSIX 标准定义的头文件

头 文 件	说 明	头 文 件	说 明
<dirent.h>	目录项	<arpa/inet.h>	Internet 定义
<fcntl.h>	文件控制	<net/if.h>	套接字本地接口
<fnmatch.h>	文件名匹配类型	<netinet/in.h>	Internet 地址族
<glob.h>	路径名模式匹配类型	<netinet/tcp.h>	传输控制协议定义
<grp.h>	组文件	<sys/mman.h>	内存管理声明
<netdb.h>	网络数据库操作	<sys/select.h>	select 函数
<pwd.h>	口令文件	<sys/socket.h>	套接字接口
<regex.h>	正则表达式	<sys/stat.h>	文件状态
<tar.h>	tar 归档值	<sys/times.h>	进程时间
<termios.h>	终端 I/O	<sys/types.h>	基本系统数据类型
<unistd.h>	符号常量	<sys/un.h>	UNIX 域套接字定义
<utime.h>	文件时间	<sys/utsname.h>	系统名
<wordexp.h>	字扩展类型	<sys/wait.h>	进程控制

表 1-3 26 项 POSIX 标准定义的 XSI 扩展头文件

头 文 件	说 明	头 文 件	说 明
<cpio.h>	cpio 归档值	<syslog.h>	系统出错日志记录
<dlfcn.h>	动态连接	<ucontext.h>	用户上下文
<fmtmsg.h>	消息显示结构	<ulimit.h>	用户限制
<ftw.h>	文件树漫游	<utmpx.h>	用户账户数据库
<iconv.h>	代码集转换实用程序	<sys/ipc.h>	IPC
<langinfo.h>	语言信息常量	<sys/msg.h>	消息队列
<libgen.h>	模式匹配函数定义	<sys/resource.h>	资源操作
<monetary.h>	货币类型	<sys/sem.h>	信号量
<ndbm.h>	数据库操作	<sys/shm.h>	共享存储
<nl_types.h>	消息类别	<sys/statvfs.h>	文件系统信息
<poll.h>	轮询函数	<sys/time.h>	时间类型
<search.h>	搜索表	<sys/timeb.h>	附加的日期和时间定义
<strings.h>	字符串操作	<sys/uio.h>	矢量 I/O 操作