FreeBSD and OpenBSD are increasingly gaining traction in educational institutions, nonprofits, and corporations worldwide because these systems provide significant security advantages over Linux. Although a lot can be said for the robustness, clean organization, and stability of the BSD operating systems, security is one of the main reasons system administrators use these two platforms.

There are plenty of books to help you get a FreeBSD or OpenBSD system off the ground, and all of them touch on security to some extent, usually dedicating a chapter to the subject. But because security is commonly named as the key concern for today’s system administrators, a single chapter on the subject can’t provide the depth of information you need to keep your systems secure.

FreeBSD and OpenBSD are rife with security “building blocks” that you can put to use, and Mastering FreeBSD and OpenBSD Security shows you how. Both operating systems have kernel options and filesystem features that go well beyond traditional Unix permissions and controls. This power and flexibility is valuable, but the colossal range of possibilities creates a need for step-by-step instructions. This book walks you through the installation of a hardened operating system, the installation and configuration of critical services, and the ongoing maintenance of your FreeBSD and OpenBSD systems.

Using an application-specific approach that builds on your existing knowledge, this book provides sound technical information on FreeBSD and OpenBSD security with plenty of real-world examples to help you configure and deploy a secure system. By imparting a solid technical foundation as well as practical know-how, it enables administrators to push their server’s security to the next level. Even administrators in other environments—such as Linux and Solaris—can find useful paradigms to emulate.

Written by security professionals with two decades of operating system experience, Mastering FreeBSD and OpenBSD Security features broad and deep explanations of how to secure your most critical systems. Where other books on BSD systems help you achieve functionality, this book will help you secure your deployments more thoroughly.
Mastering FreeBSD and OpenBSD Security

FreeBSD and OpenBSD are increasingly gaining traction in educational institutions, nonprofits, and corporations worldwide because these systems provide significant security advantages over Linux. Although a lot can be said for the robustness, clean organization, and stability of the BSD operating systems, security is one of the main reasons system administrators use these two platforms.

There are plenty of books to help you get a FreeBSD or OpenBSD system off the ground, and all of them touch on security to some extent, usually dedicating a chapter to the subject. But because security is commonly named as the key concern for today’s system administrators, a single chapter on the subject can’t provide the depth of information you need to keep your systems secure.

FreeBSD and OpenBSD are rife with security “building blocks” that you can put to use, and Mastering FreeBSD and OpenBSD Security shows you how. Both operating systems have kernel options and filesystem features that go well beyond traditional Unix permissions and controls.

This power and flexibility is valuable, but the colossal range of possibilities creates a need for step-by-step instructions. This book walks you through the installation of a hardened operating system, the installation and configuration of critical services, and the ongoing maintenance of your FreeBSD and OpenBSD systems.

Using an application-specific approach that builds on your existing knowledge, this book provides sound technical information on FreeBSD and OpenBSD security with plenty of real-world examples to help you configure and deploy a secure system. By imparting a solid technical foundation as well as practical know-how, it enables administrators to push their server’s security to the next level. Even administrators in other environments—such as Linux and Solaris—can find useful paradigms to emulate.

Written by security professionals with two decades of operating system experience, Mastering FreeBSD and OpenBSD Security features broad and deep explanations of how to secure your most critical systems. Where other books on BSD systems help you achieve functionality, this book will help you secure your deployments more thoroughly.
Mastering FreeBSD and OpenBSD Security
### Other resources from O’Reilly

<table>
<thead>
<tr>
<th>Related titles</th>
<th>Linux Security Cookbook</th>
<th>Security Warrior</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Linux Server Security</td>
<td>Network Security Tools</td>
</tr>
<tr>
<td></td>
<td>Network Security Hacks</td>
<td>Snort Cookbook</td>
</tr>
</tbody>
</table>

**oreilly.com**

*oreilly.com* is more than a complete catalog of O’Reilly books. You’ll also find links to news, events, articles, weblogs, sample chapters, and code examples.

**oreillynet.com**

*oreillynet.com* is the essential portal for developers interested in open and emerging technologies, including new platforms, programming languages, and operating systems.

**Conferences**

O’Reilly brings diverse innovators together to nurture the ideas that spark revolutionary industries. We specialize in documenting the latest tools and systems, translating the innovator’s knowledge into useful skills for those in the trenches. Visit conferences.oreilly.com for our upcoming events.

