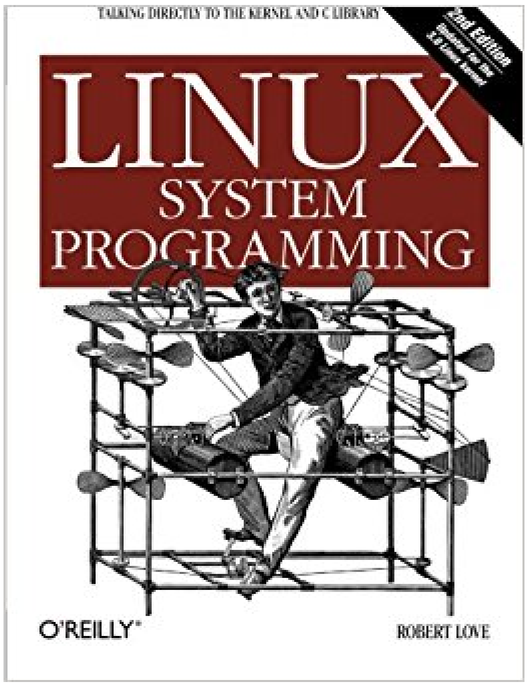
Linux System Programming: Talking Directly To The Kernel And C Library



Synopsis

Write software that draws directly on services offered by the Linux kernel and core system libraries. With this comprehensive book, Linux kernel contributor Robert Love provides you with a tutorial on Linux system programming, a reference manual on Linux system calls, and an insiderÃ¢â ¬â„¢s guide to writing smarter, faster code.Love clearly distinguishes between POSIX standard functions and special services offered only by Linux. With a new chapter on multithreading, this updated and expanded edition provides an in-depth look at Linux from both a theoretical and applied perspective over a wide range of programming topics, including:A Linux kernel, C library, and C compiler overviewBasic I/O operations, such as reading from and writing to filesAdvanced I/O interfaces, memory mappings, and optimization techniquesThe family of system calls for basic process managementAdvanced process management, including real-time processesThread concepts, multithreaded programming, and PthreadsFile and directory managementInterfaces for allocating memory and optimizing memory accessBasic and advanced signal interfaces, and their role on the systemClock management, including POSIX clocks and high-resolution timers

Book Information

Paperback: 456 pages

Publisher: O'Reilly Media; 2 edition (June 8, 2013)

Language: English

ISBN-10: 1449339530

ISBN-13: 978-1449339531

Product Dimensions: 7 x 1.1 x 9.2 inches

Shipping Weight: 1.8 pounds (View shipping rates and policies)

Average Customer Review: 4.4 out of 5 stars   20 customer reviews

Best Sellers Rank: #30,983 in Books (See Top 100 in Books)   #6 inÂ Â Books > Computers & Technology > Operating Systems > Unix   #9 inÂ Â Books > Computers & Technology > Operating Systems > Linux > Programming   #34 inÂ Â Books > Textbooks > Computer Science > Operating Systems

Customer Reviews

Q&A with Robert Love, author of "Linux System Programming, 2nd Edition" Q. Why is your book timely-- what makes it important right now? A. Developing system software on Unix has always been in vogue, but we've seen a large increase in demand with the rise of the cloud and the web. Where before user apps were primarily client-side, UI-focused programs, now system-level software running in huge data centers at massive scale provides even the simplest of our computing functionality. The code powering the largest of cloud-based providers such as Google or Twitter down to the smallest startup is all system-level software. And nearly all of it runs on Linux. Q. What information do you hope that readers of your book will walk away with? A. The system API on Linux: what it is and how to use it as an expert. The book is an API reference manual, of course, but it is also chock-full of anecdotes, insider tips, and Linux-specific techniques that take it beyond your generic Unix API guide. Q. What's the most exciting thing happening in your space? A. Definitely the ever-increasing scale of distributed systems that power the cloud and web apps we all use. I work at Google on web search infrastructure; the scale and scope of our systems is absolutely stunning. And, at the end of the day, it is all just Linux system code running on (a very large number of) Linux machines. Q. What are some of the topics readers will learn in the second edition of Linux System Programming? How to design your multithreaded application for maximum performance How to efficiently perform I/O The pitfalls behind real-time processes How to take advantage of modern hardware such as multicore processors or SSDs Why your program's I/O model dictates its threading model

