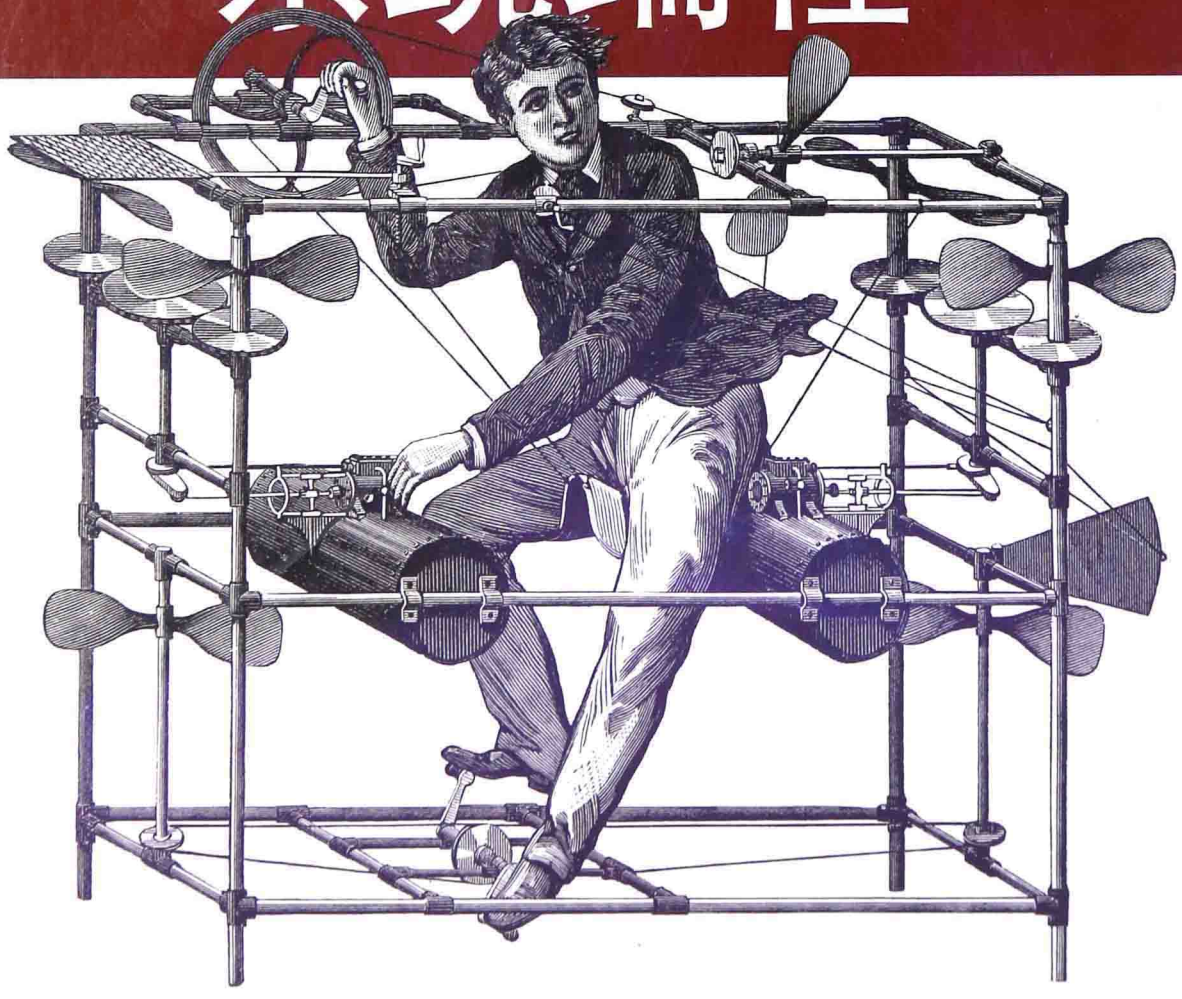
LINUX SYSTEM PROGRAMMING

第2版

根据Linux内核3.0更新

LINUX 系统编程



[美] Robert Love 著

祝洪凯 李妹芳 付途 译

O'REILLY®

人民邮电出版社
POSTS & TELECOM PRESS

O'REILLY®

Linux 系统编程

（第 2 版）

[美] Robert Love 著
祝洪凯 李妹芳 付 途 译

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

Linux系统编程 : 第2版 / （美）拉姆（Love, R.）著；祝洪凯，李妹芳，付途译. -- 北京 : 人民邮电出版社，2014.5
ISBN 978-7-115-34635-3

