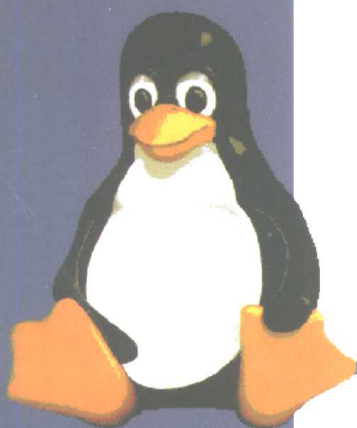




Linux 专家之路



Linux

内核源代码

范磊 编著 雨人科技 策划

人民邮电出版社
www.pptph.com.cn



附光盘
CD-ROM



Linux 专家之路



Linux

内核源代码

范磊 编著 雨人科技 策划

人民邮电出版社

图书在版编目 (CIP) 数据

Linux 内核源代码 / 范磊编著. —北京: 人民邮电出版社, 2002.1 (Linux 专家之路)
ISBN 7-115-09847-6

I. L... II. 范 III. Linux 操作系统—代码 IV. TP316.81

中国版本图书馆 CIP 数据核字 (2001) 第 096992 号

内 容 提 要

本书从操作系统的基本原理入手, 全面分析了内核数据结构之间的关系, 从而深入剖析了 Linux 操作系统的内在机制和实现方法。

本书共分 13 章, 详细地分析了 Linux 操作系统的各个功能模块, 包括系统初始化、中断、系统调用、进程管理、进程间通信模块、内存管理、文件系统、虚拟文件系统、设备驱动、网络、多对称处理等, 并且在每章后列出了精选的 Linux 内核源代码程序。

本书是一本具有很强指导性的学习参考书。本书适用于 Linux 系统管理员、Linux 系统程序员以及大专院校相关专业的师生。

Linux 专家之路 Linux 内核源代码

◆ 编 著 范 磊
策 划 雨人科技
责任编辑 张瑞喜
执行编辑 郭立罡

◆ 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 电子函件 315@pptph.com.cn
网址 <http://www.pptph.com.cn>
读者热线 010-67180876
北京汉魂图文设计有限公司制作
北京顺义向阳胶印厂印刷
新华书店总店北京发行所经销

◆ 开本: 787 × 1092 1/16
印张: 39.5
字数: 961 千字
印数: 1-4 000 册

2002 年 1 月第 1 版
2002 年 1 月北京第 1 次印刷

ISBN 7-115-09847-6/TP·2596

定价: 58.00 元 (附光盘)

本书如有印装质量问题, 请与本社联系 电话: (010)67129223

前 言

今天，Linux 正以势不可挡的趋势迅猛发展，并且其发展前景是极其广阔的。Linux 最本质的东西体现在其“自由”和“开放”的思想，“自由”意味着世界范围内的知识共享，而“开放”则意味着 Linux 对所有的人都敞开大门。在这自由而开放的天地里，你的创造激情可以得到充分的发挥。Linux 内核源代码的开放给希望深入操作系统内部世界的人提供了条件，让喜欢迎接挑战的人们可以充分检验自己的勇气和耐力。

为了同读者一同走入 Linux 这个美丽的世界，我们尽最大的努力分析 Linux 内核的内在机制。我们旨在给程序员、系统管理员、学生以及 Linux 编程爱好者提供比以前更详细和更易理解的 Linux 内核源代码分析，这对于我们开发高水平的软件将起着积极的作用。

本书分 13 章来对 Linux 内核所涉及的技术内容进行全面分析。

第 1 章简介了 Linux 的历史、开发过程及分析其内核的意义，第 2 章介绍了 Linux 内核体系结构，第 3 章讲述 Linux 操作系统的初始化技术，第 4 章讲述进程和进程调度，这是整个 Linux 操作系统的核心技术，本章给出了 Linux 对进程的描述，分析了与进程调度密切相关的时间系统，同时对 Linux 调度程序进行了较为详尽的分析。最后，对 Linux 内核独具特色的内核机制给出了具体描述。第 5 章介绍中断和中断处理。第 6 章分析了 Linux 系统调用机制。第 7 章分析 Linux 内存管理，本章主要讲述如何实现 Linux 虚拟存储管理的各种机制，这也是操作系统非常重要的部分。第 8 章介绍虚拟文件系统。第 9 章讲述 ext2 文件系统的原理。第 10 章讲述 Linux 内核进程间通信。第 11 章讲述 Linux 内核对设备的驱动程序。第 12 章讲述 Linux 网络的支持代码。第 13 章分析在多 CPU 情况下 Linux 系统的多对称处理。

