LINUX 内核精髓

精通LINUX内核必会的75个绝技

LINUX KERNEL HACKS™

高桥 浩和 主编
池田 宗广、大岩 尚宏、岛本 裕志
竹部 晶雄、平松 雅巳 著
杨婷 译
刘波 审校

O'REILLY®
机械工业出版社
China Machine Press
HZ BOOKS
华章科技

Linux 内核精髓

精通 Linux 内核必会的 75 个绝技

高桥 浩和 主编

池田 宗广、大岩 尚宏、岛本 裕志
竹部 晶雄、平松 雅巳 著

杨婷 译

刘波 审校

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'Reilly Media, Inc. 授权机械工业出版社出版

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Linux 内核精髓：精通 Linux 内核必会的 75 个绝技 /（日）高桥 浩和等著；杨婷译 . —北京：机械工业出版社，2013.1

（O'Reilly 精品图书系列）

书名原文：Linux Kernel Hacks

ISBN 978-7-111-41049-2

I. L… II. ①高… ②杨… III. Linux 操作系统 IV. TP316.89

中国版本图书馆 CIP 数据核字（2012）第 318908 号

北京市版权局著作权合同登记

图字：01-2011-7458 号

书　　名/ Linux内核精髓：精通Linux内核必会的75个绝技

书　　号/ ISBN 978-7-111-41049-2

责任编辑/ 谢晓芳

出版发行/ 机械工业出版社

地　　址/ 北京市西城区百万庄大街22号（邮政编码 100037）

印　　刷/ 藁城市京瑞印刷有限公司印刷

开　　本/ 178毫米×233毫米 16开本 26.5印张

版　　次/ 2013年2月第1版 2013年2月第1次印刷

定　　价/ 79.00元（册）

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：(010)88378991；88361066

购书热线：(010)68326294；88379649；68995259

投稿热线：(010)88379604

读者信箱：hzjsj@hzbook.com

O'Reilly Media, Inc. 介绍

O'Reilly Media通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自1978年开始，O'Reilly一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源码峰会，以至于开源软件运动以此命名；创立了Make杂志，从而成为DIY革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项O'Reilly的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar博客有口皆碑。”

——Wired

“O'Reilly凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference是聚集关键思想领袖的绝对典范。”

——CRN

“一本O'Reilly的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照Yogi Berra的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去Tim似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

编者与作者介绍

主编简介

高桥 浩和（Hirokazu Takahashi）

毕业于北海道大学电子工学系。从 VAX 全盛时代开始致力于各种 UNIX 系列操作系统的功能强化和内核调整，以及大规模系统的实时操作系统的设计等。以 ISP 的服务器构建为契机，开始正式研究 Linux。

作者简介

池田 宗广（Munehiro IKEDA）

大学时代，亲眼看到 X68000 的 gcc 生成比主流编译器还要快好几倍的代码，因此开始确信免费软件 / 开源软件的可能性。此后，在历经咖啡店店员、生产技术人员、硬件工程师后，终于开始从事 Linux 内核开发。这个行业最吸引人的就是能够跨公司甚至跨国界与世界最优秀的技术人员进行交流。现居住在美国，爱好音乐演奏，当过鼓手，也当过主唱，最近几年一直在弹贝斯。不管是作为技术人员还是贝斯手都喜欢做幕后工作，只不过天生就不喜欢半途而废。

大岩 尚宏（Naohiro Ooiwa）

任职于 Miracle Linux 株式会社的软件工程师。大学时研究的是类似手机这样使用天线接收无线高频信号的模拟线路。从事 Linux 开发工作的时候开始深入研究软件。他是《Debug Hacks》的作者，本书是 O'REILLY JAPAN 的第二本 Hacks 系列图书。

岛本 裕志（Hiroshi Shimamoto）

软件工程师。负责问题分析和调试。主要工作就是在出现故障时，根据日志和核心转储找出问题所在。因此在工作中会经常用到二进制和 CPU 运行的知识。同时也在论坛中从事过一些关于 x86 架构和调度程序的活动。目前关注虚拟化方面的活动。