Ⅰ. ①L… Ⅱ. ①拉… ②祝… ③李… ④付… Ⅲ. ①Linux操作系统 Ⅳ. ①TP316.89

中国版本图书馆CIP数据核字(2014)第035762号

内 容 提 要

系统编程是指编写系统软件，其代码在底层运行，直接跟内核和核心系统库对话。

本书是一本关于 Linux 系统编程的教程，也是一本介绍 Linux 系统编程的手册，还是一本如何实现更优雅更快代码的内幕指南。全书分为 11 章和 2 个附录，详细介绍了 Linux 系统编程基本概念、文件 I/O、缓冲 I/O、高级文件 I/O、进程管理、高级进程管理、线程、文件和目录管理、信号和时间等主题。附录给出了 gcc 和 GNU C 提供的很多语言扩展，以及推荐阅读的相关书目。

本书的作者是知名的 Linux 内核专家，多本技术图书的作者。本书需要在 C 编程和 Linux 编程环境下工作的程序员阅读，对于想要巩固基础或了解内核的高级编程人员，本书也具有参考价值。

◆ 著 [美] Robert Love
译 祝洪凯 李妹芳 付 途
责任编辑 陈冀康
责任印制 程彦红 杨林杰

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
三河市海波印务有限公司印刷

◆ 开本：787×1000 1/16
印张：26
字数：490 千字 2014 年 5 月第 1 版
印数：1 - 4 000 册 2014 年 5 月河北第 1 次印刷

著作权合同登记号 图字：01-2013-7659 号

定价：79.00 元

读者服务热线：(010)81055410 印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京崇工商广字第 0021 号

译者序

本书可以作为 Linux 系统编程的指南和手册，但它又不仅仅是简单的 man 手册。作为一名狂热的内核爱好者，Robert 在书中倾注了很多自己的分析和思考。也许正因为其对 Linux 系统理解之深，在介绍系统编程涉及的方方面面时，才能够如此驾轻就熟，并分享很多实用的技巧。

本书对 Linux 文件系统、I/O、缓存、进程、内存等的描述深入到位，处处渗透着作者的理解和经验。系统编程涉及的函数之多，很容易让初学者感觉置身于一片浩渺的天地，无所是从。本书无疑是他们的福音。一本书能够化繁为简，能够非常清晰地描述相关领域功能的各函数（以及如何用好的谆谆告诫），能够非常自然有条理地把它们组织起来，读起来很顺畅，如果作者没有十足的“功力”，是做不到的。这让我想起 Robert 在附录 B 中提到“如果你写 C 代码不能够像说母语那样流利……”。Robert 也是 80 后，年纪和我们相仿，其对技术的驾驭之深，让我无限敬慕。

本书可以作为枕边书，经常翻阅。在今天这个互联网时代，“不会就问 Google”可能已经成为技术人员解决问题最常用的诉求。他山之石，可以攻玉。这种方式简单、便捷、高效，我也经常这么做。但从个人经验看，这种方式虽然可以快速解决问题，但其知识积累较难成体系，或者不够全面。比如，想把时间转换成本地时间，了解到 localtime 函数可以实现时，就轻松地实现了，却不曾料到 localtime 函数不是线程安全的，埋下了隐患。其实这样的例子很多。现在，有这么一本书，能够帮助我们避免许多坑，为什么不看看呢？

更可贵的是，本书第 2 版增加了多线程的介绍，这些内容非常实用。因为多线程是个比较难的专题，令多少程序员望而生畏。感谢 Robert 在这些深度领域的指引，由于篇幅所限，虽不详尽，却简单扼要，简单的描述加上示例代码说明，让人一下明白多线程是怎么回事，从此不再畏惧，更多细节自会探究。

总体而言，本书内容顺畅，我觉得不但可以作为枕边书多读几遍深度了解，也可以作为手册指南经常查阅参考。

本书在翻译过程中参考了吴晋老师指导的哈尔滨工业大学浮图开放实验室为交流、

学习而翻译的本书第 1 版。饮水思源，这里谨向参与该第 1 版的所有翻译人员深表敬意和致谢。此外，也深深感谢婆婆韩学美，感谢编辑陈冀康先生以及所有其他为本书付出努力的人们。