本书配有光盘。光盘内容包括程序源代码、部分习题参考答案、雨人科技技术支持文档等。

本书由雨人科技策划。范磊同志负责全书的编著和修改，并负责配套光盘的制作，童鹤同志设计并制作了全书的插图，胡传威同志、周樵同志对全书进行了校对。在此对所有关心、支持和帮助过本书编写工作的同志表示诚挚的谢意！

书中不足之处敬请读者指正，以便再版时修订。

联系邮件地址：yurennet@263.net

雨人科技网站：www.yurennet.com

编著者
2001 年 12 月

目 录

第 1 章 Linux 操作系统概述	1
1.1 Linux 的历史	2
1.2 Linux 开发过程	4
1.3 Linux 的现状	4
1.3.1 RedHat 6.0 (http://www.redhat.com)	5
1.3.2 Slackware 3.4 (http://www.cdrom.com)	5
1.3.3 Debian 1.3.1 (http://www.debian.org)	5
1.3.4 SuSE Linux 5.2 (http://www.suse.com)	5
1.3.5 Turbo Linux 4.0 简体中文标准版 10CD	6
1.3.6 红旗 Linux	6
1.4 分析内核的意义	6
1.4.1 开发“自己的”操作系统	6
1.4.2 开发高水平软件	7
1.4.3 计算机科学的教学和科研	7
1.5 小结与练习	8
1.5.1 小结	8
1.5.2 习题与思考	8
第 2 章 内核体系结构概述	9
2.1 内核设计目标	10
2.1.1 清晰性	10
2.1.2 兼容性	10
2.1.3 可移植性	11
2.1.4 健壮性和安全性	12
2.1.5 速度	12
2.2 内核体系结构初识	12
2.3 内核体系结构的深入了解	14
2.4 Linux 内核的类型	15
2.4.1 层次 (Layer)	15
2.4.2 Linux 内核类型	16
2.5 了解 Linux 内核源代码	18
2.5.1 内核源代码结构	18

2.5.2	体系结构相关和体系结构无关的代码.....	23
2.6	Linux 内核源代码的部分特点.....	23
2.6.1	gcc 特性的使用	23
2.6.2	内核代码习惯用语.....	24
2.6.3	减少 #if 和 #ifdef 的使用.....	25
2.7	代码样例	25
2.7.1	printk 函数	26
2.7.2	等待队列	29
2.7.3	内核模块.....	32
2.8	配置与编译内核	34
2.8.1	配置内核	34
2.8.2	构建内核.....	35
2.8.3	备份的重要性.....	35
2.8.4	发布改进.....	36
2.8.5	多版本的内核代码.....	37
2.9	小结与练习	37
2.9.1	小结.....	37
2.9.2	习题与思考.....	37
第 3 章	系统初始化.....	39
3.1	初始化流程	40
3.1.1	系统加电或复位.....	41
3.1.2	BIOS 启动.....	41
3.1.3	Boot Loader.....	41
3.1.4	操作系统内核的初始化.....	41
3.2	初始化的任务	42
3.2.1	处理器对初始化的影响.....	42
3.2.2	其他硬件设备对初始化的影响.....	42
3.3	操作系统的初始化	42
3.3.1	引导 PC 机 (BIOS 启动和 LILO 引导)	43
3.3.2	准备 Linux 内核	44
3.3.3	初始化 Linux 内核	46
3.3.4	Bogo MIPS.....	48
3.3.5	分析内核选项.....	49
3.4	init 进程	53
3.5	本章代码	54
3.6	小结与练习	76
3.6.1	小结.....	76

