book reviews

ELIZABETH ZWICKY, WITH SAM STOVER AND EVAN TERAN

HOST INTEGRITY MONITORING USING OSIRIS AND SAMHAIN

Brian Wotring

Syngress, 2005. 399 pages.
ISBN 1-597490-18-0

Host integrity monitoring is not the sexy part of computer security. That would be network intrusion prevention, because networks are cooler than hosts, intrusions are cooler than integrity, and prevention is way cooler than monitoring. But host integrity monitoring has in general a strongly positive effectiveness/annoyance ratio, for some of the same reasons that it's not sexy. For instance, most of the time what it does is detect stupidity rather than malice; it's really good at tattling on authorized users who are doing things they ought not to. Fortunately for us all, your average network suffers more from stupidity than from malice.

So if you run a network of any size, you ought to be doing host integrity monitoring. It won't keep intruders out or make your hair shinier, but it will find intruders and make your site tidier. This book goes over why you ought to do host integrity monitoring, what kinds of things you need to monitor and why, and how to install and configure the most popular open source monitoring systems.

The author clearly has experience with monitoring systems and with practical security. He advocates for configurations that are as secure as practical, but no more so, with the authentic weariness of the person who has seen rookie administrators either try to do it "Right! At all costs!" or decide that all this security stuff is too much hassle anyway, and you might as well build your security products just like any other package.

Warning for ex-proofreaders and other people inclined toward pickiness: This book is edited to the usual Syngress standards, which is to say there's an error or two per page. The content is fine, but the itch to red-pen things gets overwhelming.

TELLING AIN'T TRAINING

Harold D. Stolovitch and Erica J. Keeps

ASTD Press, 2002. 189 pages.
ISBN 1-56286328-9

Training, like documentation, is one of those tasks that programmers and system administrators end up with for a number of reasons: The organization is too small to need a dedicated person, everybody is so accustomed to seeing it done badly that they don't realize how much better it could be done, or the managers are cynics who believe it's all pointless anyway, so they assign it to whoever complains about the lack as a punishment for complaining.

If you find yourself wanting to train people more effectively, I recommend this book. It captures the essence of modern training—which, like all good things, is simple in concept, difficult to implement—and communicates it vividly and practically. You will find good advice on how to decide what to cover, how to lay out the course, and what educators currently know about education and the brain. (As they point out, common sense is not your best guide to figuring out what works, and tradition is a whole lot worse.)

Do not run their advice on French verbs by your French teacher, however; I kept waiting for them to point out that oversimplification is a valid educational technique and they were doing it on purpose, and they never did. (There is more than one pattern for regular verbs in French. They classify regular verbs in -ir and -re as irregular. I believe we had already established my pedantic tendencies.)

IMPLEMENTING ITIL CONFIGURATION MANAGEMENT

Larry Klosterboer

IBM Press, 2007. 216 pages.
ISBN 0-13-242593-9

This is a book aimed at large sites, the kind of place that has an IT department that cannot all sit around the same table and eat pizza together, even when they rent the whole restaurant. Since I work for the kind of site where the IT department is measured in fractions of a person, it's not of immediate application for me. However, I've worked for enough large organizations to recognize the authentic voice of experience.

Only somebody who knows what he's talking about would address the question of how to do configuration management on a machine that nobody knows how to get into, but that can't be turned off. He gives the right answer, too: First, figure out if you can stick your fingers in your ears and pretend it doesn't exist. That's what everybody else is doing; why should you get the hot potato? Failing that, try very hard to eliminate it, because it is probably cheaper to replace it than to manage it.

If you need to implement IT Infrastructure Library (ITIL) configuration management, you definitely want this book. It assumes that you know something about ITIL but nothing much about running a large IT project. It's actually a nice overview of how to run any such project. The language is stuffier than it needs to be in places, reducing the readability, but the content is straightforward and practical and steers you through many of the minefields of big IT projects. For instance, it lays out clearly how to set up a pilot project—and what to do to save the entire project if the pilot fails.

MINDSET: THE NEW PSYCHOLOGY OF SUCCESS

Carol S. Dweck

Ballantine Books, 2006. 268 pages.
ISBN 978-0-345-47232-8

Yes, it's a second nontechnical book, or, rather, it is mostly a technical book, but not in computer science.

This book is about the difference between a fixed mindset—a belief that how you do at most things is primarily determined by the talents and interests you're born with—and a growth mindset—a belief that how you do at most things is primarily determined by how much and how well you work at them. It provides both anecdotes and research supporting the conclusion that the fixed mindset is neither true nor useful. That is, success has a lot more to do with effort than talent, and the belief that talent is the major controlling factor is harmful.

Why would I bring this up? The fixed mindset is the dominant paradigm in our culture, broadly defined, but it's also the dominant paradigm in the computing subculture more narrowly defined. I've recently seen discussions about whether you can teach people problem solving (yes, you can, and yes, it is partly about intelligence, but you can teach that, too), about whether programmers can be good system administrators and about whether Microsoft system administrators can be good UNIX system administrators. All of these "Can people in specialty X be good at specialty Y" questions presuppose a fixed mindset. These are nonsensical questions from a growth mindset. They also presuppose that all programmers, all Microsoft system administrators, and so on are somehow inherently shaped for their jobs; I can't decide whether this is because people think everybody gets the job they were born to, which you'd think most people would realize didn't work for them, or because they think some things warp your brain if you ever accept money for them. Either way, it seems a peculiar belief to me.

