book reviews

ELIZABETH ZWICKY

zwicky@greatcircle.com

with Richard Johnson, Sam Stover, Steve Manzuik, Ben Rockwood, and Ming Y. Chow

THE UNOFFICIAL LEGO BUILDER'S GUIDE

Allan Bedford

No Starch Press, 2005. 319 pp. 1-59327-054-2

The process of selecting books to review is, to put it politely, organic; it involves complex variables such as my level of interest in the topic, my level of knowledge about the topic, my estimation of readers' levels of interest, the other books in the stack, and whether or not I think a book is cool. Which is all by way of saying, no, Lego does not have much to do with advanced computing systems, but I think it's cool, and I'm betting a fair number of you do, too.

This book is cool. It's not rocket science, although there is a nice walk-through of how to design a space shuttle model. It would be a great Christmas gift for the person on your list with the big Lego collection and no very focused idea of what to do with it. You might be more reluctant to give it to anybody in your own household, as the storage suggestions may result in the reader developing entirely new ideas of the scale of a "big" Lego collection, and wanting closets-full. If you already have closets full of Lego, this book will give you the graph paper and the ideas to turn it into Lego cities, or whatever. It's suitable for older kids and young-at-heart adults. And you can feel good about giving it to kids, because it teaches some nice mathematics about ratios, making it genuinely educational.

I learned some neat stuff (the thin Legos are exactly 1/3 the size of normal-height ones), and it's my 18-month-old's second-favorite of the books I've reviewed, because it led me to build things she likes out of her Duplo. (Her favorite is a hardback with a penguin on it. She likes the penguin and finds it an especially intriguing size, for some unknown baby reason.)

THE LINUX ENTERPRISE CLUSTER: BUILD A HIGHLY AVAILABLE CLUSTER WITH COMMODITY HARDWARE AND FREE SOFTWARE

Karl Kopper

No Starch Press, 2005. 430 pp. 1-59327-036-4

Suppose you know not very much about Linux, and less about clusters, and somebody comes to you and says, "Hey, here's a pile of computers; build a cluster out of them, and, oh, by the way, we want to run business-critical software on it." If you sit down with this book and follow it through, at the end, I am convinced, you will have a reasonable solution to that problem. I don't know that it will be the best possible solution; this book walks through one particular set of tools, which undoubtedly won't be the best for every situation. I'm sure that serious Linux cluster aficionados will argue passionately about the author's choices. But there's no avoiding that problem if you want to explain the nuts and bolts of using a particular solution, which the author does very nicely.

The authors take an unusual but effective approach: they walk you through detailed recipes for setting up, not the production environment, but a test environment where you learn how all the parts work and how you can customize them for your purposes. This makes a nice balance between detailed, hand-holding exposition and getting the concepts you need to be able to extend the recipes into your environment.

If you already know something about clusters and Linux (or general UNIX system administration), go straight to chapter 5, bypassing the very general discussion of what a cluster is and a lot of background on kernel builds, SSH installation, rsync, and the like.

WEB MAPPING ILLUSTRATED

Tyler Mitchell

O'Reilly, 2005. 349 pp. 0-596-00865-1

Here's another one I think is cool. (Though it's about maps *on* the Web, not maps *of* the Web, which might have been even cooler.) I like maps, and this book made me want to run right out and add gratuitous maps to my Web site. Better yet, it made me think that the next time I'm on a project where the right thing is to put up an interactive map on a Web server, I will have an answer that doesn't involve all the Web programmers saying glumly, "Gee, that sounds really hard." (That's what happened the last two times, and I didn't get my interactive maps.) True, it wouldn't take 349 pages to explain it if it were really easy, but *Web Mapping Illustrated* tells you how to get and use open source tools to do powerful things with maps, with some basic information on getting and generating the data to go along with the tools. It's enough information to get people past the fear of the unknown.

One caution: it's meant for people who understand maps and want to put them on the Web. It gives some basic background for people who understand the Web but don't know much about maps, but it's probably only enough to make somebody like me able to make real mapmakers writhe in pain. If you want respectable maps, you're going to need either to be very conservative or to get somebody

who knows a lot about maps. This book will take you past the edge of your mapmaking competence and induce the map equivalent of the ransom-note typography that was so popular when word processors first let amateurs play around with fonts. But hey, it's fun to do, even if it's not always fun to watch.