3.6.2 习题与思考	76
第4章 进程	77
4.1 进程在内核中的表示方法	78
4.2 进程的状态	81
4.3 引用计数	81
4.4 优先级	81
4.5 创建进程	83
4.5.1 fork 和__clone	83
4.5.2 分配 PID	84
4.5.3 进程的运行	84
4.6 进程调度	87
4.6.1 调度策略	87
4.6.2 调度函数	89
4.6.3 计算 goodness 值	91
4.6.4 非实时优先级	92
4.6.5 实时优先级	93
4.7 遵守限制	95
4.7.1 权能	95
4.7.2 用户 ID 和组 ID	98
4.7.3 资源限制	98
4.8 进程的结束	99
4.8.1 exit 函数组	99
4.8.2 wait 函数组	100
4.9 本章代码	101
4.10 小结与练习	146
4.10.1 小结	146
4.10.2 习题与思考	146
第5章 中断和中断处理	147
5.1 硬件基础	149
5.1.1 可编程中断控制器	149
5.1.2 初始化中断处理数据结构	150
5.1.3 中断处理任务	151
5.2 中断及其有关数据结构	152
5.2.1 中断和中断请求 IRQ	152
5.2.2 数据结构	153
5.3 初始化中断请求	154

5.4	中断处理	157
5.5	本章代码	160
5.6	小结与练习	185
5.6.1	小结	185
5.6.2	习题与思考	185
第 6 章	系统调用	187
6.1	什么是系统调用	188
6.2	基本原理	189
6.3	系统调用的初始化	190
6.3.1	在处理系统调用时使用“陷阱门”的原因	190
6.3.2	利用“陷阱门”完成向高特权级（内核）切换	190
6.3.3	“陷阱门”特权切换时堆栈的变化	191
6.4	如何激活系统调用	191
6.4.1	system_call 函数	192
6.4.2	lcall7 函数	196
6.5	系统调用样例	197
6.5.1	sys_ni_syscal 函数	197
6.5.2	sys_time 函数	198
6.5.3	sys_reboot	199
6.5.4	sys_sysinfo 函数	200
6.6	本章代码	201
6.7	小结与练习	224
6.7.1	小结	224
6.7.2	习题与思考	224
第 7 章	内存管理	225
7.1	虚拟内存	226
7.1.1	虚拟内存的抽象模型	226
7.1.2	交换和分页	228
7.1.3	地址空间	229
7.1.4	内存管理单元（MMU）	229
7.1.5	页目录和页表	230
7.1.6	转换后备缓存	232
7.1.7	段	232
7.2	进程的内存组织	233
7.2.1	struct vm_area_struct	233
7.2.2	struct vm_operations_struct	234

7.2.3	struct mm_struct	235
7.2.4	VMA 的操作	235
7.3	分页	236
7.3.1	页面保护详述	236
7.3.2	写拷贝	237
7.3.3	页面错误	237
7.3.4	页面调出	242
7.4	交换设备	243
7.4.1	get_swap_page 函数	244
7.4.2	swap_free 函数	244
7.4.3	sys_swapon 函数	245
7.4.4	sys_swapon 函数	245
7.5	内存映射 mmap	247
7.5.1	do_mmap 函数	247
7.5.2	merge_segments 函数	248
7.5.3	do_munmap 函数	249
7.5.4	unmap_fixup 函数	250
7.6	用户空间和内核空间的动态内存	251
7.6.1	brk	251
7.6.2	vmalloc 和 vfree	252
7.7	主存储器信息转储	255
7.8	缓存和刷新机制	257
7.8.1	linux 使用的缓存	257
7.8.2	缓冲区高速缓存	258
7.9	内存的初始化	260
7.9.1	内存的初始化过程	260
7.9.2	进入用户模式	261
7.10	本章代码	261
7.11	小结与练习	306
7.11.1	小结	306
7.11.2	习题与思考	306
第 8 章	虚拟文件系统	307
8.1	概述	308
8.2	VFS 中的重要数据结构	309
8.2.1	VFS 的超级块	310
8.2.2	VFS 的索引节点	311
8.2.3	与进程联系的三个结构	312

