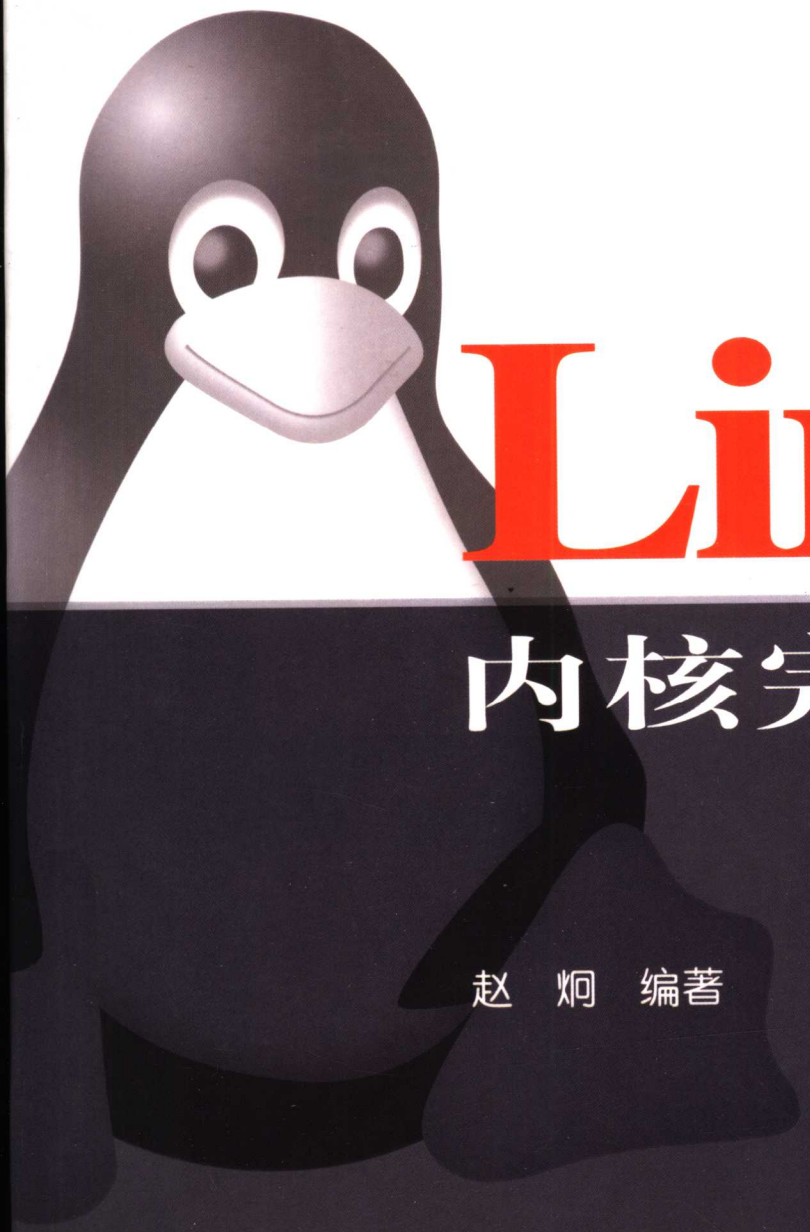


Linux

内核完全剖析

赵 炯 编著

机械工业出版社
CHINA MACHINE PRESS

Linux 内核完全剖析

赵 炯 编著

机 械 工 业 出 版 社

本书对早期 Linux 操作系统内核(v0.11)全部代码文件进行了详细的剖析，旨在让读者在尽量短的时间内对 Linux 的工作机理获得全面而深刻的理解，为进一步学习和研究 Linux 系统打下坚实的基础。虽然选择的版本较低，但该内核已能够正常编译运行，并且其中已包括了 Linux 工作原理的精髓。书中首先以 Linux 源代码版本的变迁为主线，简要介绍了 Linux 系统的发展历史，同时着重说明了各个内核版本之间的主要区别和改进方面，给出了选择 0.11 版内核源代码作为研究对象的原因。然后概要介绍了 PC 机的硬件组成结构、编制内核使用的汇编语言和 C 语言扩展部分，并且重点说明了 80x86 处理器在保护模式下运行的编程方法。接着详细介绍了 Linux 内核源代码目录树组织结构，并依据该组织结构对所有内核程序和文件进行了注释和详细说明。为了加深读者对内核工作原理的理解，书中最后一章给出了围绕 Linux 0.11 系统的多个试验。试验中使用的相关程序均可从本书配套网站（www.oldlinux.org）上下载。

