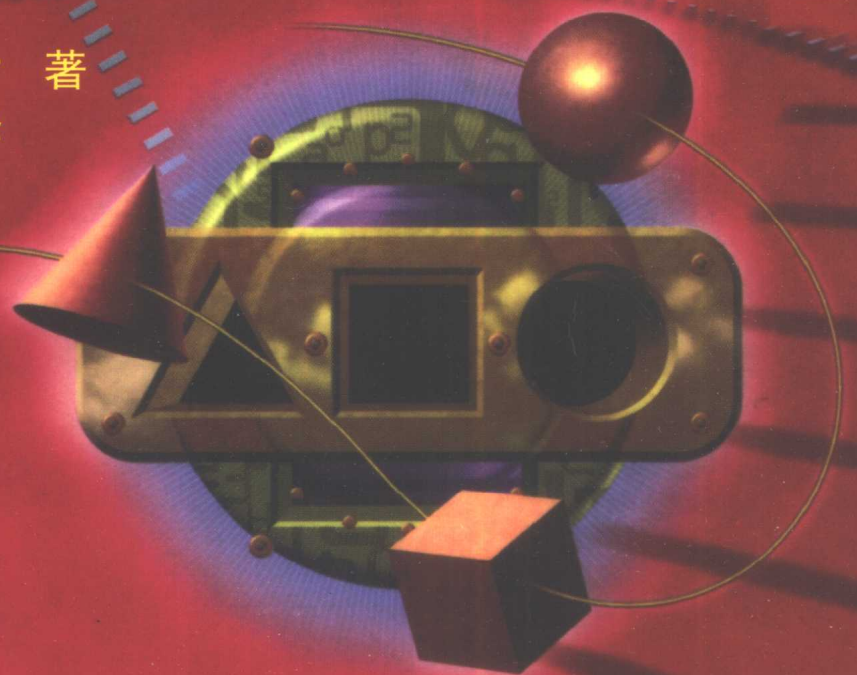
北京科海培训中心

Linux 技术内幕

[美] Moshe Bar 著
王超 译



CD-ROM

清华大学出版社
http://www.tup.tsinghua.edu.cn

McGraw-Hill 北京科海培训中心

Linux 技术内幕

[美] Moshe Bar 著

王 超 译

清华大学出版社

(京)新登字 158 号

著作权合同登记号:01-2001-3278

内 容 简 介

这是一本综合性地介绍有关 Linux 内核代码的功能和操作系统理论的高级指南。

全书由公认的 Linux 专家编写,书中介绍了 Linux(包括最新的 2.4 版本)内核的理论,Linux 设计原理和进程模型以及专家级的建议,并介绍了一些深入的课题,包括影响系统性能的因素;内核优化的范例;不同的 Linux 版本之间的差异;避免繁忙服务器崩溃的策略。

本书针对那些需要计划及实现功能强大的基于 Linux 解决方案的富有经验的 Linux 用户。对繁忙的专业人员来说,本书是目前最具眼光和最重要的参考指南。

书　名:Linux 技术内幕

作　者:Moshe Bar

译　者:王超

出版者:清华大学出版社(北京清华大学校内,邮编 100084)

印刷者:北京门头沟胶印厂

发行者:新华书店总店北京科技发行所

开　本:787×1092 1/16　印张:17.5　字数:425 千字

版　次:2001 年 9 月第 1 版　　2001 年 9 月第 1 次印刷

印　数:0001～5000

盘　号:ISBN 7-900637-24-9

定　价:39.00 元(光盘)

前　言

在最近两年，Linux 内核以令人惊异的速度快速发展：首先是 2.0 版本，然后是 2.2 版本，并且很快将出现 2.4 版本。现在，无论是对服务器还是对工作站，Linux 都已经发展成为最先进的操作系统之一。

特别是即将发布的 2.4 版本，我们的开发人员做了大量的努力，使它的内核高度可扩展，并能够有效地处理和管理大量存储器、并行的 CPU 和大量现代外设。

如果在几年前还有人能够通过读内核源代码来理解主要的概念，在今天想这样做已经变得非常困难了。不仅是因为源代码大幅度增加，而且由于内核开发人员自己也学到了新技巧和新技术，从而使算法和源代码变得越来越复杂和难于理解。