**Safari Books Online**

*Safaribooksonline.com* is the premier online reference library for programmers and IT professionals. Conduct searches across more than 1,000 books. Subscribers can zero in on answers to time-critical questions in a matter of seconds. Read the books on your Bookshelf from cover to cover or simply flip to the page you need. Try it today with a free trial.
# Table of Contents

## Preface

## Part I. Security Foundation

1. **The Big Picture**
   - What Is System Security?  4
   - Identifying Risks  6
   - Responding to Risk  19
   - Security Process and Principles  23
   - System Security Principles  26
   - Wrapping Up  30
   - Resources  31

2. **BSD Security Building Blocks**
   - Filesystem Protections  33
   - Tweaking a Running Kernel: sysctl  44
   - The Basic Sandbox: chroot  53
   - Jail: Beyond chroot  60
   - Inherent Protections  67
   - OS Tuning  70
   - Wrapping Up  73
   - Resources  73

3. **Secure Installation and Hardening**
   - General Concerns  75
   - Installing FreeBSD  87
   - FreeBSD Hardening: Your First Steps  93
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Securing a Loghost</td>
<td>378</td>
</tr>
<tr>
<td>logfile Management</td>
<td>381</td>
</tr>
<tr>
<td>Automated Log Monitoring</td>
<td>386</td>
</tr>
<tr>
<td>Automated Auditing Scripts</td>
<td>392</td>
</tr>
<tr>
<td>Wrapping Up</td>
<td>394</td>
</tr>
<tr>
<td>Resources</td>
<td>394</td>
</tr>
<tr>
<td><strong>11. Incident Response and Forensics</strong></td>
<td>396</td>
</tr>
<tr>
<td>Incident Response</td>
<td>396</td>
</tr>
<tr>
<td>Forensics on BSD</td>
<td>402</td>
</tr>
<tr>
<td>Digging Deeper with the Sleuth Kit</td>
<td>408</td>
</tr>
<tr>
<td>Wrapping Up</td>
<td>412</td>
</tr>
<tr>
<td>Resources</td>
<td>413</td>
</tr>
<tr>
<td><strong>Index</strong></td>
<td>415</td>
</tr>
</tbody>
</table>
Preface

FreeBSD and OpenBSD are often considered the “other” free operating systems besides Linux. However, in recent Netcraft surveys, the five most reliable web sites on the planet run FreeBSD. OpenBSD, too, is deployed on thousands of security servers around the world. These two BSD-based operating systems are rapidly gaining traction in educational institutions, non-profits, and corporations worldwide.

Plenty of books exist to help you get a FreeBSD or OpenBSD system off the ground. All of them touch on security, but most only dedicate a chapter to it. In sharp contrast, we think it’s worth spending an entire book on the subject. FreeBSD and OpenBSD are rife with security “building blocks” that you can use to really take security and “kick it up a notch.”

These operating systems have kernel options and filesystem features that go well beyond traditional Unix permissions and controls. This power and flexibility is valuable, but the colossal range of possibilities will leave you dizzy if you don’t take things one step at a time. Mastering FreeBSD and OpenBSD Security complements existing books on FreeBSD and OpenBSD administration. Where others help you achieve functionality, we help you build security-minded deployments. This book walks you through the installation of a hardened operating system, the installation and configuration of critical services, and ongoing maintenance of your FreeBSD and OpenBSD systems.
Audience

This book is written by system administrators for system administrators. If you’re looking for a complete idiot or dummy guide, this book is not for you. We’re talking to administrators who have installed a Unix-like operating system before. Almost any will do, but this book is all about what sets FreeBSD and OpenBSD apart from other Unices. You’ll get the most out of this book if you’re comfortable administering BSD operating systems and want to take your experience one step farther.

Administrators at various skill levels and in organizations of any size can benefit from secure BSD systems. Junior administrators who know how to get a Unix system off the ground can use this book to develop a sound foundation in systems security. Experienced administrators, like experienced cooks, will find new recipes that they can add to their existing repertoire. If you’re part of (or all of) a small staff that runs only a handful of servers, you’ll see how choosing one of the BSDs can let you spend less time on security concerns and more on your other duties. If you’re part of a large staff running many servers, you’ll see how BSD servers can be solid pillars in your infrastructure. They’re easy to deploy and scale, and maintaining them is a breeze. Securing them is easy enough, too, with the help of this book.