竹部 晶雄（Akio Takebe）

在 Xen、KVM 等与虚拟化相关的开源论坛参与开发活动。主要负责 IA64 架构、RAS 系列和 PCI pass through 的开发。在开源论坛认识了专门研究省电技术的工程师，从而开始对省

电方面产生兴趣。现在正使用 Ruby on Rails 开发云计算相关软件。

平松 雅巳（Masami Hiramatsu）

Linux 内核追踪的相关维护人员。主要工作是对 perf 和 ftrace 的动态事件进行维护。也参与了 SystemTap 的开发，最近热衷于将系统 SystemTap 的用途从专门用于追踪扩展到游戏编程等。主要使用的是 bash 和 vim，但是因为 bash 不能用 hjkl 移动光标，总的来说属于 vim 用户。喜欢使用 Ubuntu 和 Fedora。现在的研究方向是 ARM Linux、Btrfs 等。

撰稿人简介

畑山 大辅（HATAYAMA Daisuke）

crash gcore 扩展模块的维护人员。对调试和故障分析感兴趣。最喜欢做的事情就是从元数据对系统进行研究。正在努力练习马拉松长跑，争取在搞技术的同时锻炼出健康的体魄。近期目标是四小时内跑完马拉松。

藤田 朗（Akira Fujita）

任职于 NEC 软件东北株式会社。担任软件工程师。大学毕业之后开始转向软件行业。喜欢 Linux 文件系统（ext3/ext4）。喜欢 defrag，爱好五人足球。

技术审校者简介

刘波，资深 Linux 内核开发工程师、应用开发工程师和嵌入式开发工程师，现在重庆工商大学计算机科学与信息工程学院担任教师，从事 Linux 程序开发和 Oracle 管理方面的教学工作，在读博士。此外，他还专注于大规模机器学习、数值分析与计算、最优化理论（凸优化）的研究。

致谢

本书的编著工作受到了多方的大力支持。计划还未确定时，O'REILLY JAPAN 就对此表示了很大的兴趣，并给予我们参与写作的机会，对此我们要向 O'REILLY JAPAN 的各位表示衷心的感谢。特别是受到担任编辑的赤池凉子女士的很多照顾。写作进度滞后时，赤池女士依旧迅速地安排了后面的多次修改，非常感谢她。

对作为作者参与写作的池田宗广、平松雅巳先生，以及非常爽快地同意作为撰稿人紧急参与写作的藤田朗、畑山大辅先生也要表示衷心的感谢。在各位的辛勤努力下，本书才能够具有如此丰富和引人入胜的内容。

对百忙之中抽出时间负责主编的高桥浩和先生也表示衷心的感激。对写作过程中的技术

趋势以及所有章节都进行了详细的指导。也非常感谢三好和人、永野武先生为我们免费提供很多原稿。

此外，非常感谢《Debug Hacks》的作者安部东洋，为本书进行指导，使得本书质量大幅提高。

在从一开始就共同执笔的岛本裕志、竹部晶雄先生的努力下，本书才得以顺利出版。感谢你们。

最后，借此机会向本书写作过程中为我们提供协助的各方人士表示衷心的感谢。

大岩 尚宏

主编致辞

从 1991 年 Linux 内核诞生，到现在已经过去了 20 多年，现在 Linux 3.0 也即将发布。在这 20 多年间，Linux 内核已经进化成可以在便携式计算机到大型服务器的各种硬件上运行的操作系统。至今仍有众多开发人员在不断地对 Linux 内核进行开发。

虽然网络上有很多关于 Linux 的信息，但是关于熟练使用 Linux 内核或者参与 Linux 内核开发所需的信息并不多。因此我们决定从这些信息中筛选出 Linux 技术人员可能感兴趣的内容，汇编成一本书。关于省电和虚拟化的介绍是非常符合当前市场需求的。高级内核的概要分析功能也是内核开发人员的必需工具。