Moshe Bar 使 Linux 的 2.3.xx 版本的内核和 2.4 版本的内核更易于为 Linux 爱好者、系统管理员和系统程序员所理解，他的工作在内核开发人员和用户之间架起了一座桥梁。

在本书的写作过程中，Moshe 和我保持着密切的联系，因此，我不得不面对并回答他提出的许多问题，直到他完全理解了算法或功能才接受我给出的答案。

作为一位内核开发人员，我认为本书不仅详细讲述了 Linux 内核代码的功能，而且也清晰地阐述了 Linux 的内核概念，因此，本书既可作为引导读者探索 Linux 内核的教材，也可作为查找 Linux 内核概念的参考书。

这些年，Moshe Bar 写了大量有关操作系统原理方面的文章和专题论文，他还参与了许多重要的操作系统项目，比如 MVS、VM 和 BSD。这些都体现了他不仅拥有丰富、深厚的技术知识，而且也拥有清晰的技术表达能力。

Ingo Molnar

导 论

编写《Linux 技术内幕》一书的目的是帮助读者理解 Linux 的内部工作机制。而这些“技术内幕”大多数存在于 Linux 操作系统的内核部分。然而，本书并不是一般意义上的对源代码的完全注解，当然，其中也解释了一些代码，但本书的重点在于介绍代码的功能，而不是代码本身。

Linux 内核包括控制系统资源的代码、用户进程、数据输入/输出和扩展功能，例如 Linux 内核空间 Web 服务器 kHTTPd。

Linux 内核

内核是运行在保护模式下并能对硬件的特权寄存器进行访问的操作系统软件。许多操作系统使用微内核体系结构，在这种体系结构下，设备驱动程序和其他代码只在需要时装入和执行，并不常驻内存。

内核不是运行在系统上的独立进程，它是操作系统的内部核心。内核控制进程的调度以获得多任务功能，并且内核提供一系列任何用户空间的进程都能够访问的例程，这些例程常驻内存。

相反，单片机体系结构在 UNIX 的实现中更普遍，它是为实现经典的操作系统（比如 BSD）所设计的，在这种方式下，所有的设备驱动程序都是内核固有的一部分。Linux 内核几乎就是单片机体系结构的——所有的设备驱动程序都是内核固有的一部分，但不同于 BSD 的是，Linux 内核的设备驱动程序是可装载的，也就是说，可以根据用户的命令装入内存或从内存卸载。

本书主要参考 Intel x86 体系结构的 2.4.x 内核系列（本书写作时，仍称 2.3.51）。本书的所有代码例子都基于 2.3.49 内核，偶尔用一下 2.5 内核系列。

本书的读者对象

《Linux 技术内幕》的读者对象为系统和网络管理员、开发人员和能力计划管理人员。当然，本书对那些具有一定的软硬件知识的 Linux 爱好者也有吸引力。

通过本书，系统管理员将学到如何为特定的需求和硬件环境调整内核。此外，通过将内核调节到适当的状态，系统的吞吐量将显著增加。本书第 10 章“文件系统”将讨论怎样选择、安装和正确配置文件系统。Linux 开发人员将学到哪些编程技巧会使他们的程序产生最好的性能，并且他们的调试将会更容易，因为读完本书，他们将会更好地理解影响用户程序的外部因素。

最后，一般的 Linux 爱好者通过本书将发现内核技术设计方面隐藏的精华，使他们学到

如何进一步鉴赏他们喜爱的内核。

阅读本书前的预备知识

虽然预先具有内核方面的知识是不必要的，但这有助于对软硬件基本概念的理解。此外，如果你具有 Linux 接口方面的知识并对基本系统管理涉及到的工作有个初步的了解将对你阅读本书有所帮助。