HP-UX 11I VERSION 2 SYSTEM ADMINISTRATION: HP INTEGRITY AND HP 9000 SERVERS

Marty Poniatowski

Prentice Hall, 2005. 643 pp.
0-13-192759-0

If you are an experienced administrator looking for information about HP-UX commands, particularly those specific to HP hardware, you may find some information of interest here. However, the book does not go into enough depth for my taste (it talks about how to use HP's remote install process but not about its underpinnings) and doesn't have enough detail for an inexperienced administrator (it says the author usually modifies the default partition layouts, but doesn't talk about how or why). It is also security-naive; while the author does make some gestures toward security, suggesting that hosts.equiv and .rhosts be used cautiously, he doesn't warn administrators that 6 characters is not a reasonable current minimum password length, that scanning your own network is liable to annoy not just the network administrators but also the security people, that remote SNMP system management has security implications, or that giving nonprivileged users backup and restore privileges has security implications. On the whole, I can't recommend this book. In most situations, you'd be better off with a good, general system administration book and HP's documentation.

PRACTICAL DEVELOPMENT ENVIRONMENTS

Matthew B. Doar

O'Reilly, 2005. 297 pages.
ISBN 0-596-00796-5

Most books that I review will eventually find their way to more appropriate homes. A few I keep a good tight grasp on. This is one of those few. I've put up with a wide range of development environments, and I understand varying parts of them to varying extents. But I don't understand them in the same way that I understand the ins and outs of a data center, for instance. This is a structured overview of all the parts that go into a development environment, with specific examples, comparisons of the good and bad points of common tools, and questions to apply to your own environment. In other words, it's exactly what I need to help me get to the point where I understand development environments as well as I do the system administration environments I've built from the ground up.

It covers software configuration management, build tools, bug tracking, testing, documentation, release, and maintenance, and it gives equal weight to commercial and open source solutions. Its advice is consistent with my experience; yeah, those common problems really are common.

This book will be most useful if you are building a development environment, or if you want to be a toolsmith (somebody who supports programmers directly, working on the tools that let them do development). But if you're just entering the wild and woolly world of programming and you want a scorecard so you can tell the players apart, it'll help you too. And I strongly recommend it for system administrators who support programming teams.

BEHIND CLOSED DOORS: SECRETS OF GREAT MANAGEMENT

Johanna Rothman and Esther Derby

The Pragmatic Bookshelf, 2005.
167 pages. ISBN 0-9766940-2-6

This is a nice, small book on how to be a good manager, aimed at people working in large development environments. Its advice is entirely sensible (that is to say, I agree with it). There is nothing earth-shattering here, but there shouldn't be; good management books agree with each other and say mostly commonsense things that are easy to read and hard to implement. Its most radical move is a good, easy-to-swallow presentation of communication issues: how and, most important, why to say nice things about other people. I think this is an important issue for technical people, who tend to think "communication skills" is a management buzz phrase for "talks nonsense and wears a nice tie," whereas it's actually a management buzz phrase for "not torture to be around."

One of this book's strengths is that it gives nice, concrete examples. This is going to be most useful for people who're working in the sort of corporate product development environment that their examples are drawn from. The concepts are useful anywhere, but if you need the examples, you may find that these don't work as well for you if you're in a different environment.

If you are moving into management and want a short introduction to important management skills that respects technical people and explains things understandably without condescension, this is a nice place to start. You'll probably find yourself consulting some of its many references as you go forward, but just following its advice will go a long way toward making you a productive, useful manager.

AMBIENT FINDABILITY

Peter Morville

O'Reilly, 2005. 188 pages.
ISBN 0-596-00765-5

If the title didn't suggest to you that something was odd about this one, the cover would; the animal on it is in color, but it's otherwise a traditional O'Reilly cover. It's not a traditional O'Reilly book. No, you have not missed the release of a new programming tool called "Ambient" or "Findability." This is a book about the ways in which the ability to find things is changing the world.

It's an amusing and interesting book, and it convinced me that, yes, findability is really important. There's lots of inspiration here for designers of all kinds. At the same time, I found it ultimately frustrating. It feels like there ought to be a deep structure and some fundamental insights, but all I got was a bunch of neat stuff.