Assumptions This Book Makes

We’re really focused on improving the skill set of an established system administrator, so we aren’t going to explain a lot of basics. We assume you can find your way to a command line and work your way through the filesystem with speed and grace. We expect that you already have a solid understanding of basic Unix permissions, are comfortable installing and configuring hardware and software, and so on.

If at any time you feel you’re in over your head, fear not. Both operating systems have strong followings and easy to find documentation for all the basics. You can look at FAQs, HOWTOs, and handbooks online, or you can buy one of the many good references in print. The “Resources” section at the end of every chapter always lists good resources that provide additional coverage of relevant topics. In many cases, these additional resources provide the foundation in the technology you need to leverage the recommendations in this book.

The Internet is everywhere, and every administrator needs a basic understanding of local- and wide-area networking. We’re not going to tell you what TCP/IP is, how DHCP works, or how to cable up your switches and hubs. We’ll explain what you need to know when we get into a security topic that is rooted in the deep, dark corners of a protocol specification or some other relatively obscure topic. Network security and configuration are important, but we assume you’ve already got that under control.
Contents of This Book

We’ve tried to break the book up into three sections. We begin by establishing a foundation in FreeBSD and OpenBSD, move on to discuss specific deployment scenarios based on this foundation, and we wrap up with a broader look at these operating systems in your existing network.

Part I: Security Foundation

The goal of Part I is to give you the foundation for building and running secure systems with FreeBSD or OpenBSD.

Chapter 1 is an introduction to system security and general security topics that are relevant to the rest of our discussion. It tells you what you’re up against and gives you some ideas about how we’ll approach securing systems.

Chapter 2 is all about the fundamental building blocks you get for securing systems based on either OpenBSD or FreeBSD. There are some differences, so we highlight those as we go. We cover filesystem features, kernel features, inherent operating system features, and tweaking your kernel to enhance specific security postures.

Chapter 3 augments what you already know about installation. We explore the security-related options, trade-offs, and configurations you must consider when installing. We walk through installing both FreeBSD and OpenBSD, but dwell mainly on areas where choices at installation time can have important security ramifications.

Chapter 4 is a tour de force of administration concerns. You’ve got it installed, you’re running it day-to-day, so now what? We describe controlling access, installing and upgrading software, network security, backups, and system monitoring.

Part II: Deployment Situations

Every server has a specific purpose in life, and FreeBSD and OpenBSD systems are ideal candidates for handling critical infrastructure services like DNS servers, firewalls, mail gateways, and web servers. Part II covers these deployments and how you can leverage specific BSD features to improve the security posture of the services you provide. We don’t tell you everything about deploying the specific service, however; just the extra options and special circumstances where you can take advantage of OpenBSD or FreeBSD. The goal of this section is to offer guidelines for securely deploying the software that will run critical services in your network.

With each of these critical network services, we take time to explain the kinds of risks you face, the sorts of attacks you might need to repel, and why you and your organization care about running the service securely. When we talk about installing and configuring software, though, we refer back to the general techniques and
building blocks that we laid out in Part I. You’ll want to be at least passingly familiar with the techniques, because we combine them in interesting and sometimes subtle ways.

Chapter 5 describes DNS and how to build a secure DNS server. DNS is critical to every Internet service, and getting it right is fundamentally important, so we cover it first. We talk about both BIND and djbdns and how they can be installed, configured, and operated securely.

Chapter 6 covers mail: arguably the most critical electronic communication you support in your organization. We discuss setting up a secure mail architecture as well as filtering and rejecting unwanted mail. We describe both Sendmail and Postfix and how to securely install, configure, and administer them.

Chapter 7 offers a wealth of information on securing Apache-based web servers. We cover risks and threats, configuration and installation, and managing what options your users can set. We also describe thttpd, a small, fast, no-frills web server that can perform admirably in certain situations. In the end we talk about some interesting combinations of FreeBSD’s jails and web servers to isolate and contain lots of web sites in their own sandboxes.

Chapter 8 is about building firewalls. OpenBSD and FreeBSD make excellent choices as firewall platforms. Getting a firewall operational isn’t too hard, but making sure that it’s appropriately secured needs to be done carefully. In this chapter, we’ll talk about ipfw on FreeBSD and pf now available on both platforms.