尽管本书提供的代码都作了解释，但能够阅读 C 程序还是很有用的。关于 C 语言好的入门书，我一直强烈推荐由 C 语言的设计者 Kernighan/Ritchie 所著的《C 程序设计语言》。

本书包含的内容

本书首先介绍一般的操作系统理论和内核概念以及 UNIX 的相关细节。

接着各章介绍任一内核都有的五大支柱：

- 进程
- 进程间通信
- 信号(中断)处理
- 虚拟内存
- 文件系统

每章都以相同的方式组织：首先详细解释内核的相关主题的内部工作机制；然后提供一些代码例子来说明如何改变内核的功能；最后为了方便参考，列出了相关的内核数据结构(C 包含文件)。再加上每章主体所列的算法，读者就拥了全部的代码和数据，还有必要的解释。我们没有对 Linux 内核源代码提供更多的注释，而是将重点放到了更重要的主题及与内核性能有关的问题上。本书最后的附录包括系统调用、GPL 许可证和其他有用的信息。

你可以按章节顺序阅读本书来学习 Linux 内核知识，你也可以把本书作为参考书，选择你感兴趣的章节阅读。任何时候出现在文本中的代码段，都能在该章的最后找到相应的数据结构。

到哪里找更多的信息

有关 Linux 内核的最新信息能够在 Internet 上找到。然而，本质上都是原始数据。收集、汇编这些信息并组织成连贯的格式并不是一件轻松的工作。这正是《Linux 技术内幕》的最大价值——本书提供了连贯的、经过研究的、格式化的信息源。这些信息本来都是通过分散的站点、源代码和文章提供给公众的。

当然，内核的好的信息源就是内核源代码本身。站点 http://lxr.linux.no/source/提供了一个优秀的源代码浏览器可用于所有可用的体系结构和 Linux 版本。

在站点 http://www.linuxhq.com 你将找到所有内核版本的信息，特别是当前正在开发中的版本的信息。关于 GNU 工具的完整列表，可访问 www.gnu.org。

最后，紧跟内核开发者的邮件列表可以得到大量有用信息，尽管该邮件列表的量太大，而且其信噪比小得可怜，这使得及时地提取任何有用信息都变得很困难。

建议及评论

欢迎任何建议或评论，请寄至 moshe_bar@hotmail.com 或者 Moshe Bar，c/o McGraw-Hill，Professional Book Group，Two Penn Plaza，New York，NY 10121。

鸣 谢

首先，感谢上帝赐予我所需要的一切知识来写作本书。

如果没有 Linux 内核开发界的众多出众的开发者的帮助，这本书是完不成的。Ingo Molnar，Andrea Arcangeli，Stephen Tweedie，Hans Reiser 和 Steve Best 等人都对此书提供了极大的帮助。

Joe Pranevich 的文章《The Wonderful World of 2.4 Kernels》，教我如何概括性地描述 2.4内核所做的新进展。谢谢 Joe。

Matthew Dillon 所写的 FreeBSD 描述一般虚拟内存概念的论文给了我描述 Linux 的通用 VM 概念的灵感。谢谢 Matt。

许多可敬的人都关注了本书的创作和完成：Patricia Campbell，Scott Fallin，Tom Syroid，Philip Courier 和我妻子 Patricia Lambert，还有 Brad Dixon，他们都自愿地将他们很少的业余时间贡献到本书的编辑中来。Patricia Campbell 和 Brad Dixie 的技术专长使他们在打印之前指出了许多令我尴尬的错误。我在 gpl′d Kha0s OS 的开发中的长期合作伙伴——Scott Fallin 克服了许多困难来理解文本部分并帮助我使之更好。幽默的 Philip Courier 和 Patricia Lambert 在很大程度上改善了文章的风格和组成。作家朋友 Tom Syroid 利用他的写作经验帮助我使本书的出版成为现实，我非常感激他的帮助。

在本书的编辑过程中，使用了一个邮件列表（显然是基于 Linux 的），在编辑期间讨论诸如 C 编码风格、术语的正确定义等主题。这是一招妙棋，它对团队的士气和本书的质量都有巨大帮助。谢谢你们！