This book is definitely a good time, and it should be particularly enlightening to people just outside the world of the Web, or just entering it. It's full of pretty pictures and clever ideas; if you know somebody intelligent who doesn't understand why the Web really, really matters, this book should get the point across.

DATA PROTECTION AND INFORMATION LIFECYCLE MANAGEMENT

Tom Petrocelli

Prentice Hall, 2005. 256 pages.
ISBN 0-13-192757-4

This book covers data protection, starting from the types of data storage (everything from good ol' disks flung in a server through SAN and NAS, with explanations of SCSI and ATA and all of their cousins), through backup and restore, data replication, security, policies, and, finally, a brief flourish about managing not just data but information. That's a lot of stuff to try to get into 256 pages, and, in the end, I don't feel that it all fit well enough.

Most topics are covered only at a high, abstract level, which makes them hard to understand and apply. Furthermore, there are some odd omissions. For instance, in the chapter on backups, a number of failure modes are mentioned, but there's no mention of backup verification or testing, which is an obvious and important part of data protection. In the chapter on storage systems, there's no mention of RAID 4. In other places, you could be led to dangerously wrong conclusions: on-disk data encryption is not a panacea, and being able to back up open files is useful only if you have some reason to believe they're consistent enough to be usable when you restore them.

HOST INTEGRITY MONITORING WITH OSIRIS AND SAMHAIN

Brian Wotring and Bruce Potter (technical editor)

Syngress, 2005. 420 pages.
ISBN 1-597490-18-0

REVIEWED BY
RICHARD JOHNSON

rjohnson@ucar.edu

The title of this book might lead you to expect a how-to manual for building and operating Osiris and Samhain. It does indeed contain such, but the book is far more useful than that. It's also intended as a "why" manual which starts by helping you answer the very basic question of whether you, on your particular machines, even need or want to use host integrity monitoring. Beginning with the why portion, the first four chapters concisely but not too choppily define integrity monitoring, describe what typically needs to be watched, cover typical attacks and the changes they'll produce (with examples of automated worms), and delve into the planning crucial for setting and meeting your specific monitoring goals. The next three chapters get into the how-to of installing and operating host integrity monitoring software, with a chapter each dedicated to Osiris and Samhain. Finally, the book covers stepping beyond the simple change notification facilities built into each of the systems, responding to incidents detected by the system, and more advanced countermeasures or pitfalls. The book flows well from chapter to chapter, particularly with the summaries at the end of each, which, somewhat amusingly, turned out to mirror my personal notes. I found it easy to read with comprehension from cover to cover.

Although the book's target audience consists of experienced system and security administrators (call it SAGE II+), the first half is also useful for technically inclined managers. It's a nice design, as it gives those of us in charge of the implementation a solid hook for bringing our superiors up to speed. Also in this portion, the discussion of what to monitor was particularly valuable. Even with 19 years of experience as a sysadmin, I gained new insight from the discussion of where (possibly malicious) changes can hide in various OSes. More important, the foundation here makes extrapolation to OS features that weren't in use before the book was written (e.g., Mac OS 10.4's new use of arbitrary file metadata in HFS+, which can be used to hide data) almost inevitable for a technical reader.

The build chapters don't rehash the man pages or release documentation for Osiris or Samhain. Instead, they evenly cover the strengths and weaknesses of each package, followed by build and installation tutorials with a clear eye to avoiding later management problems and mitigating security risks. Even picking a system that I don't normally deal with—building and installing an Osiris client on MS Windows—I found it easy to follow the instructions. Further-

more, the Samhain build chapter was met with, "Nice, I didn't know that before, I'll have to add that," from a fairly experienced Samhain user.

A minor downside was that PGP signature verification instructions for the source code distributions in each of the build chapters were redundant. Particular advice for setting ultimate trust on a PGP key in Chapter 6, which left me feeling uneasy, was later qualified with a caveat in Chapter 7, but otherwise the sections might better have been consolidated. Somewhat more annoyingly (even though I understand why it is this way; it's another book's worth of material in itself), it would have been nice to see more advanced material on log monitoring, focusing on additional tools that can help us intelligently aggregate and process the tagged syslog output or database entries, specifically from Osiris and Samhain.

