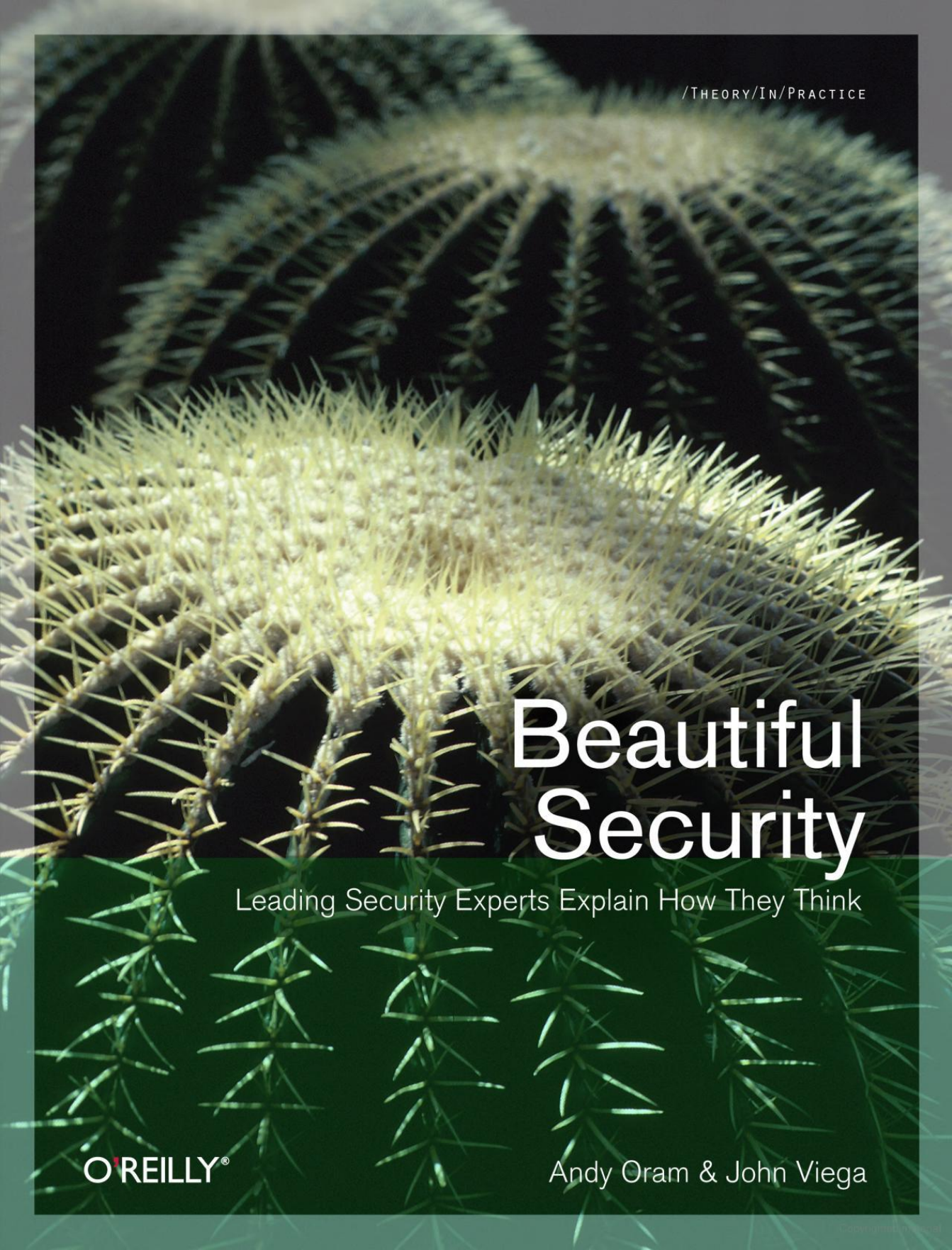
/Theory/In/Practice
Beautiful Security
Leading Security Experts Explain How They Think
O'REILLY®
Andy Oram & John Viega