8.2.4	有关操作的数据结构.....	314
8.3	高速缓存.....	317
8.3.1	块高速缓存.....	317
8.3.2	索引节点高速缓存.....	320
8.3.3	目录高速缓存.....	320
8.4	文件系统的注册、安装与卸载.....	321
8.4.1	文件系统的注册.....	321
8.4.2	文件系统的安装.....	322
8.4.3	文件系统的卸载.....	323
8.5	限额机制.....	323
8.6	文件系统的系统调用.....	325
8.6.1	open 系统调用.....	325
8.6.2	read 系统调用.....	326
8.6.3	fcntl 系统调用.....	327
8.7	本章代码.....	328
8.8	小结与练习.....	360
8.8.1	小结.....	360
8.8.2	习题与思考.....	360
第 9 章	EXT2 文件系统.....	361
9.1	基本概念.....	362
9.1.1	一组字节到逻辑块的映射.....	363
9.1.2	逻辑块到物理块的映射.....	363
9.2	EXT2 的磁盘布局和数据结构.....	364
9.2.1	EXT2 的磁盘布局.....	364
9.2.2	EXT2 超级块.....	365
9.2.3	EXT2 索引节点.....	367
9.2.4	EXT2 组描述符.....	370
9.2.5	位图.....	370
9.2.6	索引节点表及举例.....	371
9.2.7	EXT2 目录.....	372
9.2.8	在一个 EXT2 文件系统中查找一个文件.....	373
9.2.9	在 EXT2 文件系统中改变一个文件的大小.....	373
9.3	文件的访问权限和安全.....	375
9.4	连接文件.....	377
9.5	错误处理.....	377
9.6	分配策略.....	378
9.6.1	分配新块的算法.....	378

9.6.2 分配新的索引节点的算法.....	378
9.7 本章代码.....	379
9.8 小结与练习.....	410
9.8.1 小结.....	410
9.8.2 习题与思考.....	410
第 10 章 进程间通信.....	411
10.1 管道 (pipe).....	412
10.1.1 Linux 管道机制的实现.....	413
10.1.2 管道的应用.....	414
10.1.3 命名管道 (FIFO).....	415
10.2 System V IPC 机制.....	416
10.2.1 消息队列.....	416
10.2.2 信号量.....	428
10.3 共享内存.....	437
10.4 信号.....	443
10.4.1 信号的引入.....	443
10.4.2 信号掩码.....	444
10.4.3 系统调用.....	445
10.4.4 进程与信号的关系.....	446
10.5 本章代码.....	447
10.6 小结与练习.....	502
10.6.1 小结.....	502
10.6.2 习题与思考.....	502
第 11 章 设备驱动.....	503
11.1 Linux 的设备管理.....	504
11.1.1 I/O 软件.....	505
11.1.2 设备驱动程序.....	506
11.1.3 Linux 驱动程序的几个通用函数.....	508
11.2 中断.....	509
11.2.1 硬件对中断的支持.....	510
11.2.2 Linux 对中断的管理.....	511
11.2.3 Linux 对中断的处理.....	512
11.3 块设备驱动程序.....	512
11.3.1 块设备驱动程序的登记.....	513
11.3.2 块设备基于缓冲区的数据交换.....	513
11.3.3 RAM 盘驱动程序的实现.....	514

11.3.4	硬盘驱动程序的实现.....	515
11.4	字符设备驱动程序.....	517
11.4.1	字符设备的注册.....	517
11.4.2	工作内存.....	518
11.4.3	基本入口点.....	519
11.5	网络设备的初始化.....	519
11.6	本章代码.....	520
11.7	小结与练习.....	542
11.7.1	小结.....	542
11.7.2	习题与思考.....	542
第 12 章	网络.....	543
12.1	概述.....	544
12.1.1	Linux 的网络层次结构.....	544
12.1.2	面向对象的设计方法实现网络层次.....	544
12.2	网络协议.....	545
12.2.1	网络参考模型.....	545
12.2.2	TCP / IP 协议工作原理及数据流.....	546
12.2.3	Internet 协议.....	548
12.3	套接字 (socket).....	551
12.3.1	套接字在网络中的地位和作用.....	551
12.3.2	套接字接口的种类.....	552
12.3.3	套接字的工作原理.....	553
12.3.4	socket 的通信过程.....	554
12.3.5	socket 为用户提供的系统调用.....	557
12.4	套接字缓冲区 (sk_buff).....	557
12.4.1	套接字缓冲区的特点.....	557
12.4.2	套接字缓冲区操作基本原理.....	558
12.4.3	sk_buff 数据结构的核心内容.....	559
12.4.4	套接字缓冲区提供的函数.....	561
12.4.5	套接字缓冲区的上层支持例程.....	562
12.5	网络设备接口.....	563
12.5.1	基本结构.....	563
12.5.2	命名规则.....	564
12.5.3	设备注册.....	564
12.5.4	网络设备数据结构.....	565
12.5.5	支持函数.....	567
12.6	本章代码.....	570