在有限的篇幅内能够介绍的内容并不是很多，我们只是希望能够以此为契机，激发更多人对 Linux 内核的兴趣，并实际参与内核开发。

高桥 浩和

前言

内核是操作系统的核心，操作系统的基本功能都是由内核提供的。文件生成和数据包传输等也是通过内核的功能实现的。但这些都不是简单的任务。平时可能意识不到，但这其中确实包含了很多先进技术。例如，在文件系统方面，配置文件时尽量减少磁盘扫描，在网络方面，由于路由表的入口数量庞大，因此设计时尽量保证对系统整体影响较小的设计。在内存管理、进程管理方面也作出了很多努力。解读这种先进技术也是内核构建的魅力之一。

然而，最近的 Linux 所提供的并不只有基本功能。随着功能的不断发展，现在已经出现了很多特定领域的便捷功能和独特功能。即使是内核黑客也很少有人能够完全掌握。

本书从 Linux 内核的众多先进功能中选取了一些必备并且有趣的内容进行介绍，同时也对内部的运行机制和结构进行了阐述。此外，本书还介绍了熟练使用这些功能所需的工具、设置方法以及调整方法等。

省电就是其中一项内容。除了使用方法以外，本书还介绍了省电的理念、与硬件的关系等。此外，还提到了当前广受关注的虚拟化、资源管理、标准文件系统中所采用的 ext4 等已有功能和新功能。对于已有功能，本书结合最新的源码，介绍它的更改内容和新增功能。其中也包括文档中没有记载，且必须对内核内部有一定理解才能得知的信息，因此，即使是比较了解这个功能的人也可能会有新的发现。另外，本书还介绍了内核的相关工具，其中 `gcore` 在重要的系统中就是非常可靠的工具。

最新的 Linux 内核中安装了强大的追踪、概要分析功能，具备很多方便实用的功能。这些功能不仅能够很方便地达到预期的目的，而且对于分析内核功能也非常有用。甚至对于内核构建的高手也有一定帮助。

全书列举了非常多的实例，让读者更快地学会如何使用。对于想要熟练使用内核的读者来说，本书也是非常好的参考书。

本书还为想要了解 Linux 内核的读者以及读过本书后开始对 Linux 内核开发产生兴趣的读者，介绍了获取内核源码的方法和内核开发方法等内核构建入门所需的信息。我们希望读者能够通过本书更加了解 Linux 的世界。

在电脑刚刚诞生的时候，有一段时期人们认为“如果想要提高编程水平就查看 UNIX 代码”。因为最快的方式就是参考天才所编写的最先进的代码并进行模仿。而在阅读 Linux

内核的代码时，相信大家也会深有同感。

Linux 内核是开源软件，无论是谁都可以参与开发。Linux 内核的代码花费了大量的时间和精力来编写。各领域都由具有专业知识的维护人员进行长期的管理，从而得到不断的改进。基于电子邮件的开发也在不断进行，因此可以看到各种讨论，并了解到当前代码的发展历程。每次看到 Linux 内核的代码，都会让人感叹其中凝聚的智慧和努力，也感受到当时的辛苦。希望读者能够从本书开始接触 Linux 这个不一般的世界，诞生更多的内核高手。

本书主要内容

本书介绍的是 Linux 内核所提供的功能。不仅有比较基础的功能，还有一些功能需要具有一定的知识才能使用。

此外，还介绍了使用功能时需要用到的信息和命令。除了内核以外，本书还将介绍相关的应用程序。基本上是基于 TUI 进行说明的，但也有一部分关于 GUI 的介绍。

涉及的主要版本为 Linux 内核 2.6.18 到写作时最新的 Linux 内核 3.0[注1]。其中一部分还介绍了 Red Hat Enterprise Linux 4（RHEL4：基于 Linux 内核 2.6.9）的功能。示例代码已经在工作中经常使用的 RHEL 和任何用户都可以使用的 Fedora、CentOS 等中进行过严格测试。