Beautiful Security

Edited by Andy Oram and John Viega

O'REILLY®
Beijing • Cambridge • Farnham • Köln • Sebastopol • Taipei • Tokyo

Copyrighted material

Beautiful Security
Edited by Andy Oram and John Viega

Copyright © 2009 O'Reilly Media, Inc. All rights reserved.
Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://my.safaribooksonline.com/*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Production Editor: Sarah Schneider
Copyeditor: Genevieve d'Entremont
Proofreader: Sada Preisch

Indexer: Lucie Haskins
Cover Designer: Mark Paglietti
Interior Designer: David Futato
Illustrator: Robert Romano

Printing History:
April 2009: First Edition.

O'Reilly and the O'Reilly logo are registered trademarks of O'Reilly Media, Inc. *Beautiful Security*, the image of a cactus, and related trade dress are trademarks of O'Reilly Media, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc., was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher and authors assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

ISBN: 978-0-596-52748-8

[V]

1239647579

Copyrighted material

CONTENTS

Copyrighted material

Copyrighted material

Copyrighted material

Copyrighted material

Preface

If one believes that news headlines reveal trends, these are interesting times for computer security buffs. As *Beautiful Security* went to press, I read that a piece of software capable of turning on microphones and cameras and stealing data has been discovered on more than 1,200 computers in 103 countries, particularly in embassies and other sensitive government sites. On another front, a court upheld the right of U.S. investigators to look at phone and Internet records without a warrant (so long as one end of the conversation is outside the U.S.). And this week's routine vulnerabilities include a buffer overflow in Adobe Acrobat and Adobe Reader—with known current exploits—that lets attackers execute arbitrary code on your system using your privileges after you open their PDF.

Headlines are actually not good indicators of trends, because in the long run history is driven by subtle evolutionary changes noticed only by a few—such as the leading security experts who contributed to this book. The current directions taken by security threats as well as responses can be discovered in these pages.

All the alarming news items I mentioned in the first paragraph are just business as usual in the security field. Yes, they are part of trends that should worry all of us, but we also need to look at newer and less dramatic vulnerabilities. The contributors to this book have, for decades, been on the forefront of discovering weaknesses in our working habits and suggesting unconventional ways to deal with them.

Copyrighted material

Why Security Is Beautiful

I asked security expert John Viega to help find the authors for this book out of frustration concerning the way ordinary computer users view security. Apart from the lurid descriptions of break-ins and thefts they read about in the press, average folks think of security as boring.

Security, to many, is represented by nagging reminders from system administrators to create backup folders, and by seemingly endless dialog boxes demanding passwords before a web page is displayed. Office workers roll their eyes and curse as they read the password off the notepad next to their desk (lying on top of the budget printout that an office administrator told them should be in a locked drawer). If this is security, who would want to make a career of it? Or buy a book from O'Reilly about it? Or think about it for more than 30 seconds at a time?

To people tasked with creating secure systems, the effort seems hopeless. Nobody at their site cooperates with their procedures, and the business managers refuse to allocate more than a pittance to security. Jaded from the endless instances of zero-day exploits and unpatched vulnerabilities in the tools and languages they have to work with, programmers and system administrators become lax.

This is why books on security sell poorly (although in the last year or two, sales have picked up a bit). Books on hacking into systems sell much better than books about how to protect systems, a trend that really scares me.

Well, this book should change that. It will show that security is about the most exciting career you can have. It is not tedious, not bureaucratic, and not constraining. In fact, it exercises the imagination like nothing else in technology.

Most of the programming books I've edited over the years offer a chapter on security. These chapters are certainly useful, because they allow the author to teach some general principles along with good habits, but I've been bothered by the convention because it draws a line around the topic of security. It feeds the all-too-common view of security as an add-on and an afterthought. *Beautiful Security* demolishes that conceit.