Talking Directly to the Kernel and C Library

Huge caveat: this book is about application programming, not internal system (kernel) development. Coming from a Windows background I bought this book thinking it would be about writing programs for the system memory space, ie drivers and kernel modifications. That is not the case. In the Linux world "system programming" means anything that makes kernel calls, i.e., uses the system interface, whereas "application programming" is writing scripts. This definition completely differs from that in the Windows/Intel world where "system programming" means writing software that operates at privilege level 0 of the CPU, i.e., anything in the system memory space (usually drivers and various OS components). So, if you are coming from a non-Linux environment be aware of that. For example, the author considers a writing "text editor" to be system programming, whereas in Windows and the MacOS text editors are considered applications and writing them is considered application programming.This book covers all the basic calls in an introductory way. For example, the first chapter with meat in it, Chapter 2, covers "File I/O" and gives beginner level descriptions of calls like read(), seek() and select(). The main advantage of the book is that is pretty thorough in coverage, giving basic descriptions of every major system interface.Overall the book is decent, but is completely outmatched by other similar, much better books. For example, "The Linux

Programming Interface" by Kerrisk has everything in this book plus a lot more and much better examples. In particular a big failing of this book is that is has no realistic examples, just toy snippets. A much better introductory book is "Understanding UNIX/LINUX Programming: A Guide to Theory and Practice" by Bruce Molay which has extensive, realistic examples that do real stuff.If you want to just gloss over Linux programming and get a "feel" for how it works quickly, this is decent book, but for anybody doing serious work there are better options.

As a software engineer who works in a Linux environment, I was happy to find a systems programming reference. This book is a very nice reference, with insights to the kernel-level implementation of many of the various system calls. I highly recommend this book to any software developer performing systems programming in a Linux (or, in general, a Unix) environment.

This book is excellent for learning low-level C programming.It covers most topics of the OS programming (I/O, thread, memory) in concise manner.Unless you will do OS programming for your entire life, I think this book is better than the standard books:The Linux Programming Interface: A Linux and UNIX System Programming HandbookÃ Â by Kerrisk orÃ Â Advanced Programming in the UNIX Environment, 3rd EditionÃ Â by Stevens.They are excellent books, but NOT appropriate for introduction to the topic.Learning from those two books is like learning from encyclopedia.One shortcoming is that it doesn’t contain sockets.If it does, then this book would be my favorite book for systems programming.

Great.

Minor omissions include not mentioning posix_spawn() which eliminates fork() deficiencies, but otherwise it is a solid introduction to system programming under Linux. I like that it is mostly concerned with current state of the art.

I’m 25% in reading and I find it very clear, informative and quite deep.Starting now the process handling section, that I found quite interesting, and will update my review.

Learned a lot.

Great overview. Found I wanted more detail in almost every chapter that I referred to.

Linux System Programming: Talking Directly to the Kernel and C Library The Linux Programming Interface: A Linux and UNIX System Programming Handbook CompTIA Linux+ Powered by Linux Professional Institute Study Guide: Exam LX0-103 and Exam LX0-104 (Comptia Linux + Study Guide) Kernel of the Kernel (Suny Series in Islam) Linux Kernel Development (3rd Edition) Understanding the Linux Kernel, Third Edition Easy Linux For Beginners: A Complete Introduction To Linux Operating System & Command Line Fast! Python Programming: Python Programming for Beginners, Python Programming for Intermediates, Python Programming for Advanced C++: The Ultimate Crash Course to Learning the Basics of C++ (C programming, C++ in easy steps, C++ programming, Start coding today) (CSS,C Programming, ... Programming,PHP, Coding, Java Book 1) CompTIA Linux+ Guide to Linux Certification Assessment, Evaluation, and Programming System for Infants and Children (AEPSÃ Â®), Second Edition, Curriculum for Three to Six Years (AEPS: Assessment, Evalutaion, and Programming System (Unnumbered)) C++ and Python Programming: 2 Manuscript Bundle: Introductory Beginners Guide to Learn C++ Programming and Python Programming C++ and Python Programming 2 Bundle Manuscript. Introductory Beginners Guide to Learn C++ Programming and Python Programming Python Programming: The Complete Step By Step Guide to Master Python Programming and Start Coding Today! (Computer Programming Book 4) A Practical Guide to Linux Commands, Editors, and Shell Programming (3rd Edition) Assembly Language Step-by-Step: Programming with Linux Beginning Linux Programming Hacking University: Learn Python Computer Programming from Scratch & Precisely Learn How the Linux Operating Command Line Works: 2 Manuscript Bundle Get Talking and Keep Talking French Total Audio Course: The essential short course for speaking and understanding with confidence (Teach Yourself) Get Talking and Keep Talking Japanese Total Audio Course: The essential short course for speaking and understanding with confidence (Teach Yourself Language)