Chapter 9 outlines the topic of intrusion detection system (IDS) on FreeBSD or OpenBSD. We cover the purposes for using IDSes as well as alternative approaches such as log analysis and intrusion prevention. We give you some good guidance on how to build an effective architecture and monitor it for nefarious activity.

**Part III: Auditing and Incident Response**

Auditing and incident response are topics in system administration theory that are critical but often overlooked. They are not specific services that you run as much as concerns you keep in the back of your mind all the time.

Chapter 10 talks about managing the audit trails. A properly configured system should be warning you about suspicious activity, but how do you manage all the alerts and warnings? We talk about what you want to log, how you can log it securely, and how to manage the logs you generate.

Chapter 11 describes incident response and computer forensics. When the inevitable happens and you have an incident to respond to, how will you do it? We talk about responding to attacks, and tracking down how the attack succeeded, through forensic analysis.
Conventions Used in This Book

We use both typography and common Unix documentation conventions to give you additional information in the text.

Typographic Conventions

Plain text
Indicates menu titles, menu options, menu buttons, and keyboard accelerators (such as Alt and Ctrl).

Italic
Indicates new or technical terms, system calls, URLs, hostnames, email addresses, filenames, file extensions, pathnames, and directories.

Constant width
Indicates commands, options, switches, variables, attributes, keys, functions, types, objects, HTML tags, macros, the contents of files, or the output from commands.

Constant width bold
Shows commands or other text that should be typed literally by the user.

Constant width italic
Shows text that should be replaced with user-supplied values.

This icon signifies a tip, suggestion, or general note.

This icon indicates a warning or caution.

There are times when it is very important to pay attention to the typography because it distinguishes between two similarly named, but different concepts. For example, the host command and the /etc/hosts file, or the jail(2) system call versus the jail(8) command. Sometimes the typeface is an important clue to help you remember which one we’re referring to in a given context.

Conventions in Examples

You will see two different prompts in the examples we give for running commands. We follow the time-honored Unix convention of using % to represent a non-root shell (e.g., one running as your normal user(8)) and # to represent a root-equivalent shell. Commands that appear after a % prompt can (and probably should) be run by an
unprivileged user. Commands that appear after a # prompt must be run with root privileges. Example P-1 shows three different commands that illustrate this point.

Example P-1. Several commands with different prompts

```
% ls -lo /var/log
% sudo ifconfig lo0 127.0.0.2 netmask 255.255.255.255
# shutdown -r now
```

The `ls` command runs as a normal user. The `ifconfig` command runs as root, but only because a normal user uses `sudo` to elevate his privileges momentarily (`sudo` is discussed in detail in Chapter 4). The last command shows the # prompt, assuming that you have already become root somehow before executing the `shutdown` command.

**Using Code Examples**

This book is here to help you get your job done. In general, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you’re reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O’Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product’s documentation does require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: *Mastering FreeBSD and OpenBSD Security* by Yanek Korff, Paco Hope, and Bruce Potter. Copyright 2005 O’Reilly Media, Inc., 0-596-00626-8.

If you feel your use of code examples falls outside fair use or the permissions given above, feel free to contact us at permissions@oreilly.com.

**Comments and Questions**

Please address comments and questions concerning this book to the publisher:

O’Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
(800) 998-9938 (in the United States or Canada)
(707) 829-0515 (international or local)
(707) 829-0104 (fax)
We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at:

http://www.oreilly.com/catalog/mfreeopenbsd/

To comment or ask technical questions about this book, send email to:

bookquestions@oreilly.com

For more information about our books, conferences, Resource Centers, and the O’Reilly Network, see our web site at:

http://www.oreilly.com

Safari Enabled

When you see a Safari® Books Online icon on the cover of your favorite technology book, that means the book is available online through the O’Reilly Network Safari Bookshelf.

Safari offers a solution that’s better than e-books. It’s a virtual library that lets you easily search thousands of top tech books, cut and paste code samples, download chapters, and find quick answers when you need the most accurate, current information. Try it for free at http://safaribooksonline.com.

Acknowledgments

Many people helped make this book possible, some of them in big ways and others in critical, yet nearly invisible ways. We’d like to acknowledge them here.