In the end, the book carries readers along, educating them and leaving them wanting more (with an idea of where to go to get that more). If you're thinking of trying host integrity monitoring, though without the noise and maintainability problems common to such systems in the 1990s, this book will serve you well. More important, this book will help you figure out why you want to monitor host integrity in the first place, and then tune what you monitor to meet your goals.

FILE SYSTEM FORENSIC ANALYSIS

Brian Carrier

Addison Wesley, 2005. 569 pages.
ISBN 0-321-26817-2

REVIEWED BY SAM STOVER

sam.stover@gmail.com

I think this book is hands-down the best resource for file systems (FAT, NTFS, EXT2/3, and UFS1/2) and partition types (DOS, Apple, BSD, GPT, and Solaris Slices) I've read. It is not, however, designed as an introductory guide for a novice forensic analyst. The author does not focus on walking the reader through evidence handling, chain of custody, etc., nor does he focus on tutoring the reader in the use of common forensics tools such as EnCase. The goal of this book is to provide a foundation for a forensics investigator to work from, and I think it achieves that goal. And, as mentioned, it serves as a great reference for anyone doing any kind of file system and/or partition work.

A lot of the information in this book is available in other forms such as RFCs, vendor standards, etc., but this book brings everything together in one place. The book starts by giving a brief overview of the principles involved in digital forensics investigation, but then moves quickly into a low-level discussion of the different types of partitions used. From there, the aforementioned file systems are examined in intimate detail.

Prior to reading this book, I felt that I had a pretty good grasp of the different file systems and how they are put together. After going through the NTFS chapters, I soon realized how much I didn't know, and I suspect that a lot of forensic investigators fall into the same trap. Current forensic tools do a lot of the heavy lifting with respect to file system analysis, and thus they make it too easy to conduct an investigation without completely understanding everything from the ground up.

In the way that W. Richard Stevens has provided us with an invaluable reference for the TCP/IP protocol stack, Carrier has given us an analogous reference for partitions and file systems. To further the analogy, some of the material in this book is very complex and could require a fair bit of effort from the reader to fully grok the file system or partition in question.

One last comment is that the author's toolkit, called The Sleuth Kit (TSK), is used throughout the book to demonstrate the examples. I stated earlier that it was not a goal of this book to tutor a user on a particular forensic tool, and I stand by that. This book does not teach you how to use TSK, but there is a seven-page appendix that gives you the basics so that you can use the tool yourself to emulate what's happening in the book.

Overall, this book is definitely a "must have" for anyone who wants to learn how file systems work—whether that is to be applied to forensic analysis or otherwise. It will occupy a space on my bookshelf right next to *TCP/IP Illustrated*, and will probably be referenced just as frequently.

ROOTKITS: SUBVERTING THE WINDOWS KERNEL

Jamie Butler and Greg Hoglund

Addison-Wesley, 2006. 324 pages.
ISBN 0-321-29431-9

REVIEWED BY STEVE MANZUIK

hellnbak@gmail.com

As someone who has been called a Windows security expert, I always find it a pleasure to come across a technical book that covers subject matter that is perhaps a little less known than your standard Windows security concepts. It is even a greater pleasure to find that same book is able to teach even an experienced security geek such as myself a few new tricks.

Most people, and when I say most people I am referring to myself, use technical books as a way to cure insomnia, while occasionally learning something along the way. Although *Rootkits: Subverting the Windows Kernel* is high on technical content, I found myself doing more learning than sleeping. When I first sat down to read

Rootkits, it was 2 a.m. and I was ready for something to put me to sleep. Instead, I found myself hopping out of bed to grab my laptop and make note of some of the techniques taught in the book.

The concept of a rootkit has been around for a very long time, especially in the *NIX world. Over the years we have seen them evolve from lame hacker tricks to more in-depth, harder-to-detect subversion methods. I have had many different roles in my career, but before landing in the eEye Digital Security Research Department I was an independent security consultant. I performed many incident response engagements for clients, which usually involved some sort of Linux rootkit installed on compromised systems. Butler and Hoglund have taken the concepts of the "old school" rootkits and applied them to the "new school"—Microsoft Windows. So if you are a Windows person, or interested in the Windows kernel, this is a book for you. Be sure to also check out the accompanying Web site (http://www.rootkit.com): you will find all kinds of samples used in the books and a great discussion forum where you can exchange ideas with the authors as well as with other security geeks.