我的精神伙伴 Robo-chan 在我情绪低落时给我鼓励，使我士气大振。谢谢 Roberto。

同时，感谢 TopTier Israel 和 Baan 工作组，感谢他们在我写作本书时给我一个系统管理咨询的工作，感谢他们向我提供机器以测试我所写的程序。

配套光盘内容

1. 最新的 Linux 内核，包括 2.3.51 和 2.3.99-pre4 的文件修改。
2. 内核的核心修改。
3. 各种网络驱动程序的修改。

光盘的使用

为了使用本光盘所带的内核(2.3.99-pre5，2.3.51，2.3.49(含 LVM)以及 2.3.40)，操作步骤如下：

1. 将它们从光盘拷贝到合适的工作目录，如/home/moshe/kernels/。
2. 选择一个内核，如 2.3.99-pre5，然后使用下面命令解压：

```
tar xvf linux-2.3.99-pre5.tar
```

其内容将被解压到一个子目录 linux/下。将工作目录改到那里并使用本书第 2 章“编译内核”中的指令生成内核。

目　录

第 1 章　开放源代码——实现一个现代操作系统

毫无疑问，Linux 异常成功的主要原因在于它的通用公共许可证(General Public License，GPL)。开放源代码的概念(有时候被称为自由软件)实际上很久的事了。自由软件的第一位拥护者是自由软件基金会(Free Software Foundation，FSF)的 Richard Stallman。他为他所写的许多优秀的现在仍广泛传播的软件设计了 GPL。他所写的软件中最为卓越的是 Emacs 编辑环境和为 C 和 C++所写的 GCC 编译器。Richard Stallman 现在长期做一个完全为 GNU 的操作系统的工作——一个叫 GNU Hurd 的项目。虽然开发了近十年，这个操作系统仍然没有实现，但该项目中所涌现出的许多强大的技术都被应用到其他操作系统中，比如 Linux、BSD 和其他操作系统。

操作系统在两方面受益于开放源代码方法的开发：可靠性和性能。Linux 的可靠性很大部分来自成百上千位开发者的检查，他们审计、改进、修改和测试代码。正如 Eric Raymond 在他的那篇著名的论文《The Cathedral and Bazaar》中所说："假如有足够多的测试员和开发者，几乎任何问题都能很快暴露出来并得到很好的解决。

同样，Linux 的性能受益于相同的原因。公众对代码的检查致使内核执行路径的全部效能得到持续的改进。此外，许多学术机构采用 Linux 操作系统用于研究，因此，能够使用诸如硬件概貌和虚拟机跟踪器的高级技巧来发现代码中迄今为止仍隐藏的性能瓶颈。

最后，通过将 Linux 内核移植到各种各样的处理器平台，其某些部分必须调整或重写。通过这样的比较和调整，出现了好的代码并很快移植到其他平台的后端。

有如此多的眼睛盯着 Linux 代码，就形成了很好的质保部门(QA)，这是任何封闭模式软件开发组织所不可能提供的。进而产生了高质量的软件。

即使最纯粹的开发模式(比如开放源代码模式)其本身在不能代替适宜的设计和编码方法。在设计和编码方法方面，和所有权模式相比，开放源代码模式再一次大大胜出。在 Linux 上要考虑内核改进的例子。因为只要有黑客的存在，就必须注重完美的设计、适宜的数据结构设计和性能相关的问题。甚至在最隐藏的代码段中，你有时也会发现一些漂亮的高度技巧的完美有效的设计。

1.1　Linux 的历史

正如大多数成功的项目一样，Linux 一开始也是由于实际的需要而开始于 1991 年在芬兰建立的项目。当时，赫尔辛基大学的学生 Linus B. Torvalds 自己买了一台基于 i386 的 PC，i386 是能够在芯片上支持虚拟内存管理的第一代 Intel CPU。由于不是非常满意 MS/DOS 操作系统，Torvalds 决定在他的 PC 上实现替代的 Minix 操作系统(需要增强 Minix 的性能)。