本书适合作为高校计算机专业学生学习操作系统课程的辅助和实践教材，也适合 Linux 爱好者作为学习内核工作原理的自学参考书籍，还可供一般技术人员作为开发嵌入式系统的参考书使用。

图书在版编目(CIP)数据

Linux 内核完全剖析/赵炯编著. —北京：机械工业出版社，2006.1
ISBN 7-111-18032-1

Ⅰ. L... Ⅱ. 赵... Ⅲ. Linux 操作系统
Ⅳ. TP316.89

中国版本图书馆 CIP 数据核字（2005）第 145021 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
策　　划：胡毓坚
责任编辑：车　忱
责任印制：石　冉

北京中兴印刷有限公司印刷

2006 年 1 月第 1 版·第 1 次印刷
787mm×1092mm 1/16·56 印张·1466 千字
0 001—3 000 册
定价：79.00 元

序

本书是一本有关 Linux 操作系统内核基本工作原理的入门读物。

本书的主要目标

本书的主要目标是用尽量少的篇幅，对完整的 Linux 内核源代码进行解剖，使读者对操作系统的基本功能和实际实现方式获得全面的理解。

本书读者应是一些知晓 Linux 系统的一般使用方法或具有一定的编程基础，但比较缺乏阅读目前最新内核源代码的基础知识，又急切希望能够进一步理解 UNIX 类操作系统内核工作原理和实际代码实现的爱好者。目前，这部分读者人数在 Linux 爱好者中所占的比例是很高的，而面向这部分读者以比较易懂和有效的手段讲解内核的书籍资料不多。

现有书籍不足之处

目前已有的描述 Linux 内核的书籍，均尽量选用最新 Linux 内核版本（例如 Redhat Linux 7.0 使用的 2.2.16 稳定版等）进行描述，但由于目前 Linux 内核整个源代码已经非常大（例如 2.2.20 版具有 268 万行代码！），因此这些书籍仅能对 Linux 内核源代码进行选择性或原理性的说明，许多系统实现细节被忽略。所以并不能使读者对实际 Linux 内核有清晰而完整的理解。

Scott Maxwell 的《Linux 内核源代码分析》基本上是面对 Linux 中级水平的读者，需要较为全面的基础知识才能完全理解。而且可能是由于篇幅所限，该书并没有对所有 Linux 内核代码进行注释，略去了很多内核实现细节，例如内核中使用的各个头文件（*.h）、生成内核代码映像文件的工具程序、各个 make 文件的作用和实现等均没有涉及。因此对于处于初中级水平之间的读者来说，阅读该书有些困难。

John Lions 著的《莱昂氏 UNIX 源代码分析》一书虽然是学习 UNIX 类操作系统内核源代码很好的书籍，但是由于其采用的是 UNIX V6 版，其中系统调用等部分代码是用早已废弃的 PDP－11 系列机的汇编语言编制的，因此在阅读和理解与硬件部分相关的源代码时就会遇到较大的困难。

A. S. Tanenbaum 的书《操作系统：设计与实现》是一本有关操作系统内核实现很好的入门书籍，但该书所叙述的 MINIX 系统是一种基于消息传递的内核实现机制，与 Linux 内核的实现有所区别。因此在学习该书之后，并不能很顺利地即刻着手进一步学习较新的 Linux 内核源代码实现。