John chose for this book a range of authors who have demonstrated insight over and over in the field and who had something new to say. Some have designed systems that thousands rely on; some have taken high-level jobs in major corporations; some have testified on and worked for government bodies. All of them are looking for the problems and solutions that the rest of us know nothing about—but will be talking about a lot a few years from now.

The authors show that effective security keeps you on your toes all the time. It breaks across boundaries in technology, in cognition, and in organizational structures. The black hats in security succeed by exquisitely exercising creativity; therefore, those defending against them must do the same.

Copyrighted material

With the world's infosecurity resting on their shoulders, the authors could be chastised for taking time off to write these chapters. And indeed, many of them experienced stress trying to balance their demanding careers with the work on this book. But the time spent was worth it, because this book can advance their larger goals. If more people become intrigued with the field of security, resolve to investigate it further, and give their attention and their support to people trying to carry out organizational change in the interest of better protection, the book will have been well worth the effort.

On March 19, 2009, the Senate Committee on Commerce, Science, and Transportation held a hearing on the dearth of experts in information technology and how that hurts the country's cybersecurity. There's an urgent need to interest students and professionals in security issues; this book represents a step toward that goal.

Audience for This Book

This book is meant for people interested in computer technology who want to experience a bit of life at the cutting edge. The audience includes students exploring career possibilities, people with a bit of programming background, and those who have a modest to advanced understanding of computing.

The authors explain technology at a level where a relatively novice reader can get a sense of the workings of attacks and defenses. The expert reader can enjoy the discussions even more, as they will lend depth to his or her knowledge of security tenets and provide guidance for further research.

Donation

The authors are donating the royalties from this book to the Internet Engineering Task Force (IETF), an organization critical to the development of the Internet and a fascinating model of enlightened, self-organized governance. The Internet would not be imaginable without the scientific debates, supple standard-making, and wise compromises made by dedicated members of the IETF, described on their web page as a "large open international community of network designers, operators, vendors, and researchers." O'Reilly will send royalties to the Internet Society (ISOC), the longtime source of funding and organizational support for the IETF.

Organization of the Material

The chapters in this book are not ordered along any particular scheme, but have been arranged to provide an engaging reading experience that unfolds new perspectives in hopefully surprising ways. Chapters that deal with similar themes, however, are grouped together.

Copyrighted material

Chapter 1, *Psychological Security Traps*, by Peiter "Mudge" Zatko

Chapter 2, *Wireless Networking: Fertile Ground for Social Engineering*, by Jim Stickley

Chapter 3, *Beautiful Security Metrics*, by Elizabeth A. Nichols

Chapter 4, *The Underground Economy of Security Breaches*, by Chenxi Wang

Chapter 5, *Beautiful Trade: Rethinking E-Commerce Security*, by Ed Bellis

Chapter 6, *Securing Online Advertising: Rustlers and Sheriffs in the New Wild West*, by Benjamin Edelman

Chapter 7, *The Evolution of PGP's Web of Trust*, by Phil Zimmermann and Jon Callas

Chapter 8, *Open Source Honeyclient: Proactive Detection of Client-Side Exploits*, by Kathy Wang

Chapter 9, *Tomorrow's Security Cogs and Levers*, by Mark Curphey

Chapter 10, *Security by Design*, by John McManus

Chapter 11, *Forcing Firms to Focus: Is Secure Software in Your Future?*, by James Routh

Chapter 12, *Oh No, Here Come the Infosecurity Lawyers!*, by Randy V. Sabett

Chapter 13, *Beautiful Log Handling*, by Anton Chuvakin

Chapter 14, *Incident Detection: Finding the Other 68%*, by Grant Geyer and Brian Dunphy

Chapter 15, *Doing Real Work Without Real Data*, by Peter Wayner

Chapter 16, *Casting Spells: PC Security Theater*, by Michael Wood and Fernando Francisco

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, filenames, and Unix utilities.

`Constant width`
: Indicates the contents of computer files and generally anything found in programs.