12.7 小结与练习	578
12.7.1 小结	578
12.7.2 习题与思考	578
第 13 章 多对称处理	579
13.1 并行程序设计概念及其原语	581
13.1.1 原子操作	581
13.1.2 test-and-set 原语	583
13.1.3 信号量	584
13.1.4 自旋锁	589
13.2 APIC 和 CPU-TO-CPU 通信	591
13.3 SMP 对内核的影响	592
13.3.1 对调度的影响	592
13.3.2 smp_local_timer_interrupt 函数	595
13.3.3 lock_kernel 函数和 unlock_kernel 函数	596
13.3.4 softirq_trylock 函数	597
13.3.5 cli 宏和 sti 宏	597
13.3.6 irq_enter 函数和 irq_exit 函数	598
13.4 本章代码	599
13.5 小结与练习	606
13.5.1 小结	606
13.5.2 习题与思考	606
附录 部分习题参考答案	607

第1章

Linux 操作系统概述

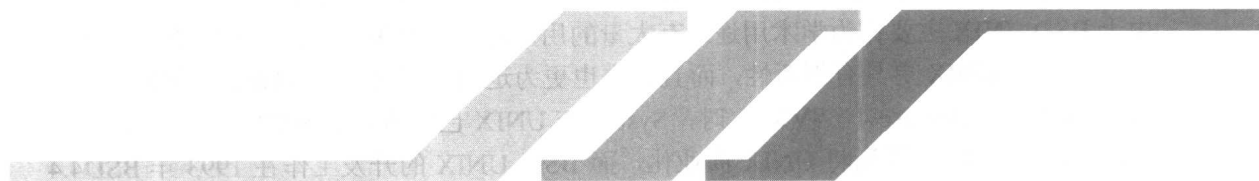
Linux 的历史

Linux 的开发过程

Linux 的现状

分析内核的意义

小结与练习



Linux 在操作系统家族中高举“自由”的旗帜，它让任何人都可以自由获取系统源代码，这是其他商业性操作系统所无法做到的，通过 Internet 的连接，世界各地的计算机爱好者迅速地接受它、并且深深地为它所着迷。很多人对 Linux 的发展倾注了极大的热情，使它迅速的成长起来。

Linux 的“自由”性使用户有机会去了解操作系统内核的实际工作方式：内核以独占的方式执行底层任务；保证系统正常运行（协调多个并发进程）；管理进程使用的内存，使它们相互之间不产生冲突；满足进程访问磁盘的请求等。

本书将用 Linux 去揭开操作系统的神秘面纱。

1.1 Linux 的历史

Linux 是在 UNIX 的基础上发展而来的，UNIX 是由 AT&T 贝尔实验室的 Ken Thompson 和 Dennis Ritchie 于 1969 年在一台已经废弃了的 PDP-7 上开发的。最初 UNIX 只是一个用汇编语言写成的单用户操作系统。Thompson 和 Ritchie 成功地说服管理部门为他们购买更新的计算机，以便该开发小组可以实现一个文本处理系统，UNIX 就是在 PDP-11 上用 C 语言重新编写的（发明 C 语言的部分目的就在于此）。不久之后，它发展成了一个文本处理系统。令人意想不到的是，他们在实现了该文本处理工具的同时，实现了一个操作系统。

该文本处理工具的实现，使 UNIX（以及 UNIX 上运行的工具）在 AT&T 中得到广泛应用。1973 年，Thompson 和 Ritchie 在一个操作系统会议上就这个系统发表了一篇论文，该论文引起了学术界对 UNIX 系统的极大兴趣。

由于 1956 年反托拉斯法案的限制，AT&T 不能涉足计算机业务，只允许它象征性地收取费用发售该系统。就这样，UNIX 被广泛发布，并从学术科研用户逐步又扩展到政府和商业用户。