由于译者水平有限，错漏之处在所难免，请各位读者不吝批评指正。

李妹芳

2013 年 11 月 24 日

序

Linux 内核开发人员在抱怨时，经常会抛出这么一句话：“用户空间只不过是给内核玩玩而已。”[1]

内核开发人员咕哝这句话，是想尽可能摆脱用户空间代码运行失败的责任。对他们而言，用户空间开发人员应该负责解决自己的代码 bug，因为内核绝对不会有任何问题。

为了证明往往不是内核问题，一名资深的 Linux 内核开发人员在 3 年前曾在会议上分享了一个关于“为何用户空间这么让人讨厌？”的讲座，指出真实世界中很多人每天都依赖的一些非常糟糕的用户空间代码。有些内核开发人员甚至创建了一些工具，来说明用户空间代码是如何滥用硬件和白白消耗笔记本电池的。

然而，虽然内核开发人员可以不无轻蔑地认为用户空间代码只是给内核玩玩（test load），而实际上所有的内核开发人员每天也都在依赖用户空间代码。否则，他们在屏幕上只能看到内核输出 ABABAB 这类东西。

现在，Linux 已经成为有史以来最灵活最强大的操作系统，不仅运行在最小的手机设备和嵌入设备上，而且运行在世界上超过 90%的最强大的超级计算机上。和 Linux 相比，没有其他任何一个操作系统可以扩展得这么好，并能够解决不同的硬件类型和环境所面临的所有挑战。

有了 Linux 内核，在 Linux 的用户空间上运行的代码也可以在其他平台上运行，提供人们所依赖的真正应用和工具。

在这本书中，Robert Love 真正做到“诲人不倦”，告诉读者关于 Linux 系统的所有系统调用。因此，他写下本书，从而可以帮助你从用户空间角度看，深入理解 Linux 内核是如何工作的，以及如何充分利用系统提供的各种功能。

这本书的内容有助于你编写可以运行在不同版本的 Linux 和不同硬件类型上的代码。通过这本书，你可以理解 Linux 是如何工作的，以及如何有效利用其灵活性。

最后，它还教你如何实现“不让人讨厌”的代码，这一点非常重要。

——Greg Kroah-Hartman

[1] 译注：原文是：“User space is just a test load for the kernel.”，实在觉得很难表达出“抱怨”的味道。

致

谨致 Doris 和 Helen。

前言

这本书是关于 Linux 上的系统编程。“系统编程”是指编写系统软件，其代码在底层运行，直接跟内核和核心系统库对话。换句话说，本书的主题是 Linux 系统调用和底层函数说明，如 C 库定义的函数。

虽然已经有很多书探讨 UNIX 上的系统编程，却很少有专注于探讨 Linux 方面的书籍，而探讨最新版本的 Linux 以及 Linux 特有的高级接口的书籍更是凤毛麟角。此外，本书还有一个优势：我为 Linux 贡献了很多代码，包括内核及其上面的系统软件。实际上，本书中提到的一些系统调用和系统软件就是我实现的。因此，本书涉及很多内幕资料，不仅介绍系统接口如何工作，还阐述它们实际上是如何工作，以及如何高效利用这些接口。因此，本书既是一本关于 Linux 系统编程的教程，也是一本介绍 Linux 系统调用的手册，同时还是一本如何实现更优雅、更快代码的内幕指南。这本书内容翔实，不管你是否每天都在编写系统级代码，本书给出的很多技巧都有助于你成为更优秀的软件工程师。

目标读者和假设

本书假定读者熟悉 C 编程和 Linux 编程环境——不要求很精通，但至少比较熟悉。如果你不习惯于 UNIX 文本编辑器——Emacs 和 vim（后者成为最广泛使用的编辑器，而且评价很高），那么至少应该熟悉一个。你还应该对如何使用 gcc、gdb、make 等工具很熟悉。已经有很多书籍介绍了关于 Linux 编程的工具和实践，本书最后的附录 B 给出一些有用的资源。

我并没有假设用户了解 UNIX 或 Linux 系统编程。本书是从零开始，从最基本的开始介绍，一直到高级接口和一些优化技巧。我希望不同层次的读者都能够从本书学到一些新东西，觉得本书有价值。在写本书过程中，我自己就感觉颇有收获。

同样，我并不想去说服或鼓励读者做什么。目标读者显然是那些希望能够（更好地）在系统上编程的工程师，但是希望奠定更坚实的基础的高级编程人员还可以找到很多其他有趣的资料。本书也适合那些只是出于好奇的黑客，它应该能够满足他们的好奇心。本书目标是希望能够满足大部分的编程人员。