Yanek Korff

First and foremost, I’d like to thank my wife, whose patience continues to surprise me. This book would never have been possible without her help and her support. Also, although she’s not old enough to harbor a grudge or appreciate gratefulness, I’d like to thank my one-year-old daughter. She’s only ever known a workaholic father and doesn’t realize she should be jealous.

An obvious thank you to my parents for putting me on the road to geekdom back in early 90s, and of course putting me through college. May my educators forgive me for everything I’ve forgotten.

I’d also like to thank Viren Shah who introduced me to FreeBSD. I wouldn’t be where I am today without the support and mentoring he’s provided me over the years.
Finally, thanks to my good friend Matt Rowley, owner of much computer junk. Some of that junk and the advice that came with it were integral to this book’s creation.

Paco Hope

I’d like to thank my wife, Rebecca, who administered everything that doesn’t run FreeBSD (like children, houses, and pets) while I was building Frankenstein’s BSD lab in our basement. I am grateful for my time in the Department of Computer Science at the University of Virginia, where I cut my teeth as a system administrator. I thank the folks at Cigital, Inc. for introducing me to risk-based approaches to software and system security. Lastly, I thank Adrian Filipi, who gave me my first BSD/386 floppies back in 1993.

Bruce Potter

I would like to thank my wife for being incredibly understanding throughout the writing of this book and the million other things I had going on in the last year. She was amazing, even when I was not. I’d like to thank my kids, Terran and Bobby, and “Uncle Andy” for giving me time to write. Also, I would like to thank all the members of The Shmoo Group for helping me become the geek I am today. Without their friendship and expertise, I don’t know where my career would be today (full of moose, no doubt). The same goes to my folks who supported me through my fits and starts in college. And finally, a specific thanks to Joel Sadler, who gave me my first FreeBSD disk in 1995 telling me, “Here, try this. It’s better than Linux.”

Our Reviewers

We appreciate all the feedback we received from our technical reviewers. They definitely kept us on our toes and made this book better by lending their expert advice and opinions. Thanks to Flávio Marcelo Amaral, Ren Bitonio, Mark Delany, Adrian Filipi, Eric Jackson, Jose Nazario, Neil Neely, Wayne Pascoe, Viren Shah, and Shi-Min Yeh.

O’Reilly

Finally, we thank the staff at O’Reilly, especially Tatiana Diaz, Nathan Torkington, Allison Randal, David Chu, Andrew Savikas, and the innumerable others who have made this book a reality without our knowledge of their existence. An extra thank you goes to Tatiana for helping us reboot this effort after it locked up in the middle of 2004.
The goal of Part I is to give you the foundation for building and running secure systems with FreeBSD or OpenBSD.

- Chapter 1, The Big Picture
- Chapter 2, BSD Security Building Blocks
- Chapter 3, Secure Installation and Hardening
- Chapter 4, Secure Administration Techniques
Security is hard. We have all heard this phrase as a rationale for insecure systems and poor administrative practices. What’s worse, administrators seem to have different ideas about what “security” entails. There are two common approaches to securing systems: some view security as a destination while others see it as a journey.

Those who see security as a destination tend to characterize system security in terms of black and white; either a system is secure or it is not. This implies that you can attain security. You can arrive at the end of a journey and you’ll somehow be secure; you win. One problem with this viewpoint is determining where “there” is. How do you know when you’ve arrived? Furthermore, how do you stay there? As your system changes, are you still at your secure goal? Did you move away from it, or were you not there to begin with? As you can probably tell, this is not our philosophy.

Instead of being a destination, we think security is best described as a journey—a product of ongoing risk management. Rather than trying to make your system impregnable, you continually evaluate your exposure to risks and keep the system as secure as you need it to be. An appropriate level of security is achieved when the risks facing a system balance against the level of effort spent mitigating those risks. No one buys a $5,000 vault to safeguard a pair of fuzzy slippers. You judge the value of what you’re protecting against the kinds of threats it faces and the likelihood those threats will succeed, and then you apply appropriate safeguards. This is a much more practical way to view modern day information security.