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

Copyrighted material

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: "*Beautiful Security*, edited by Andy Oram and John Viega. Copyright 2009 O'Reilly Media, Inc., 978-0-596-52748-8."

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

Safari® Books Online

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://my.safaribooksonline.com/*.

How to Contact Us

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/9780596527488

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 website at:

http://www.oreilly.com

Copyrighted material

Copyrighted material

CHAPTER ONE

Psychological Security Traps

Peiter "Mudge" Zatko

During my career of attacking software and the facilities they power, many colleagues have remarked that I have a somewhat nonstandard approach. I tended to be surprised to hear this, as the approach seemed logical and straightforward to me. In contrast, I felt that academic approaches were too abstract to realize wide success in real-world applications. These more conventional disciplines were taking an almost completely random tack with no focus or, on the opposite end of the spectrum, spending hundreds of hours reverse-engineering and tracing applications to (hopefully) discover their vulnerabilities before they were exploited out in the field.

Now, please do not take this the wrong way. I'm not condemning the aforementioned techniques. In fact I agree they are critical tools in the art of vulnerability discovery and exploitation. However, I believe in applying some shortcuts and alternative views to envelope, enhance, and—sometimes—bypass these approaches.

In this chapter I'll talk about some of these alternative views and how they can help us get inside the mind of the developer whose code or system we engage as security professionals.

Why might you want to get inside the mind of the developer? There are many reasons, but for this chapter we will focus on various constraints that are imposed on the creation of code and the people who write it. These issues often result in suboptimal systems from the security viewpoint, and by understanding some of the environmental, psychological, and philosophical frameworks in which the coding is done, we can shine a spotlight on which areas of a system

Copyrighted material

are more likely to contain vulnerabilities that attackers can exploit. Where appropriate, I'll share anecdotes to provide examples of the mindset issue at hand.

My focus for the past several years has been on large-scale environments such as major corporations, government agencies and their various enclaves, and even nation states. While many of the elements are applicable to smaller environments, and even to individuals, I like to show the issues in larger terms to offer a broader social picture. Of course, painting with such a broad brush requires generalizations, and you may be able to find instances that contradict the examples. I won't cite counterexamples, given the short space allotted to the chapter.

The goal here is not to highlight particular technologies, but rather to talk about some environmental and psychological situations that caused weak security to come into being. It is important to consider the external influences and restrictions placed on the implementers of a technology, in order to best understand where weaknesses will logically be introduced. While this is an enjoyable mental game to play on the offensive side of the coin, it takes on new dimensions when the defenders also play the game and a) prevent errors that would otherwise lead to attacks or b) use these same techniques to game the attackers and how they operate. At this point, the security game becomes what I consider *beautiful*.

The mindsets I'll cover fall into the categories of learned helplessness and naïveté, confirmation traps, and functional fixation. This is not an exhaustive list of influencing factors in security design and implementation, but a starting point to encourage further awareness of the potential security dangers in systems that you create or depend on.

Learned Helplessness and Naïveté

Sociologists and psychologists have discovered a phenomenon in both humans and other animals that they call *learned helplessness*. It springs from repeated frustration when trying to achieve one's goals or rescue oneself from a bad situation. Ultimately, the animal subjected to this extremely destructive treatment stops trying. Even when chances to do well or escape come along, the animal remains passive and fails to take advantage of them.

To illustrate that even sophisticated and rational software engineers are subject to this debilitating flaw, I'll use an example where poor security can be traced back to the roots of backward compatibility.

Backward compatibility is a perennial problem for existing technology deployments. New technologies are discovered and need to be deployed that are incompatible with, or at the very least substantially different from, existing solutions.

At each point in a system's evolution, vendors need to determine whether they will forcibly end-of-life the existing solutions, provide a migration path, or devise a way to allow both the legacy and modern solutions to interact in perpetuity. All of these decisions have numerous ramifications from both business and technology perspectives. But the decision is usually

Copyrighted material

driven by business desires and comes down as a decree to the developers and engineers.* When this happens, the people responsible for creating the actual implementation will have the impression that the decision has already been made and that they just have to live with it. No further reevaluation or double guessing need take place.