不管出于什么目的，最重要的是，希望你会觉得本书很有意思。

本书的内容

本书共包含 11 章和 2 个附录。

第 1 章，入门和基本概念

本章是入门介绍，简要介绍了 Linux、系统编程、内核、C 库和 C 编译器。即使是高级用户也应该看看本章内容。

第 2 章，文件 I/O

本章介绍文件，它是 UNIX 环境的最重要的抽象，介绍文件 I/O，它是 Linux 编程模型的基础。其内容涉及读写文件以及一些其他基础的文件 I/O 操作。最后还探讨了 Linux 内核是如何实现和管理文件的。

第 3 章，缓冲 I/O

本章探讨了基础文件 I/O 接口的一个方面：缓存大小管理，它从解决方案角度探讨了缓冲 I/O 和标准 I/O。

第 4 章，高级文件 I/O

本章阐述了高级 I/O 接口、存储映射和优化机制。它探讨了如何避免查找的很多技巧，并介绍了 Linux 内核 I/O 调度器。

第 5 章，进程管理

本章介绍了 UNIX 第二大重要抽象：进程，以及与基础进程管理相关的一系列系统调用，包括“久经风霜”的 fork 调用。

第 6 章，高级进程管理

本章继续探讨高级进程管理，包括实时进程。

第 7 章，线程

本章探讨了线程和多线程编程。它重点讨论高级设计概念，包括对 POSIX 线程 API（即 Pthreads）的介绍。

第 8 章，文件和目录管理

本章阐述了文件和目录的创建、移动、拷贝、删除以及其他操作。

第 9 章，内存管理

本章探讨内存管理。它首先介绍内存的 UNIX 概念，比如进程地址空间、分页，然后探讨了从内核获取内存以及把内存返还给内核的接口，最后介绍高级内存相关的接口。

第 10 章，信号

本章涉及信号。它首先介绍信号及其在 UNIX 系统中的作用，然后阐述信号接口，从最基础的接口开始探讨，一直到高级接口。

第 11 章，时间

本章探讨了时间、睡眠和锁管理。它从基础的接口开始介绍，一直到 POSIX 时钟和高精度的定时器。

附录 A

附录 A 给出了 gcc 和 GNU C 提供的很多语言扩展，比如把函数标识为常函数、纯函数和内联函数的属性。

附录 B

附录 B 给出了推荐阅读书目，它们不但是本书很好的补充，而且也涵盖本书没有涉及的前提背景知识。

本书涉及的版本

Linux 系统接口是可以定义为由 Linux 内核（操作系统内核）、GNU C 库（glibc）和 GNU C 编译器（gcc，正式命名为 GNU Compiler Collection，不过我们只关注与 C 相关的）提供的应用二进制接口和应用编程接口。本书探讨的系统接口是分别由以下版本定义的：Linux 内核 3.9、glibc 2.17 和 gcc 4.8。本书提到的接口应该可以向前兼容，即可以兼容更高版本的内核、glibc 和 gcc。也就是说，新版本的组件应该继续遵循本书阐述的接口和行为。此外，本书探讨的很多接口一直是 Linux 的一部分，因此它们对于老版本的内核、glibc 和 gcc 也是向后兼容的。

如果把一个不断发展的操作系统比作运行的目标，那 Linux 就是个快速奔跑的猎豹。Linux 发展是按天衡量的，而不是按年，内核和其他组件的频繁发布不断改变 Linux 的方方面面。没有一本书可以不断地追赶上这么快的节奏。

然而，系统编程定义的编程环境是不变的。内核开发人员尽了最大努力不要打破系

统调用，glibc 开发人员非常在乎向前兼容和向后兼容。Linux 工具链生成了跨版本的可兼容代码。因此，虽然 Linux 本身可能变化很快，Linux 系统编程还是很稳定，而一本基于该系统的书，尤其在 Linux 生命周期的这个时期，却是具有极大持久性的。我想说得很简单：不要担心系统接口的变化，可以下决心购买本书！