本书不涉及 Linux 内核的实际安装和以算法等为主体的内容。

本书使用方法

本书可以按顺序依次阅读，另外由于每一节之间都是独立的，因此也可以从感兴趣的章节开始阅读。第 1 章介绍了内核的基础知识，如果是第一次接触内核，建议先学习第 1 章。本书在介绍已有功能时也加入了一些新的信息。相信即使是经验丰富的人也可以在本书中有新的发现，因此希望各位读者能够将本书从头到尾完整读一遍。本书还收录了一些作者珍藏的信息。详细内容请参见参考文献。

本书约定

等宽字体（sample）

表示文件名、文件的内容、控制台的输出、变量名称、命令、命令选项、数据包名称、模块名称、驱动程序名称、键、内核配置、样本代码、其他代码等。

等宽粗体（sample）

表示应替换为用户输入的命令或文本等。

注 1：写作本书时已经发布了 3.0-rc 版本。

斜体（sample）

表示根据环境决定的值等。

小贴士：表示提示、建议、补充事项等。

注意事项：表示注意、警告等。

每一节标题左侧的温度计图标表示该节的相对难易度。

意见与提问

关于本书的内容，我们尽最大的努力进行了验证和确认，但可能还是会存在错误或不正确的地方，或者是会引起误解或混淆的表述、输入错误等。如果在阅读本书的过程中发现了这些问题，请告知我们，以便进行改善。

株式会社 O'REILLY（奥莱利）JAPAN

邮编 160-0002　东京都新宿区坂町 26 番地 27 Intelligent 大厦 1 层

电话　　03-3356-5227

FAX　　03-3356-5261

电子邮件　　japan@oreilly.co.jp

关于本书的技术性问题和意见请发送到下列邮件地址。

japan@oreilly.co.jp

本书的网站上可以找到示例代码[注2]、勘误表和附加信息。

http://www.oreilly.co.jp/books/9784873115016/

关于 O'REILLY 的其他信息请参考下列网站。

http://www.oreilly.co.jp/（日语）

http://www.oreilly.com/（英语）

注 2：这些示例代码是笔者写作时使用的程序，并不保证在各种环境下都可以运行。另外，有时会不经过提示进行修改。示例代码不一定都能对应，敬请谅解。

目录

编者与作者介绍
主编致辞
前言

第 1 章　内核入门 ……1

HACK #1　如何获取 Linux 内核 ……1
HACK #2　如何编译 Linux 内核 ……7
HACK #3　如何编写内核模块 ……18
HACK #4　如何使用 Git ……22
HACK #5　使用 checkpatch.pl 检查补丁的格式 ……41
HACK #6　使用 localmodconfig 缩短编译时间 ……44

第 2 章　资源管理 ……47

HACK #7　Cgroup、Namespace、Linux 容器 ……47
HACK #8　调度策略 ……55
HACK #9　RT Group Scheduling 与 RT Throttling ……59
HACK #10　Fair Group Scheduling ……62
HACK #11　cpuset ……65
HACK #12　使用 Memory Cgroup 限制内存使用量 ……68

HACK #13 使用 Block I/O 控制器设置 I/O 优先级 ····· 74
HACK #14 虚拟存储子系统的调整 ····· 80
HACK #15 ramzswap ····· 85
HACK #16 OOM Killer 的运行与结构 ····· 91

第 3 章 文件系统 ····· 98

HACK #17 如何使用 ext4 ····· 98
HACK #18 向 ext4 转换 ····· 101
HACK #19 ext4 的调整 ····· 104
HACK #20 使用 fio 进行 I/O 的基准测试 ····· 111
HACK #21 FUSE ····· 118

第 4 章 网络 ····· 121

HACK #22 如何控制网络的带宽 ····· 121
HACK #23 TUN/TAP 设备 ····· 126
HACK #24 网桥设备 ····· 129
HACK #25 VLAN ····· 133
HACK #26 bonding 驱动程序 ····· 136
HACK #27 Network Drop Monitor ····· 141