Imagine that the decision was made to maintain compatibility with the legacy technology in its replacement. Management further decrees that no further development or support work will take place on the legacy solution, in order to encourage existing customers to migrate to the replacement.

Although such decisions place burdens on the development in many ways—with security implications—they are particularly interesting when one solution, usually the new technology, is more secure than the other. In fact, new technologies are often developed *explicitly* to meet the need for greater security—and yet the old technology must still be supported. What security problems arise in such situations?

There are different ways to achieve backward compatibility, some more secure than others. But once the developers understand that the older, less secure technology is allowed to live on, solutions that would ease the risk are often not considered at all. The focus is placed on the new technology, and the legacy technology is glued into it (or vice versa) with minimal attention to the legacy's effects. After all, the team that is implementing the new technology usually didn't develop the legacy code and the goal is to ultimately supplant the legacy solution anyway—right?

The most direct solution is to compromise the robustness and security strength of the new technology to match that of the legacy solution, in essence allowing both the modern and legacy technology to be active simultaneously. Learned helplessness enters when developers can't imagine that anything could be done—or worse, even should be done—to mitigate the vulnerabilities of the legacy code. The legacy code was forced on them, it is not perceived to be their bailiwick (even if it impacts the security of the new technology by reducing it to the level of the old), and they feel they are powerless to do anything about it anyway due to corporate decree.

A Real-Life Example: How Microsoft Enabled L0phtCrack

Years ago, to help system administrators uncover vulnerabilities, I wrote a password-cracking tool that recovered Microsoft user passwords. It was called L0phtCrack at the time, later to be renamed LC5, and then discontinued by Symantec (who had acquired the rights to it) due to concerns that it could be considered a munition under the International Tariff on Arms Regulations (ITAR).† Many articles on the Net and passages in technical books have been written about how L0phtCrack worked, but none have focused on *why* it worked in the first

* Or at least it often appears to the developers and engineers that this is the case.

† This might not be the end of L0phtCrack....

Copyrighted material

place. What were some of the potential influences that contributed to the vulnerabilities that L0phtCrack took advantage of in Microsoft Windows?

In fact, the tool directly exploited numerous problems in the implementation and use of cryptographic routines in Windows. All these problems originated in the legacy LAN Manager (or LANMAN) hash function that continued to be used in versions of Windows up to Vista. Its hash representation, although based on the already aging Data Encryption Standard (DES), contained no salt. In addition, passwords in LANMAN were case-insensitive. The function broke the 14-character or shorter password into two 7-byte values that were each encrypted against the same key and then concatenated. As I described in a post to BugTraq in the late 1990s, the basic encryption sequence, illustrated in Figure 1-1, is:

1. If the password is less than 14 characters, pad it with nulls to fill out the allocated 14-character space set aside for the password. If the password is greater than 14 characters, in contrast, it is truncated.
2. Convert the 14-character password to all uppercase and split it into two 7-character halves. It should be noted that if the original password was 7 or fewer characters, the second half will always be 7 nulls.
3. Convert each 7-byte half to an 8-byte parity DES key.
4. DES encrypt a known constant ("KGS!@#$%") using each of the previously mentioned keys.
5. Concatenate the two outputs to form the LM_HASH representation.