When following a risk mitigation process, you will periodically pass up the opportunity to enable certain security mechanisms, even though you’re capable of doing so. The additional effort may not be warranted given the level of risk your organization faces. You will eventually reach a point of diminishing returns where you simply accept some risks because they are too costly to mitigate relative to the likelihood of
the threat or the actual damage that would occur. Sure, it may be fun to use encrypted filesystems, store all OS data on a CD-ROM, and deploy every other countermeasure you can think of, but do you really need to?

We define security in the context of risk. Risk is present as long as the system exists, and risks are constantly changing, so security cannot be a destination; it must be an ongoing process. “Doing security,” then, is an iterative process of identifying and responding to risks. This is the philosophy that we encourage you to take in securing your infrastructure.

As you’ll see in the rest of this book, FreeBSD and OpenBSD are robust operating systems that offer myriad ways to maintain secure systems. Throughout the book we provide security-minded walkthroughs of software installation, configuration, and maintenance. Along the way you’ll notice that we seem to point out more security-related configuration options than you care to implement. Just because we explore options doesn’t mean that you should implement them. Come at it from the perspective of managing risk and you’ll maximize the cost-benefit of “doing security.”

Before we get ahead of ourselves, however, we need to cover a few concepts and principles. In this chapter, we define system security, specifically for OpenBSD and FreeBSD systems, but also more generally. We look at a variety of attacks so that you, as an administrator, will have some perspective on what you’re trying to defend against. We’ll look at risk response and describe how exactly you can go about securing your FreeBSD and OpenBSD systems.

**What Is System Security?**

Security professionals break the term security into three parts: confidentiality, integrity, and availability. This “CIA Triad” is a set of security requirements; if you’re not taking into account all three of these concerns, you’re not working towards providing security. We offer a lot of recommendations in this book that should help you work towards building secure systems, but we don’t tell you how these recommendations fit in with the CIA Triad. That’s not what this book is about, and it would detract from the real message. Nevertheless, as you’re looking at building encrypted tunnels for transferring files, jailing applications, and so on, think about what part of the Triad you’re focusing on. Make sure you’ve addressed all three parts before your project is done.

Whether we’re talking about physical security, information security, network security, or system security, the CIA Triad applies. The question is, exactly how does it apply to system security?

**Confidentiality**

Confidentiality is all about determining the appropriate level of access to information. Confidentiality is also why we discuss the most basic level on FreeBSD and
OpenBSD systems by traditional Unix permissions. There are a variety of files scattered across the filesystem that are readable only by the root user. Most notable, perhaps, is `/etc/master.passwd`, which contains hashes for users’ passwords. The vast majority of files are readable by everyone, however. Even system configuration files like `/etc/resolv.conf`, `/etc/hosts`, and so on are world readable. Is this wrong? Not necessarily. Again, confidentiality isn’t about having to protect data from prying eyes; it’s about classifying data and making sure that information deemed sensitive in some way is protected appropriately.

Filesystem level protections are of course only one facet of confidentiality. Data may be exposed through some service designed to serve information like DNS, or a web server. In these cases, the method you employ to protect data won’t necessarily be filesystem permissions; perhaps you’ll control what systems are allowed to query your DNS server, or which web-authenticated users are permitted to view a certain document tree. When you need to protect data from eavesdropping as it moves across a network, you’ll probably use encryption. When implemented appropriately, it helps ensure that only the intended recipient can read the transmitted data.

**Integrity**

Data integrity relates to trust. If you cannot guarantee the integrity of some information on your system, you can’t trust it. Consequently, resources for which integrity is an important issue need to be identified and appropriately protected against modification.

Confidentiality may not have been an issue for your `/etc/resolv.conf` file. Allowing users to see what resolvers your system depends on is okay. Allowing users to modify the list of resolvers is not! Your system’s resolvers are a data source. When you access a server providing anonymous CVS access to your OpenBSD sources, your system will ask one of the servers listed in `/etc/resolv.conf` to find the IP address for the name you provided. If you can’t guarantee the integrity of the data in this file, you can’t trust the IP address you get from the resolver. As a consequence, you can’t trust the sources you download either.

Like confidentiality, the filesystem permissions model helps enforce data integrity. Unfortunately file permissions aren’t enough by themselves. If someone has broken through your filesystem protections somehow, you won’t know that your data has been tampered with. That is, not without good auditing. Moreover, you won’t be able to restore a known good configuration without data backups.