I could expound at length on these subjects. I have before and I probably will again. But it would save me a lot of trouble if people would just go out and read this book.

SECURITY POWER TOOLS

Bryan Burns, Jennifer Grannick, Steve Manzuik, and many others

O'Reilly Media, 2007, 570 pages.
ISBN 13: 978-0-596-00963-2

REVIEWED BY SAM STOVER

One of the great things about doing book reviews is that once I run out of books that interest me, I start looking at books that I might otherwise overlook. Being in the security field, I'm almost embarrassed to say that I would have overlooked this book—and that would have been a shame. The point is summed up nicely in the foreword, where the big question of "Why write this book?" is discussed. Why indeed? There are plenty of other books on Metasploit, Nessus, Nmap, etc. etc. Does the world need another one? Yes, it does. Not only is this book the best "all-in-one" compilation of all the important security tools, it shows you how to use them. No, I mean really *use them*.

Before we get into the usage, I want to take a second to specifically mention Chapter 1, "Legal and Ethics Issues." I can't quite say that this chapter is worth the price of admission alone, but it's close. It's the least technical chapter in the book, but arguably the most important, and certainly the one you need to read first. It was written by a real lawyer about real legal issues that arise in the security world. If you don't understand the potential ramifications of your actions, then you should get out of the sandbox. Finally, we get some substance instead of the usual "We are not lawyers, so don't do anything stupid" disclaimer so prevalent in security books. Kudos to the authors—this chapter has been a long time in coming.

Now that you've read just enough legalese to make you nervous, let's get into the tech stuff. The book is broken into seven sections, ranging from reconnaissance to exploitation to mitigation. There's even a dash of forensics in there too—this book really does cover all the bases. As I mentioned before, the usual suspects are discussed in great detail, as are some newer tools, particularly in the wireless reconnaissance and penetration chapters. One thing that I truly appreciated was the constant stream of references to other works. This book might touch on pretty much everything from rootkits to firewalls, but the authors acknowledge many other specialized books that pick up where this one leaves off. This is just another example of the classiness that is the hallmark of O'Reilly. In that vein, grammatical errors, typos, and cut-and-paste hacks were all absent, as usual. There might have been twelve self-acknowledged "nonwriters" authoring the chapters, but each topic reads cleanly and professionally.

One problem inherent with this type of book is the expectation and experience level of the reader. Some veterans might be a little frustrated with the amount of introductory material, but I think this was kept to a minimum. The book is accessible to the novice but tailored for the tech-savvy. Simply put, I would recommend this book to anyone who needs a great all-around reference and is concerned with getting stuff done. There are plenty of networking topics for the system admins and vice versa. Since the writers and editors did such a great job writing for a highly technical—but not necessarily security-minded—audience, anyone with a good technical background should snap this book up as soon as possible.

HACKING: THE ART OF EXPLOITATION, 2ND EDITION

Jon Erickson

No Starch Press, 2008. 488 pages.
ISBN 978-1-59327-144-2

REVIEWED BY EVAN TERAN

This is a good book. It does a great job of first establishing the mindset of a hacker and then walking the reader step by step through the various techniques of finding interesting ways to solve problems. This in itself is what the author claims is the defining characteristic of a hacker, and I agree.

Unlike the first edition, this book spends about 100 pages explaining what programming is, first with pseudo-code giving very general examples, then working through all the major features of the C programming language, and finally how this all ties to assembly language for the x86 processor. Although I feel this is all requisite knowledge for the subject, if you are already a competent C programmer you are likely going to want to skip chapter "0x300" altogether, especially if you are also comfortable with x86 assembly.

In addition to the general programming overview, the networking chapter has been expanded to include much more information about programming using the BSD sockets API. Once again, if you are already familiar with programming sockets, you may want to skip this chapter. However, since not every programmer writes networked applications, this was a valuable addition to the book.

All of the basics for program exploitation are explained well. Everything from the commonplace stack-based buffer overflow to how to make a format string vulnerability corrupt memory in a useful way is covered. The explanations and examples of the various techniques have been improved since the first edition and use more realistic code samples.

There is also a completely new chapter, "Countermeasures," which goes over ways that a hacker can get around some of the basic preventives a programmer might use. These techniques, ranging from trivial to modern, include logging, byte restrictions, stack randomization, and nonexecutable stacks. Basically this chapter is a way of addressing many of the newer techniques that would prevent the book's examples from working on a real-world machine.

For example, pretty much all of the stack exploit examples earlier in the book assume that the stack is located at the same location for each process. In fact, the LiveCD has ASLR (Address Space Layout Randomization) disabled in order to make this work. One thing that I didn't care for is that this was mentioned as an afterthought. In addition to that, the solution of bouncing off the linux-gate is incomplete, since it does not work in the newest Linux kernels. The author does suggest another solution, which works OK (as in, not all the time, but once is enough).

Personally, if someone asked me to recommend a book on exploitation and they had no knowledge of programming, I would probably recommend they start with a book that focuses specifically on C first. In my opinion, program exploitation is an advanced programming topic that will do nothing but frustrate beginners. Overall, I would recommend this book to those who are competent C programmers, with the caveat that they may want to either skim or skip the early background chapters.