第 5 章 虚拟化 ····· 147

HACK #28 如何使用 Xen ····· 147
HACK #29 如何使用 KVM ····· 153
HACK #30 如何不使用 DVD 安装操作系统 ····· 159
HACK #31 更改虚拟 CPU 分配方法，提高性能 ····· 161
HACK #32 如何使用 EPT 提高客户端操作系统的性能 ····· 166
HACK #33 使用 IOMMU 提高客户端操作系统运行速度 ····· 173
HACK #34 使用 IOMMU+SR-IOV 提高客户端操作系统速度 ····· 183
HACK #35 SR-IOV 带宽控制 ····· 187
HACK #36 使用 KSM 节约内存 ····· 189
HACK #37 如何挂载客户端操作系统的磁盘 ····· 194

HACK #38 从客户端操作系统识别虚拟机环境 ····· 200
HACK #39 如何调试客户端操作系统 ····· 205

第 6 章 省电 ····· 213

HACK #40 ACPI ····· 213
HACK #41 使用 ACPI 的 S 状态 ····· 224
HACK #42 使用 CPU 省电（C、P 状态）····· 226
HACK #43 PCI 设备的热插拔 ····· 236
HACK #44 虚拟环境下的省电 ····· 240
HACK #45 远程管理机器的电源 ····· 246
HACK #46 USB 的电力管理 ····· 251
HACK #47 显示器的省电 ····· 254
HACK #48 通过网络设备节省电能 ····· 260
HACK #49 关闭键盘的 LED 来省电 ····· 263
HACK #50 PowerTOP ····· 269
HACK #51 硬盘的省电 ····· 276

第 7 章 调试 ····· 282

HACK #52 SysRq 键 ····· 282
HACK #53 使用 diskdump 提取内核崩溃转储 ····· 288
HACK #54 使用 Kdump 提取内核崩溃转储 ····· 293
HACK #55 崩溃测试 ····· 297
HACK #56 IPMI 看门狗计时器 ····· 299
HACK #57 NMI 看门狗计时器 ····· 305
HACK #58 soft lockup ····· 307
HACK #59 crash 命令 ····· 312
HACK #60 核心转储过滤器 ····· 326
HACK #61 生成用户模式进程的进程核心转储 ····· 329
HACK #62 使用 lockdep 查找系统的死锁 ····· 335
HACK #63 检测内核的内存泄漏 ····· 341

第 8 章　概要分析与追踪 ····· 346
HACK #64　使用 perf tools 的概要分析（1）····· 346
HACK #65　使用 perf tools 的概要分析（2）····· 349
HACK #66　进行内核或进程的各种概要分析····· 353
HACK #67　追踪内核的函数调用 ····· 360
HACK #68　ftrace 的插件追踪器 ····· 366
HACK #69　记录内核的运行事件 ····· 371
HACK #70　使用 trace-cmd 的内核追踪 ····· 378
HACK #71　将动态追踪事件添加到内核中 ····· 382
HACK #72　使用 SystemTap 进行内核追踪 ····· 388
HACK #73　使用 SystemTap 编写对话型程序····· 394
HACK #74　SystemTap 脚本的重复利用 ····· 399
HACK #75　运用 SystemTap ····· 402

第 1 章

内核入门

一提起内核包，总会让人感觉似乎困难至极、如临深渊一般。但其基本的操作与其他开放源代码软件包并没有什么不一样，都是首先获取源代码，进行解读，然后修改或者添加新功能对应的代码，并编译、测试。本章将介绍这些内核包操作中最基础的知识，以及 Linux 内核特有的方法。