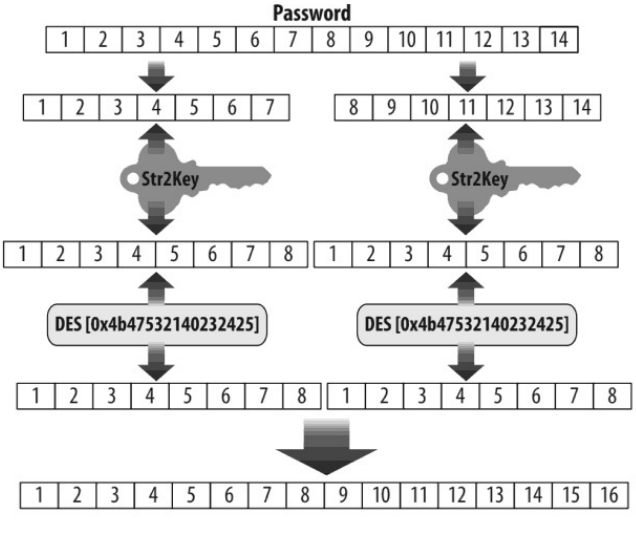

FIGURE 1-1. Summary of old LANMAN hash algorithm

Copyrighted material

This combination of choices was problematic for many technical reasons.

The developers of Windows NT were conscious of the weaknesses in the LANMAN hash and used a stronger algorithm for its storage of password credentials, referred to as the NT hash. It maintained the case of the characters, allowed passwords longer than 14 characters, and used the more modern MD4 message digest to produce its 16-byte hash.

Unfortunately, Windows systems continued to store the weaker version of each password next to the stronger one—and to send both versions over the network each time a user logged in. Across the network, both the weaker 16-byte LANMAN hash and the stronger 16-byte NT hash underwent the following process, which is represented in Figure 1-2:

1. Pad the hash with nulls to 21 bytes.
2. Break the 21-byte result into three 7-byte subcomponents.
3. Convert each 7-byte subcomponent to 8-byte parity DES keys.
4. Encrypt an 8-byte challenge, which was visibly sent across the network, using the previously mentioned DES keys.
5. Concatenate the three 8-byte outputs from step 4 to make a 24-byte representation that would be sent over the network.

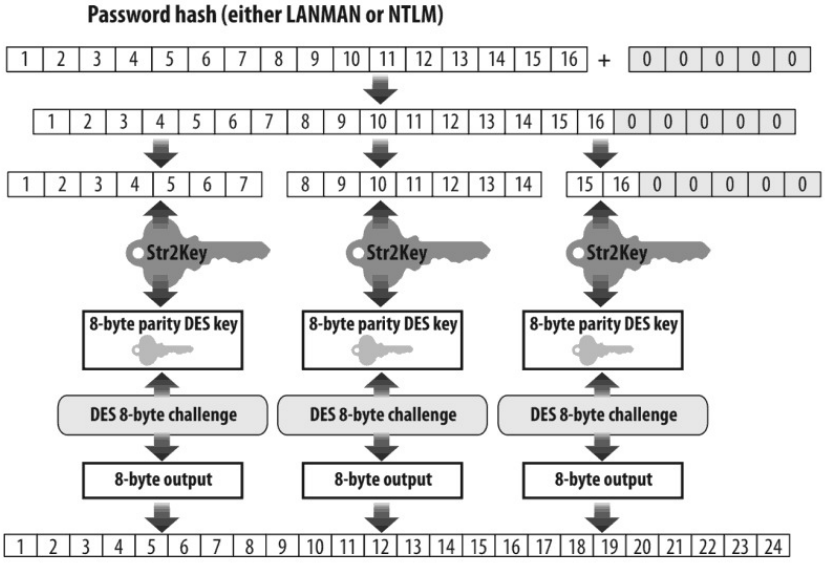

FIGURE 1-2. Handling both LANMAN and NT hashes over the network

Copyrighted material

Microsoft preferred for all their customers to upgrade to newer versions of Windows, of course, but did not dare to cut off customers using older versions or even retrofit them with the new hash function. Because the password was a key part of networking, they had to assume that, for the foreseeable future, old systems with no understanding of the new hash function would continue to connect to systems fitted out with the more secure hash.

If systems on both sides of the login were new systems with new hash functions, they could perform the actual authentication using the stronger NT hash. But a representation of the older and more vulnerable LANMAN hash was sent right alongside its stronger sibling.