伯克利加州大学是学术用户中的一个。在此，UNIX 得到了计算机系统研究小组（CSRG）的广泛应用，并且他们开发出 UNIX 的一个新系列，这就是广为人知的伯克利 UNIX（BSD）。除了 AT&T 所提供的 UNIX 系列，BSD 是最有影响力的 UNIX 系列，它在 UNIX 中增加了许多显著特性，例如 TCP/IP 网络、更好的用户文件系统（UFS）、工作控制，并且改进了 AT&T 的内存管理代码。

多年以来，BSD 版本的 UNIX 一直在学术环境中占据主导地位，但发展成为 System V 版本的 AT&T UNIX 则最终成为商业领域的领头羊。从某种程度上来说，这是有社会原因的，学校倾向于使用非正式的 BSD UNIX，而商业界则倾向于从 AT&T 获取 UNIX。

由于 BSD UNIX 主要作为学术用途，有大量的用户对它进行编程改进，所以它的风格一般要比 AT&T 的 UNIX 更具有创新性，而且改进也更为迅速。但是，在 AT&T 发布最后一个正式版本 System V Release 4（SVR4）时，System V UNIX 已经吸收了 BSD 的大多数优点。从 1984 年开始，AT&T 逐渐将 UNIX 商业化，而 BSD UNIX 的开发工作在 1993 年 BSD4.4 版本完成以后就逐渐收缩，并很快终止。BSD 的进一步改进由外界开发者保持下来，一直延

续到现在。正在进行的 UNIX 系列开发中至少有 4 个独立的版本是直接起源于 BSD 4.4。

实际上 UNIX 的变种并不止 BSD 和 System V。UNIX 主要使用 C 语言来编写,这就使得它的重新设计和移植比较容易。UNIX 的这些特点大受商业界硬件供应商的欢迎,比如 IBM 公司就多次对 UNIX 进行了再开发。厂商们设计开发出新的硬件,并简单地将 UNIX 移植到新的硬件上,为了占有市场,这些版本故意以不同的侧重点发布出来,以便更好地吸引用户。

版本混乱的状态促进了标准化工作的进行,其中最主要的就是 POSIX 系列标准,它定义了一套标准的操作系统接口和工具。从理论上说,POSIX 标准代码很容易移植到任何遵守 POSIX 标准的操作系统中,而且严格的 POSIX 测试已经把这种理论上的可移植性转化为现实。直到今天,几乎所有正式的系统都以支持 POSIX 标准为目标。

1984 年, Richard Stallman 独立开发出一个类 UNIX 的操作系统,该操作系统具有完全的内核、开发工具和终端用户应用程序。在 GNU (“GNU”与“Not UNIX”首字母的缩写相似)计划的启发下, Stallman 对开发这个产品有了自己的技术理想,开发出一个质量高而且自由的操作系统。Stallman 使用了“自由”(Free)这个词,这意味着用户可以免费获取软件,而且更重要的是,可以自由使用、拷贝、查询、重用、修改甚至是分发这份软件,完全没有软件使用协议的限制。这也正是 Stallman 创建自由软件基金会(FSF)资助 GNU 软件开发的本意(FSF 也在资助其他科研方面的开发工作)。

15 年来, GNU 工程已经吸收、产生了大量的程序,包括 Emacs、gcc (GNU 的 C 编译器)、bash (shell 命令),和其他 Linux 用户所熟知的应用程序。现在正在进行开发的项目是 GNU Hurd 内核,这是 GNU 操作系统的最后一个主要部件(实际上 Hurd 内核早已能够使用了,不过当前的版本号为 0.3 的系统在什么时候能够完成,还是未知数)。