Data integrity is also an issue during network transfers. How can you be sure that the information has not been modified in transit? The BSD operating systems will provide “signatures,” which uniquely identify file distributions. When you download a package or source tarball or install a port, you can check your local files against the remote signatures. If it’s a match, your file has not been modified while in transit.
About the Authors

Yanek Korff graduated with a bachelor’s degree in computer science from the College of William and Mary and is currently a Certified Information Systems Security Professional (CISSP). Mr. Korff joined Bell Atlantic as a systems engineer where he played a role in the strategy, design, and deployment of a key northern Virginia test facility. He later joined Cigital, Inc., a software quality management company, where he helped design the system infrastructure. He is now a member of the information security division at America Online. During his career, Mr. Korff has been able to identify and mitigate information security risks particularly relating to host-based BSD security. By leveraging his experience, he has been able to apply security fundamentals to influence business and industry practices.

Paco Hope is a Senior Software Security Consultant with Cigital, Inc. His areas of expertise include software security, embedded systems, PKI, and host security. Mr. Hope has published articles on X.509 revocation and Unix host security features. Prior to joining Cigital, Inc., he served as director of product development for Tovaris, Inc., based in Charlottesville, Virginia, and as head systems administrator in the department of computer science at the University of Virginia. Mr. Hope was a double major in computer science and English at the College of William and Mary and received an M.S. in computer science from the University of Virginia.

Bruce Potter is a Senior Associate with Booz Allen Hamilton. Prior to working at Booz Allen, Mr. Potter served as a software security consultant for Cigital, Inc., in Dulles, Virginia. Mr. Potter is the founder of the Shmoo Group of security professionals. His areas of expertise include wireless security, large-scale network architectures, smartcards and promotion of secure software engineering practices. Mr. Potter coauthored the books 802.11 Security (O’Reilly) and Mac OS X Security (New Riders). Mr. Potter was trained in computer science at the University of Alaska, Fairbanks.

Colophon

Our look is the result of reader comments, our own experimentation, and feedback from distribution channels. Distinctive covers complement our distinctive approach to technical topics, breathing personality and life into potentially dry subjects.

The image on the cover of Mastering FreeBSD and OpenBSD Security depicts fencers. Whether used for sport or for war, the art of fencing can be traced back to some of the earliest known civilizations. For example, fencers entertained Pharaohs in ancient Egypt. The Greeks and Romans, meanwhile, had systems of martial arts that included swordsmanship. The modern sport of fencing originated in the first Olympic Games, in 1896, and consists of three different weapons: foil, épée, and sabre. The lightest of these weapons is the foil. A foil fencer can only score hits by landing thrusts to the trunk. The modern electrical scoring apparatus,
worn by the fencer, will record a hit for any blow landed with a force of at least 4.90 newtons. Less flexible and heavier than the foil, the épée usually has a large hand guard. This bell-shaped guard is important because the épée fencer is not as limited in her targets—the entire body, including the hand, is considered a valid target to score hits. An épée fencer registers a hit with 7.35 newtons of force. The sabre differs from these first two swords in that it is an edge, rather than a point, weapon. A sabre fencer may land points to any part of the upper body (head, torso, and arms). A touch with the point, flat, or edge of the sword will register a hit.

Adam Witwer was the production editor, and Nancy Reinhardt was the copyeditor for Mastering FreeBSD and OpenBSD Security. Linley Dolby proofread the text. Sarah Sherman and Claire Cloutier provided quality control. Lucie Haskins wrote the index.

Emma Colby designed the cover of this book, based on a series design by Edie Freedman. The cover image is a 19th-century engraving from the Dover Pictorial Archive. Karen Montgomery produced the cover layout with Adobe InDesign CS using Adobe’s ITC Garamond font.

David Futato designed the interior layout. This book was converted by Judy Hoer to FrameMaker 5.5.6 with a format conversion tool created by Erik Ray, Jason McIntosh, Neil Walls, and Mike Sierra that uses Perl and XML technologies. The text font is Linotype Birka; the heading font is Adobe Myriad Condensed; and the code font is LucasFont’s TheSans Mono Condensed. The illustrations that appear in the book were produced by Robert Romano, Jessamyn Read, and Lesley Borash using Macromedia FreeHand MX and Adobe Photoshop CS. The tip and warning icons were drawn by Christopher Bing. This colophon was written by Adam Witwer.