国内出版的部分 Linux 源代码分析的图书也有类似缺点。

在使用这些书籍进行学习时会有一种“盲人摸象”的感觉，不能真正理解 Linux 内核系统具体实现的整体概念，尤其是那些 Linux 系统初学者或刚学会如何使用 Linux 系统的人在使用这些书学习内核原理时，内核的整体运作结构并不能清晰地在脑海中形成。本人在多年的 Linux 内核学习过程中对此也深有体会。而现今流行的 Linux 系统非常庞大和复杂，因此不适合作为操作系统初学者的入门学习起点。这也是作者基于 Linux 早期内核版本写作本书的动机之一。

阅读早期内核的其他好处

目前,已经出现了不少基于 Linux 早期内核而开发的专门用于嵌入式系统的内核版本,如 μClinux 等(在 www.linux.org 上有专门目录),世界上也有许多人认识到通过早期 Linux 内核源代码学习的好处,目前国内也已经有人正在组织人力出版类似本书的书籍。因此,通过阅读 Linux 早期内核版本的源代码,的确是学习 Linux 系统的一种行之有效的方法,并且对研究和应用 Linux 嵌入式系统也有很大的帮助。

在对早期内核源代码的注释过程中,作者发现,早期内核源代码几乎就是目前所使用的较新内核的一个精简版本。其中已经包括了目前新版本中几乎所有的基本功能原理的内容。正如《系统软件:系统编程导论》一书的作者 Leland L. Beck 在介绍系统程序以及操作系统设计时,引入了一种极其简化的简单指令计算机(SIC)系统来说明所有系统程序的设计和实现原理,从而既避免了实际计算机系统的复杂性,又能透彻地说明问题。这里选择 Linux 的早期内核版本作为学习对象,其指导思想与 Leland 的一致。这对 Linux 内核学习的入门者来说,是理想的选择之一。

对于那些已经比较熟悉内核工作原理的人,为了能让自己在实际工作中对系统的实际运转机制不产生一种空中楼阁的感觉,因此也有必要阅读内核源代码。

当然,使用早期内核作为学习的对象也有不足之处,例如早期内核不支持虚拟文件系统 VFS,也不支持网络系统。但由于本书是 Linux 内核的入门教材,简单也正是早期内核版本的优点之一。通过学习本书,可以为进一步学习这些高级内容打下扎实的基础。

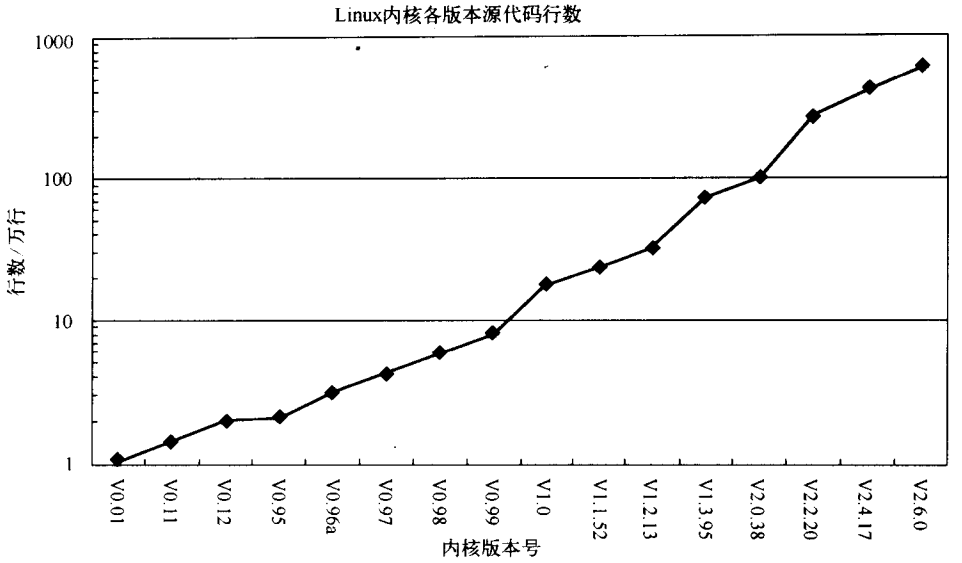
阅读完整源代码的重要性和必要性

正如 Linus 在一篇新闻组投稿上所说的,要理解一个操作系统的真正运行机制,一定要阅读其源代码。操作系统本身是一个整体,具有很多看似不重要的细节,但是若忽略这些细节,就不能真正了解一个实际操作系统的实现方法和手段。