Butler and Hoglund take the reader through the technical details of Windows rootkits, sparing nothing. This book is filled with useful information that will help reader understand exactly what a rootkit is, how they are used, and how to create your own rootkits that can subvert various detection routines. Of course, this book would not be complete without information on how to build a good host-based intrusion prevention system to resist such attacks.

Whether you are a junior security person or one of the old-timers in the industry, I highly recommend this book to you. I have spent a little over a decade in the IT and IT security industry, and I found this book complete enough to leave some new knowledge in my brain.

Oh, and if you are having trouble sleeping, fire me off an email and I can give you a list of the books in my library that actually do help put one to sleep.

ORACLE PL/SQL FOR DBAS

Arup Nanda and Steven Feuerstein

O'Reilly, 2005. 429 pages.
ISBN 0-596-00587-3

REVIEWED BY
BEN ROCKWOOD

benr@cuddletech.com

Oracle PL/SQL for DBAs is O'Reilly's latest addition to its ever growing series of Oracle PL/SQL books, most of which are written by or co-authored with guru Steven Feuerstein. Unlike previous titles, such as *Learning Oracle PL/SQL* or *Oracle PL/SQL Programming*, this book is squarely aimed at experienced DBAs looking to better leverage capabilities of the database through PL/SQL interfaces. For the sake of completeness, the first chapter contains a whirlwind tour of PL/SQL from the ground up, but readers new to PL/SQL will find themselves lost in the dust.

The book focuses on three main topics: security, performance, and scheduling. In the performance category is in-depth discussion of cursors and table functions. Both chapters are chockfull of solutions and ways to better craft your queries and write your procedures, but little in the way of theory is offered. Security is covered by discussion of encryption within the database, auditing, row-level security, and a great chapter on generating random values. Scheduling is handled in a single chapter but supplies a great deal of insight on the topic.

Again, this is a book for experienced DBAs. It answers the question "How?" but not the question "Why?" Almost no background on cursors is given, for instance, making you reach for Google or your favorite DBAs reference. And the introduction to encryption is good for a laugh but little more. Clearly the authors have a talent for demonstrating functionality but have left background explanations for other, more suitable guides.

Perhaps the book's most redeeming value is the sense it provides of just how much can be done from within Oracle itself. The scheduling chapter, for instance, would be of great use to DBAs who have become too reliant on cron. While DBAs and sysadmins won't need every feature outlined in this book, there is clearly value in realizing what's available and using that information to better leverage your existing deployments.

PRIVACY: WHAT DEVELOPERS AND IT PROFESSIONALS SHOULD KNOW

J.C. Cannon

Addison-Wesley, 2005. 347 pages.
ISBN 0-321-22409-4

REVIEWED BY
MING Y. CHOW

mchow@eecs.tufts.edu

Cannon delves into all facets of privacy, low-level and high-level. As indicated by the subtitle, the book targets developers and IT professionals, but I would recommend it for end users and managers as well. The first half provides a very comprehensive overview of privacy. Privacy-Enhancing Technologies (PETs) and Privacy-Aware Technologies (PATs) are presented, with numerous examples and features. Cannon also discusses privacy frameworks and legislation, including the Health Insurance Portability and Accountability Act (HIPPA), the Gramm-Leach-Bliley Act (GLBA), and even the Digital Millennium Copyright Act (DMCA).

The privacy issues with spam and emerging technologies (e.g., RFID tags), including solutions to mitigate privacy risks, are discussed in perceptive detail.

For developers, Cannon provides rich insights and effective techniques for incorporating privacy into both the development process and the products themselves, including discussions of privacy analysis, privacy specification, dataflow diagramming, and database protection. He presents a large-scale example of development from top to bottom. There are even checklists and templates that managers and developers can use immediately.

The public's growing concern about privacy is well founded, and they are demanding that government, business, and developers step up their efforts to preserve privacy. Cannon does a tremendous job of stressing the importance and value of integrating privacy into products and in business and, more important, in explaining how to do so. Throughout the book, he emphasizes trust, incorporating privacy into the development process early, and enabling users to control their own privacy. It is a huge and important challenge today to give end users security they can understand and privacy they can control. I recommend this book without reservation to those who want a competitive edge in this dire field.