本书使用的体例

本书遵循以下字体体例：

斜体（*Italic*）

表示新的术语、URL、E-mail 地址、文件名和文件扩展名。

等宽字体（`Constant width`）

用于程序清单以及段落中的程序单元，如变量或函数名称、数据库、数据类型、环境变量、声明和关键字。

等宽粗体字（**`Constant width bold`**）

显示命令或者其他由用户输入的文本。

等宽斜体字（*`Constant width italic`*）

表示必须根据用户提供的值或者由上下文决定的值进行替代的文本。

该图标表示提示、建议或普通注意事项。

该图标表示告警或警告。

本书中的大部分代码片段形式简单、可重用。它们看起来如下：

```
while (1) {
        int ret;

        ret = fork ();
        if (ret == -1)
                perror ("fork");
}
```

为了使代码片段既看起来简洁，又是可用的，我们做了很大努力。不要特殊的头文件、各种宏定义以及不可识别的简写。我们并没有构建非常巨型的程序，而是给出

很多简单的示例。由于示例必须易于描述、可用，并且简单清晰，我希望这些示例可以作为第一次阅读本书的有用教程，而在后续阅读过程中，依然可以作为参考手册。

本书中几乎所有的示例都是自包含的。这意味你可以把这些示例代码复制到自己的文本编辑器中，并利用这些代码片段。除非特别提出，所有代码片段都应该不需要任何特殊的编译器标志位就可以编译通过（在极少数情况下，需要链接到某个特定的库）。我建议通过以下命令来编译一个源文件：

```
$ gcc -Wall -Wextra -O2 -g -o snippet snippet.c
```

通过以上命令，会把源文件 snippet.c 编译成可执行的二进制文件 snippet，支持很多告警检查、重要明智的优化以及调试。本书提供的代码通过这种编译方式应该可以编译通过，而不会生成错误和告警信息——虽然你可能需要首先把代码补充完整。

当一节介绍新的函数时，该函数是以 UNIX 的 man 页面格式给出，看起来如下：

```
#include <fcntl.h>

int posix_fadvise (int fd, off_t pos, off_t len, int advice);
```

需要包含的头文件，以及所有定义，都是在最上方，然后是该调用的完整形式。

使用本书的样例代码

本书是为了帮助你完成工作。通常来说，你可以在自己的程序和文档中使用本书的代码。除非你使用了本书的大量代码，否则你无需联系我们获取许可。例如，写一个程序用到本书的几段代码不需要获得许可，销售和分发 O'Reilly 丛书的代码需要获得许可；引用本书的样例代码来解决一个问题不需要获取许可，使用本书的大量代码到你的产品文档中需要获得许可。

我们不要求你（引用本书时）给出出处，但是如果你这么做，我们对此表示感谢。出处通常包含标题、作者、出版社和 ISBN。例如“*Linux System Programming*, Second Edition,by Robert Love (O' Reilly). Copyright 2013 Robert Love, 978-1-449-33953-1.”。

如果你觉得你对本书样例代码的使用超出了这里给出的许可范围，请与我们联系：permissions@oreilly.com。

由于本书的示例代码非常多而且很短，所以没有提供在线资源。

致谢

本书初稿得到了很多人的帮助。由于无法一一列出，在这里我谨向那些一路给予帮助、鼓励、认可和支持的朋友们致以诚挚的谢意。

Andy Oram 是一位非常优秀的编辑。如果没有他的努力工作，就不会有本书。Andy 不但对技术了解非常深入，而且有诗人般的英语表达能力。

本书还得到一些非常资深的技术专家帮助审查，他们是自己所在领域的真正专家，如果没有他们的帮助，本书的最终版本和你现在看到的相比会失色很多。这些技术专家包括 Jeremy Allison、Robert P. J. Day、Kenneth Geisshirt、Joey Shaw 和 James Willcox。虽然他们给了非常大的帮助，但本书还是难免有些错误。

在 Google，和同事们一起工作是非常快乐的事情，他们是我遇到的最聪明、最专注的工程师。每天都是挑战，这是我们工作状态的最佳描述。感谢系统方面的项目，帮助我写下很多东西，以及有种氛围，鼓励着我完成本书。

特别感谢 Paul Amici、Mikey Babbitt、Nat Friedman、Miguel de Icaza、Greg Kroah-Hartman、Doris Love、Linda Love、Tim O'Reilly、Salvatore Ribaudo 及其家属、Chris Rivera、Carolyn Rodon、Joey Shaw、Sarah Stewart、Peter Teichman、Linus Torvalds、Jon Trowbridge、Jeremy VanDoren 及其家属、Luis Villa、Steve Weisberg 及其家属和 Helen Whisnant。

最后，感谢我的父母 Bob 和 Elaine。

——Robert Love，Boston

目录