Linux Kernel Development

Synopsis

Linux Kernel Development details the design and implementation of the Linux kernel, presenting the content in a manner that is beneficial to those writing and developing kernel code, as well as to programmers seeking to better understand the operating system and become more efficient and productive in their coding. The book details the major subsystems and features of the Linux kernel, including its design, implementation, and interfaces. It covers the Linux kernel with both a practical and theoretical eye, which should appeal to readers with a variety of interests and needs. The author, a core kernel developer, shares valuable knowledge and experience on the 2.6 Linux kernel. Specific topics covered include process management, scheduling, time management and timers, the system call interface, memory addressing, memory management, the page cache, the VFS, kernel synchronization, portability concerns, and debugging techniques. This book covers the most interesting features of the Linux 2.6 kernel, including the CFS scheduler, preemptive kernel, block I/O layer, and I/O schedulers. Features * Authored by a well-known member of the Linux kernel development team with a reputation for a highly readable and focused writing style * Updated and improved coverage of all the major subsystems and features of the latest version of the Linux 2.6.xx kernel, with new coverage of kernel data structures * Allows developers to learn how to modify and enhance kernel code by providing examples based on real kernel code * Details on interrupt handlers and bottom halves * Extended coverage of virtual memory and memory allocation * Information on debugging kernel code * Examples of kernel synchronization and timers * Useful insight into submitting kernel patches and working with the Linux kernel community

Book Information

Paperback: 440 pages

Publisher: Pearson Education; 3 edition (April 5, 2010)

Language: English

ISBN-10: 8131758184

ISBN-13: 978-8131758182

Product Dimensions: 9.2 x 6.8 x 0.6 inches

Shipping Weight: 2.6 pounds

Average Customer Review: 4.6 out of 5 starsÂ Â See all reviewsÂ (102 customer reviews)

Best Sellers Rank: #11,493,896 in Books (See Top 100 in Books) #22 inÂ Books > Computers & Technology > Operating Systems > Linux > Kernel & Peripherals

Customer Reviews

I have been doing Linux kernel/system level development on and off since 1999. This is the book that I think should be owned by any Linux newbie who wants starting their kernel hacking. Even if people do not directly do Linux kernel development, it is a good book complementary to any serious operating systems course in college - it helps gain a better idea of how and why.The book is quite easy to follow and read and does not try to overwhelm readers with tons of information (consequently it does not address many details in Linux kernel). I consider this is a major strength of the book which parts away from other books (comparing to "Understanding the Linux Kernel", which has quite some details on each subsystem, but if you take the book as your guide to kernel programming, you feel you are overwhelmed by the information and often clueless on where to start to write some simple stuffs. This does not mean I think the latter is a bad one - it is a very good one indeed). Considering the fact that Linux kernel evolves so fast, it may make sense to focus on the core parts and once you understand them, it may become easy for you to track and understand changes later. Even as a professional programmer doing kernel development, occasionally referencing a well-written book like this is very helpful.I am a bit reluctant to rate it 5 stars though due to many typos observed, which I guess is the result of rush to publishing (and the poor job of proofreading). Fortuanately, most can be understood by reading the contexts around them. But a few are really misleading or totally wrong. For example, on page 169, there is a sample code to show how page allocation/free is done in kernel. It uses __get_free_pages() to allocate pages, but uses free_pages() to free these pages. As the author has just said a page ago, __free_pages() should be used to free (struct page*) pages, otherwise corruption will ensure (free_pages is used to free pages with logic address as parameter).

I was shopping for a good overview reference book of the Linux kernel, I did not want too much depth into each component, what I wanted was a "brief" overview of all the different components. If you’re looking for depth into each module, then this is not the book for you. If you’re interested in Linux and want a good overview book that you can finish quickly and have a working knowledge of the different components and how they tie in together then this is a great piece. I think "Linux Device Drivers" by Corbet is a better reference if your interest is strictly device driver and "Understanding Linux Networking Internals" by Benvenuti is better if you want to know more about the IP stack. Overall Robert Love goes through kernel development at a great level for an overview with just enough depth and enough examples. I use the book not every day but I often have it on my desk for reference.