在 20 世纪 70 年代，AT&T 意识到他们的 UNIX 操作系统将成为很有价值的商业产品，于是他们决定在第 7 版中不再发布 UNIX 的源代码，在以前他们是把源代码和软件一块发布的。结果，许多大学不得不取消 UNIX 的研究转而研究理论。

不幸的是，只学理论学生们得不到信心，而这信心只有将概念实际应用到实际的操作系统上才能获得。因此，Andrew Tanenbaum 决定开发一个包含 UNIX 所有概念和功能的操作系统，并且要编写全部新的源代码。这样他就可以避免 AT&T 的严格地许可证限制而仍然可以教学生学习 UNIX。Minix 最初设计运行在 PC 机上，并基于 AT&T UNIX 版本 7 的功能集。然而，Tanenbaum 很快大大加强了 Minix 使之包含了现代操作系统所有的功能和特性。今天 Minix 仍然用作教学工具，并且被一些站点用作产品系统。若要全面了解 Minix，请学习 Andrew S. Tanenbaum 和 Albert S. Woodhull 的课堂讲义(Prentice Hall 出版的《操作系统》)。

Torvalds 很快增强了 Minix 以提供学习所需的功能和特性。他觉得 Minix 是一个学术性的操作系统，因此，他开始从头创建一个操作系统。Torvalds 最重要的决定是将他的新的操作系统源代码以 Linux 的名字在 Internet 上供自由使用(也就是开放源代码)，Linux 的名字是 Linus 和 UNIX 的缩写。

1991 年 8 月 Linux 的第一版(0.01 版)可以从 Internet 上下载。同年 10 月，Torvalds 正式声明 0.02 版可以使用。0.02 版已经能够执行 UNIX 用户登录程序(使用 bash shell，GNU gcc 编译器和其他基本应用程序)，但是能够执行的程序数量不多。

因为该项目的开放源代码本质，而且源代码全球立即可以得到，许多黑客、计算机迷和 PC 爱好者很快就开始查看代码并改进它。许多人开始给 Torvalds 发建议，Torvalds 成为 Linux 源代码开发系统的“公认”基准。对于他们的编程建议，Torvalds 拒绝了许多，但他也将一些建议溶入了 Linux。这样的开发持续了三年多，直到 Linux 的第一个正式版本发布。

在 1992 年和 1993 年，Linux 的第一版发布了。发行版包含了操作系统的全部功能，包括 Linux 内核，X 窗口系统和广泛的数百个的应用程序包。发行版还包括了一个安装器，用于准备操作系统的二进制图像和引导/关机脚本，并确保所有组件能相互兼容。最后但并不是最近，发行版还提供文档。今天还有许多版本存在，最成功的有 RedHat，SuSE 和 Caldera。

1994 年 3 月，Linux 的 1.0 版本可以获得。我清楚地记得，仍然会偶尔出现不稳定的奇怪现象，但已经很好用了。Linux 1.0 提供了 TCP/IP、SLIP 和打印机支持，并有足够的驱动程序来支持广泛的 PC 设备。然后，Linux 的兴旺就真正开始了，世界各个角落的数百万的爱好者开始使用该系统。

Linux 稳定发展到今天，已经成为 IT 市场的主要动力，Linux 发展的几个阶段如表 1.1 所示。

表 1.1 Linux 开发的时间表

1991.8	0.01 版
1991.10	正式声明 0.02 版
1993.11	内核 0.99 的 Slackware 第一个发布版
1994.3	1.0 版

（续表）

1995.6	第一次移植到 Alpha 体系结构
1996.10	Debian Linux 用于在轨道上飞行的宇宙飞船(用 IBM 膝上型电脑上)
1999.1	2.2.0 版
2000.3	2.3.49 版(前 2.4.0)

1.2　Linux 功能

Linux 从 1991 年开始经历了很长一段路。随着公开源代码的现象传播越来越广泛，有越来越多的人对 Linux 内核和其子系统做贡献。同时，成千上万的用户包以不断增长的频率加入到 Linux 中来。随着 WWW 的同步增长，一些 Web 站点完全用于每日声明 Linux 的新的软件版本。