虽然通过阅读一些操作系统原理经典书籍(例如 M. J. Bach 的《UNIX 操作系统设计》)能够对 UNIX 类操作系统的工作原理有一些了解,但实际上对操作系统的真正组成和内部关系实现的理解仍不是很清晰。正如 Tanenbaum 所说的,"许多操作系统教材都是重理论而轻实践","多数书籍和课程为调度算法耗费大量的时间和篇幅而完全忽略 I/O,其实,前者通常不足一页代码,而后者往往要占到整个系统三分之一的代码总量。"内核中大量的重要细节均未提到。因此并不能让读者理解一个真正的操作系统实现的奥妙所在。只有在详细阅读过完整的内核源代码之后,才会对系统有一种豁然开朗的感觉,对整个系统的运作过程有深刻的理解。以后再选择最新的或较新内核源代码进行学习时,也不会碰到大问题,基本上都能顺利地理解新代码的内容。

如何选择要阅读的内核代码版本

那么,如何既能达到上述要求,又不被太多的内容而搞乱头脑,选择一个合适的 Linux 内核版本进行学习,提高学习的效率呢?作者通过对大量内核版本进行比较和选择后,最终选择了与目前 Linux 内核基本功能较为相近,又非常短小的 0.11 版内核作为入门学习的最佳版本。下图是对一些主要 Linux 内核版本行数的统计。



目前的 Linux 内核源代码量都有几百万行，2.6.0 版内核代码行数约为 592 万行，极其庞大，对这些版本进行完全剖析几乎是不可能的。而 0.11 版内核不超过 2 万行代码量，因此完全可以在一本书中剖析清楚。麻雀虽小，五脏俱全。为了对所研究的系统有感性的了解，并能利用实验来加深对原理的理解，作者还专门重建了基于该内核的可运行的 Linux 0.11 系统。由于其中含有 GNU gcc 编译环境，因此使用该系统也能做一些简单的开发工作。

另外，使用该版本可以避免对现有较新内核版本中已经变得越来越复杂的各子系统部分的研究(如虚拟文件系统 VFS、Ext2 或 Ext3 文件系统、网络子系统、新的复杂的内存管理机制等)。

阅读本书需具备的基础知识

在阅读本书时，读者必须具备一些基本的 C 语言知识和 Intel CPU 汇编语言知识。有关 C 语言最佳的参考资料仍然是 Brian W. Kernighan 和 Dennis M. Ritchie 编写的《The C Programming Language》一书。而汇编语言的资料则可以参考任意一本讲解与 Intel CPU 相关的汇编语言教材。另外还需要一些嵌入式汇编语言的知识。有关嵌入式汇编的权威信息都包含在 GNU gcc 编译器手册中。我们也可以从 Internet 上搜索到一些有关嵌入式汇编比较有价值的短文。本书中也包含了一些嵌入汇编基本的语法说明。

除此之外，还希望读者具备有关 80x86 处理器结构和编程的知识以及有关 80x86 硬件体系结构和接口编程的知识；读者还应具备使用 Linux 系统的简单技能。

另外，由于 Linux 系统内核最早是根据 M.J.Bach 的《UNIX 操作系统设计》一书的基本原理开发的，源代码中许多变量或函数的名称都来自该书，因此在阅读本书时若能适当参考该书，会更易于理解内核源代码。

Linus 在最初开发 Linux 操作系统时，参照了 MINIX 操作系统。例如，最初的 Linux 内核版本完全照搬了 MINIX 1.0 文件系统。因此，在阅读本书时，Tanenbaum 的书《操作系统：设计与实现》也具有较大的参考价值。