HACK #1 如何获取 Linux 内核

本节介绍获取 Linux 内核源代码的各种方法。

“获取内核”这个说法看似简单，其实 Linux 内核有很多种衍生版本。要找出自己想要的源代码到底是哪一个，必须首先理解各种衍生版本的意义。

接下来将简单介绍 Linux 内核的开发模式，并分析各种衍生版本在其中所处的地位，然后介绍获取这些衍生版本的源代码的方法。

内核的种类

想要获取正确的 Linux 内核源代码，首先必须了解 Linux 内核的开发模式。

Linux 内核是由多个开发者以分散型的模式进行开发的。这里出现的“分散型”，是指多个衍生源码树同时存在。下面将简单介绍一些具有代表性的源码树及其地位。

Linus 树

最具有代表性的源码树，应属 Linux 内核的最初创始人——Linus Torvalds 所管理的 Linus 树。新版本 Linux 内核的发布，就意味着 Linus 树的源代码被贴上了新发布版本的标签。到 2011 年为止，Linux 内核的版本号一直是用 2.6.x 这样的三个数字来表示的[注 1]。Linus 树一直被认为是 Linux 内核源代码的“根源”，因此一旦其发布了新版本，其他的开发树就会将自己独特的开发成果移植到这个版本上，在此基础上再次进行开发。Linus 树由于其“根源”的地位而称为主线（mainline）。

注 1：Linux 2.6.39 的下一版本将是 Linux 3.0。

一旦发布新版本 Linus 树，就会立刻打开一个“合并窗口”（merge window），接受下一版本需要作出的改变。合并窗口将开启约两周时间。合并窗口关闭后，就会发布下一版本的候选版，即所谓的“rc 内核”[注2]。从 rc 内核发布后到下一版本发布的期间为测试期，这一期间基本只接受关于 bugfix 的修改。rc 版内核每隔约一周时间会依次推出 rc1、rc2……当 Linus 判断其质量已经达到可以发布的水平时，就会作为新版本发布。按照最近的实际情况来看，基本上在 rc6 ～ rc9 左右就会发布新版本，也就是说 Linux 内核每隔 2 ～ 3 个月就会发布新版本。新版本发布后，又会打开下一版本的合并窗口，然后对 rc 版进行测试。Linux 内核就是按照这样的周期来开发的。

小贴士：Linus 树的内核由于完全没有任何华而不实的东西，因此称为“香草”（vanilla）内核或“库存”（stock）内核。

linux-next 树

这是一个为发布将来的版本而积累新代码并进行测试的源码树，主要由 Stephen Rothwell 等人进行管理和运营。原则上要添加新功能或者进行安装配置时，首先要在 linux-next 树中进行测试，在确认各自之间可以兼容之后再添加到 Linus 树内。

stable 树

这是一个主要只针对过去发布的内核版本进行 bug 修改，使其更加稳定的树，由 Greg Kroah-Hartman、Chris Wright 进行维护管理。这个树的版本号是在 Linus 树的版本号后面加一位数字，以 2.6.x.y 这样的 4 个数字来表示。针对某个 Linus 树版本的稳定（stable）版维护一般持续 6 个月左右，但也有持续更久的。

开发树

Linux 内核可以说是各种功能的集合体。例如内存管理、文件系统、网络、各种设备驱动程序、CPU 架构固有部分等。这些功能部分称为“子系统”，各子系统分别在不同的源码树中进行开发。在开发、修改过程中也有一些不属于特定子系统的内容，这些内容首先会被发送到 Andrew Morton 管理的 mm 树（准确地说是 mmotm：mm on the moment，补丁包的缩写）。这样的源码树统称为“开发树”。

在各开发树中开发出的源代码在经过 linux-next 中的测试后再植入 Linus 树。

开发树的数量多如繁星。如果哪天你因为想要开发某个功能而在手边的源代码上进行了修改，这也可以说是一个“开发树”。

Linus 树、开发树等作为所有树的根源，也称为“upstream”，即“上游”。但这是广义上的叫法，有时也仅指最上游的 Linus 树。

注 2：rc 是 release candidate（发布候选）的缩写。