Linux 可用于许多平台，包括 Intel、Sparc、Alpha、MIPS、Motorola 68000 系列和 Power PC。早期，Linux 遵从 POSIX 标准，允许在 Linux 下开发的应用程序非常容易地被移植到其他遵从 POSIX 标准的操作系统中。在基于 Intel CPU 的系统上，Linux 二进制遵从 iBSCS 标准，这允许静态连接程序可以不用重编译就能在 FreeBSD 或 Solaris 下运行。

从技术上讲，Linux 提供了下列功能：

- 多用户环境
- 多进程、多处理器(SMP)环境
- 进程间通信(IPC，管道，套接字)
- 进程控制
- POSIX 风格的终端管理
- 支持 TCP/IP、Ipv4、Ipv6
- 支持各种硬件
- 用可选的 LRU(Least Recently Used，最近最少使用)算法的请求调页技术和页面着色
- 交换
- 缓冲高速缓存
- 动态和共享库
- 支持许多文件系统(ext2fs，UFS，NTFS，HPFS，MS/DOS，ISO9660，coda 和许多其他的系统)

1.3　Linux 2.4 内核的新特性

Linux 2.2 主要是在 Linux 2.0 和 Linux 1.x 系列上进行改进，它支持许多新的文件系统，支持文件缓存的新系统，并且更易于扩展。Linux 2.4 就建立在这些改进上，此外，在许多

方面 Linux 2.4 都是最好的 Linux 内核。

Linux 内核是模块组件和子系统的集合，包括设备驱动程序、协议和其他类型的组件。这些都是通过应用程序接口(API)连接到 Linux 内核的核心上，API 提供了标准方法来扩展 Linux 内核。本书的大部分集中讨论下述 Linux 操作系统的组件，这些组件做大多数的工作。

- 在 2.4 内核系列中，Linux 巨大改进了处理大量进程和任务的能力。任务数的限制现在超过了 4090。调度表的效能也有了一些改进，和它的以前版本相比，Linux 2.4 能够更好地处理有大量并发进程的系统。
- 和以前的 Linux 内核版本相比，Linux 2.4 也能够处理更大的“企业级”硬件。例如，加上合适的补丁，Linux 2.4 能够对超过 4GB 的 RAM 寻址(我曾经见过一个机器配置 12GB 的 RAM，寻址空间达 8.3GB)。
- SMP(Symmetric Multi Processor，对称多处理器)系统的扩展性现在比得上其他有所有权的操作系统，某些情况下甚至更好。
- 同样，也增加了支持其他处理器平台的功能。除了前述的传统体系结构，Linux 2.4 现在还直接支持新的 Transmeta Crusoe 处理器。3Com PalmPilot 和 Psion5 都能够正常运行 Linux。新的 Intel IA64(Intel x86 体系结构的下一代，64 位的继任者)可能在 2000 年出品，但没有直接包括内核。
- 增加了许多新的文件系统的支持(Irix efs 文件系统和用于 DVD 盘的 UDF 标准)，同时，对某些文件系统(QNX 和 ext1)的支持已经过时。
- 引入了 tasklets 技术(处理低层中断的创新性方法)，使 TCP/IP 堆栈(和其他子系统及用户程序一起)更有效能[①]。新的网络协议软件能够处理 DECNet 标准。
- Linux 2.2 和 Linux 2.0 都内嵌了下述功能：不管什么时候 Java 应用程序执行时都会自动启动 Java 解释器(如果有的话)(Linux 是最早的在内核级这样做的操作系统之一)。Linux 2.4 仍然支持在必要时装入 Java 解释器的功能，但是特定的 Java 驱动程序被删除了，用户必须更新他们的配置来使用“Misc.”驱动程序。
- Linux 2.4 内核包含有一个内核 web 守护进程，即 kHTTPd。这个功能允许对更多有效静态 Web 页面的处理。

在后续各章，我们将详细探讨 Linux 内核的这些功能和许多其他特性。

① 在第 3 章介绍 tasklet 如何工作。