Ext2 文件系统与 MINIX 文件系统

目前 Linux 系统上所使用的 Ext2(或最新的 Ext3)文件系统是在内核 1.x 之后开发的。其

功能详尽并且性能也非常完整和稳固，是目前 Linux 操作系统上默认的标准文件系统。但是，作为对 Linux 操作系统完整工作原理入门学习所使用的部分，原则上是越精简越好。Linux 内核 0.11 版当时仅包含最为简单的 MINIX 1.0 文件系统，对于理解一个操作系统文件系统的实际组成和工作原理已经足够。这也是选择 Linux 早期内核版本进行学习的主要原因之一。

在完整阅读本书之后，相信您定会发出这样的感叹："对于 Linux 内核系统，我现在终于入门了！"。此时，您应该有十分的把握去进一步学习最新 Linux 内核中各部分的工作原理和过程了。

同济大学
赵炯　博士

目　录

第1章 概 述

本章首先回顾 Linux 操作系统的诞生和发展，读者由此可以理解本书选择 Linux 系统早期版本作为学习对象的一些原因。然后具体说明选择早期 Linux 内核版本的优点和不足之处，以及如何进一步学习。最后对各章的内容进行简要介绍。

1.1 Linux 的诞生和发展

Linux 操作系统是 UNIX 操作系统的一种克隆系统。它诞生于 1991 年 10 月 5 日(这是第一次正式向外公布的时间)。此后借助于 Internet，经过全世界各地计算机爱好者的共同努力，现已成为当今世界上使用最多的一种 UNIX 类操作系统，并且使用人数还在迅猛增长。

Linux 操作系统的诞生、发展和成长过程依赖于以下五个重要支柱：UNIX 操作系统、MINIX 操作系统、GNU 计划、POSIX 标准和 Internet。下面根据这五个基本线索来追寻一下 Linux 的酝酿、开发以及最初的发展。首先分别介绍其中的四个基本要素，然后根据 Linux 的创始人 Linus Torvalds 从对计算机感兴趣而自学计算机知识、到心里开始酝酿编制一个自己的操作系统、到最初 Linux 内核 0.01 版公布以及从此如何艰难地一步一个脚印地在全世界 hacker 的帮助下推出比较完善的 1.0 版本这段时间的发展经过。

当然，目前 Linux 内核版本已经开发到了 2.6.x 版。而大多数 Linux 系统中所用到的内核是稳定的 2.6.12 版内核(其中第 2 个数字若是奇数则表示是正在开发的版本，不能保证系统的稳定性)。

1.1.1 UNIX 操作系统的诞生

UNIX 操作系统是美国贝尔实验室的 Ken Thompson 和 Dennis Ritchie 于 1969 年夏天在 DEC PDP-7 小型计算机上开发的一个分时操作系统。

1969 年夏天，Ken Thompson 为了能在闲置不用的 PDP-7 计算机上运行他非常喜欢的星际旅行(Space travel)游戏，趁他夫人回家乡加利福尼亚度假期间，在一个月内开发出了 UNIX 操作系统的原型。当时使用的是 BCPL 语言(基本组合编程语言)，后经 Dennis Ritchie 于 1972 年用移植性很强的 C 语言进行了改写，使得 UNIX 系统在大专院校得到了推广。

1.1.2 MINIX 操作系统

MINIX 系统是由 Andrew S. Tanenbaum 开发的。Tanenbaum 在荷兰阿姆斯特丹的 Vrije 大学数学与计算机科学系工作，是 ACM 和 IEEE 的资深会员(全世界也只有很少人是两会的资深会员)。共发表了 100 多篇文章和 5 部计算机著作。

Tanenbaum 虽出生在美国纽约，却是荷兰侨民(1914 年他的祖辈来到美国)。他在纽约上中学、麻省理工学院上大学、加州大学 Berkeley 分校念博士学位。由于读博士后的缘故，他来到了家乡荷兰。从此就与家乡一直有来往。后来就在 Vrije 大学开始教书、带研究生。荷兰首都阿姆斯特丹是个常年阴雨绵绵的城市，但对于 Tanenbaum 来说，这最好不过了，因为在这样的环境下