I was a Linux kernel newbie writing a device driver and started reading "Linux Device Drivers" by Rubini. On hindsight, this was a bad idea. Rubini's book goes deep into driver code quickly with good details but it only sparingly touches the higher level kernel overview or essential concepts. These missing pieces are covered very well in Love's book and I should have understood them before reading Rubini's book; important basic concepts covered in good detail include: - user thread vs kernel thread. - kernel-space process context vs kernel-space interrupt context. - tasklet as a non-concurrent form of softirq and is not related in any way to tasks. - bottom-half methods comprising softirq, tasklet and work queue; and that BH and task queue are obsolete and deprecated. - semaphore sleeping vs spinlock spinning (busy-wait). - spinlock adversely affecting scheduling latency while semaphore does not.Love's book shows ambly that he is an expert in Linux kernel matters and speaks with authority. At the same time he has the ability of a good teacher to explain obscure and critical kernel concepts clearly. I heartily recommend this as the first book one should read about the Linux kernel, well before books such as Bovet's "Understanding the Linux Kernel" or Rubini's device driver book.This 2nd edition introduces more materials and explanation to cover the updated 2.6 kernel. As far as I can see, it is a worthy new edition to own.

This book is for a reader who is an accomplished C programmer and for someone who wants to learn how to do Linux Kernel Development. The author has been contributing to Linux for more than 15 years and he was a member of the team that developed Android mobile platform's kernel. Although the author explains some of the topics in detail (for example Process Scheduling), he glosses over some of the other topics (for example Process Management). In order to understand some of the theoretical concepts presented in the book, it is better to have a background of Operating Systems. Therefore, it is better to study this book along with a theoretical book on Operating Systems (Silberschatz, Galvin). Having said that, this book can serve as a useful introduction to someone who wants to know the design and implementation of the Linux kernel.In the first few chapters, the author provides instructions for obtaining the Kernel source code and compiling it. In the rest of the chapters, the author gives details of each of the parts of the Linux kernel. In the chapter on Kernel Data Structures (Chapter 6), the author gives a detailed explanation of the most important data structures that are used in Linux (linked lists, queues, maps and red-black trees). The chapter on Debugging (Chapter 18) is full of useful tips for debugging the Linux Kernel. What I like most about the book is that the author is very practical with his approach and concludes his book by saying that "the only way to start (learning the Linux Kernel) is by reading and writing code".

Linux: Linux Command Line - A Complete Introduction To The Linux Operating System And Command Line (With Pics) (Unix, Linux kemel, Linux command line, ... CSS, C++, Java, PHP, Excel, code) (Volume 1) LINUX: Easy Linux For Beginners, Your Step-By-Step Guide To Learning The Linux Operating System And Command Line (Linux Series) Linux Kernel Development Linux For Beginners: The Ultimate Guide To The Linux Operating System & Linux Linux Administration: The Linux Operating System and Command Line Guide for Linux Administrators CompTIA Linux+ Powered by Linux Professional Institute Study Guide: Exam LX0-103 and Exam LX0-104 (Comptia Linux + Study Guide) LINUX Kernel Internals Understanding the Linux Kernel IA-64 Linux Kernel: Design and Implementation learning by doing: LINUX kernel guide(Chinese Edition) Managing the Linux kernel with AgentX: Design and Implementation Using Linux Kernel Version 2.0 to 2.2 (Keeping Ahead) The Linux Kernel Linux Kernel Architecture (Sams White Book Series) Linux Kernel Architecture Understanding the Linux Kernel by Daniel P. Bovet (Nov 24 2005) Understanding the Linux Kernel 3th (third) edition Text Only The Linux Kernel Primer Linux Web Server Development: A Step-by-Step Guide for Ubuntu, Fedora, and other Linux Distributions Smart Home Automation with Linux (Expert's Voice in Linux)