在 GNU 发展的中期(1991 年 8 月),芬兰的一个学生在 comp.os.minix 新闻组贴上了以下这段话:“你好!所有使用 minix 的人,我正在为 386 (486) AT 做一个免费的操作系统(只是为了爱好,不会像 gun 那样很大很专业)。”这名学生就是 Linus Torvalds,他对当时 UNIX 变种版本不能对 80386 类计算机很好地支持十分不满,他决定要开发出一个全功能的、支持 POSIX 标准的、类 UNIX 的操作系统内核。Linus (虽然他原名为 Torvalds,但是所有人都称他为 Linus)独立把这个内核开发到 0.02 版,并且可以在这个版本上运行 gcc、bash 和少量的应用程序,这些就是他开始的全部工作了。后来,他又开始在因特网上寻求广泛的帮助。作为一个具备所有特性的类似 POSIX 的操作系统, Linux 并非仅由 Linus 一人开发,而是由全世界几百个程序员共同开发的。虽然 Linux 内核开发没有一个管理组织,但是 Linus 已经为内核开发定了一个基调。一名俄罗斯学生得到一块新的主板,就可以写一个新的驱动程序支持这块主板,菲律宾的一个系统管理员需要备份软件,就编写一份新的文件管理程序,并把它传给任何需要的人。

不到三年, Linus 的 UNIX——Linux,已经升级到 1.0 版本。它的源代码量也呈指数形式增长,而且实现了基本的 TCP/IP 功能(网络部分的代码后来重写过,而且还可能会再次重写)。此时 Linux 已经拥有大约 10 万用户了。

现在的 Linux 内核由 150 多万行代码组成, Linux 也已经拥有了大约 1000 万用户(由于 Linux 可以自由获取和拷贝,所以具体的统计数字是不可能的)。Linux 内核 GNU/Linux 附同

GNU 工具已经占据了 UNIX 50% 的市场，一些公司正在把内核、应用程序同安装软件打包在一起，生产出 Linux 的发行版本，这些公司包括 Red Hat 和 Caldera。现在的 GNU/Linux 已经备受瞩目，得到了诸如 Sun、IBM、SGI 等公司的广泛支持，SGI 最近决定在基于 Intel 公司的 Merced 系列计算机上不再搭载自己的 UNIX 变种版本 IRIX，而是直接采用 GNU/Linux，Linux 甚至被指定为 Amiga 公司将要发布的新操作系统的基础。

1.2 Linux 的开发过程

实际上，并没有一个专门的、独立的组织负责开发 Linux 系统，它是由世界各地的自愿者通过 Internet 共同开发的，Linux 团体大部分通过邮递列表和 USENET 的消息组进行通信，这样任何人都有机会辅助开发和调试 Linux 的内核、连接新的软件、编写文档和帮助新用户。

Linux 最强大的生命力正是在于这种公开的开发过程。UNIX 是一个简单且非常优秀的模型，在 Linux 创建之前，UNIX 已经有 20 年的发展历史。Linux 从 UNIX 的各个流派中不断吸取成功经验，抛弃 UNIX 的缺点，使 Linux 成为了 UNIX 系列中的佼佼者。开发商业的 UNIX 是一个很大的工程，并且为了保证下次操作系统的升级性能，通常有数以百计的程序员、测试员、文档员和系统管理员参与，所以商业 UNIX 那种开发过程使得它的源代码非常复杂。但是，Linux 就完全不同，每个人都可以通过 Internet 自由获取 Linux 内核源程序，每个人都可以对源程序加以修改。在使用中如果发现了缺陷或还有什么新的创意，都可以加以改进，用户也可以直接在系统中增加功能，而不用向操作系统供应商解释自己的想法。这在源程序不公开的操作系统中是很难办到的。

Linux 开发团体大部分通过邮递列表 USENET 的消息组通信。许多协定已经跳过开发过程，如果一个 Linux 爱好者想将自己的代码添加到“正式”内核中，只需给 Linus Torvalds 发一个 E-mail，他就会进行测试，如果测试通过，代码就会被包括进内核。

1.3 Linux 的现状

Linux 的本质只是个 Kernel（内核）而已，一些公司和爱好者在不断地开发和推出新 kernel，而相应的应用工具和软件则大部分采用 GNU 的和其他的自由软件。由于 Linux 基本遵循 POSIX 标准，所以大多数 UNIX 软件的编译移植不会有太大的困难（UNIX 可移植性相当好，同一套源代码在小至微机大至巨型机上都可编译）。将 Linux kernel 和这些外围的程序整合起来的任务是由 Linux 发行商完成的，他们不断推出不同的发行版本。

常见的 Linux 发行版本如下。