他就可以经常待在家中摆弄他的计算机了。

MINIX 是他 1987 年编制的，主要用于学生学习操作系统原理。到 1991 年时版本是 1.5。目前主要有两个版本在使用：1.5 版和 2.0 版。当时该操作系统在大学使用是免费的。目前 MINIX 系统可以从许多 FTP 上下载。

对于 Linux 系统，他后来曾表示对其开发者 Linus 的称赞。但他认为 Linux 的发展很大原因是由于他为了保持 MINIX 的小型化，能让学生在一个学期内就能学完，因而没有接纳全世界许多人对 MINIX 的扩展要求。因此在这样的前提下 Linus 编写了 Linux 系统。当然 Linus 也正好抓住了这个好时机。

作为一个操作系统，MINIX 并不是优秀者，但它同时提供了用 C 语言和汇编语言编写的系统源代码。这是第一次使得有抱负的程序员或黑客能够阅读操作系统的源代码。在当时，这种源代码是软件商们一直小心守护着的秘密。

1.1.3 GNU 计划

GNU 计划和自由软件基金会 FSF(the Free Software Foundation)是由 Richard M. Stallman 于 1984 年一手创办的。旨在开发一个类似 UNIX 并且是自由软件的完整操作系统：GNU 系统(GNU 是“GNU's Not UNIX”的递归缩写，它的发音为“guh-NEW”)。各种使用 Linux 作为核心的 GNU 操作系统正在被广泛使用。虽然这些系统通常被称作“Linux”，但是 Stallman 认为，严格地说，它们应该被称为 GNU/Linux 系统。

到 20 世纪 90 年代初，GNU 项目已经开发出许多高质量的免费软件，其中包括有名的 emacs 编辑系统、bash shell 程序、gcc 系列编译程序、gdb 调试程序等等。这些软件为 Linux 操作系统的开发创造了一个合适的环境，是 Linux 能够诞生的重要因素。

1.1.4 POSIX 标准

POSIX(Portable Operating System Interface for Computing Systems)是由 IEEE 和 ISO/IEC 开发的一簇标准。该标准基于现有的 UNIX 实践和经验，描述了操作系统的调用服务接口，用于保证编制的应用程序可以在源代码一级上在多种操作系统上移植和运行。它是在 20 世纪 80 年代一个 UNIX 用户组(usr/group)的工作基础上取得的。该 UNIX 用户组原来试图将 AT&T 的 System V 操作系统和 Berkeley CSRG 的 BSD 操作系统的调用接口之间的区别重新调和集成。并于 1984 年定制出了 usr/group 标准。

1985 年，IEEE 操作系统技术委员会标准小组委员会(TCOS-SS)开始在 ANSI 的支持下责成 IEEE 标准委员会制定有关程序源代码可移植性操作系统服务接口正式标准。到了 1986 年 4 月，IEEE 制定出了试用标准。第一个正式标准是在 1988 年 9 月份批准的(IEEE 1003.1-1988)，即以后经常提到的 POSIX.1 标准。

到 1989 年，POSIX 的工作被转移至 ISO/IEC 社团，并由 15 个工作组继续将其制定成 ISO 标准。到 1990 年，POSIX.1 与已经通过的 C 语言标准联合，正式批准为 IEEE 1003.1-1990(也是 ANSI 标准)和 ISO/IEC 9945-1:1990 标准。

POSIX.1 仅规定了系统服务应用程序编程接口(API)，仅概括了基本的系统服务标准。因此工作组期望对系统的其他功能也制定出标准。这样 IEEE POSIX 的工作就开始了。刚开始有十个批准的计划在进行，有近 300 人参加每季度为期一周的会议。着手的工作有命令与工具标准(POSIX.2)、测试方法标准(POSIX.3)、实时 API(POSIX.4)等。到了 1990 年上半年已经有