By taking the path of least resistance to backward compatibility and ignoring the ramifications, Microsoft completely undermined the technical advances of its newer security technology.

L0phtCrack took advantage of the weak LANMAN password encoding and leveraged the results against the stronger NTLM representation that was stored next to it. Even if a user chose a password longer than 14 characters, the cracking of the LANMAN hash would still provide the first 14, leaving only a short remnant to guess through inference or brute force. Unlike LANMAN, the NT hash was case-sensitive. But once the weak version was broken, the case specifics of the password in the NT hash could be derived in a maximum of 2^x attempts (where x is the length of the password string) because there were at most two choices (uppercase or lowercase) for each character. Keep in mind that x was less than or equal to 14 and thus trivial to test exhaustively.

Although NTLM network authentication introduced a challenge that was supposed to act as a salt mechanism, the output still contained too much information that an attacker could see and take advantage of. Only two bytes from the original 16-byte hash made it into the third 7-byte component; the rest was known to be nulls. Similarly, only one byte—the eighth—made it from the first half of the hash into the second 7-byte component.

Think of what would happen if the original password were seven characters or less (a very likely choice for casual users). In the LANMAN hash, the second group of 7 input bytes would be all nulls, so the output hash bytes 9 through 16 would always be the same value. And this is further propagated through the NTLM algorithm. At the very least, it takes little effort to determine whether the last 8 bytes of a 24-byte NTLM authentication response were from a password that was less than eight characters.

In short, the problems of the new modern security solution sprang from the weaker LANMAN password of the legacy system and thus reduced the entire security profile to its lowest common denominator. It wasn't until much later, and after much negative security publicity, that Microsoft introduced the capability of sending only one hash or the other, and not both by default—and even later that they stopped storing both LANMAN and NT hashes in proximity to each other on local systems.

Copyrighted material

Password and Authentication Security Could Have Been Better from the Start

My L0phtCrack story was meant to highlight a common security problem. There are many reasons to support multiple security implementations, even when one is known to be stronger than the others, but in many cases, as discussed earlier, the reason is to support backward compatibility. Once support for legacy systems is deemed essential, one can expect to see a fair amount of redundancy in protocols and services.

The issue from a security standpoint becomes how to accomplish this backward compatibility without degrading the security of the new systems. Microsoft's naïve solution embodied pretty much the worst of all possibilities: it stored the insecure hash together with the more secure one, and for the benefit of the attacker it transmitted the representations of both hashes over the network, even when not needed!

Remember that *learned helplessness* is the situation where one comes to the conclusion that he is helpless or has no recourse by training rather than from an actual analysis of the situation at hand. In other words, someone tells you that you are helpless and you believe them based on nothing more than their "say so." In engineering work, learned helplessness can be induced by statements from apparent positions of authority, lazy acceptance of backward compatibility (or legacy customer demand), and through cost or funding pressures (perceived or real).

Microsoft believed the legacy systems were important enough to preclude stranding these systems. In doing this they made the decision to keep supporting the LM hash.

But they took a second critical step and chose to deal with the protocol problem of legacy and modern interactions by forcing their new systems to talk to both the current protocol and the legacy one without considering the legacy security issues. Instead, they could have required the legacy systems to patch the handful of functions required to support logins as a final end-of-life upgrade to the legacy systems. Perhaps this solution was rejected because it might set a dangerous precedent of supporting systems that they had claimed had reached their end-of-life. They similarly could have chosen not to send both old and new hashes across the network when both systems could speak the more modern and stronger variant. This would have helped their flagship "New Technology" offering in both actual and perceived security.

Ultimately Microsoft enabled their systems to refrain from transmitting the weaker LANMAN hash representation, due to persistent media and customer complaints about the security weakness, in part prompted by the emergence of attack tools such as L0phtCrack. This shows that the vendor could have chosen a different path to start with and could have enabled the end users to configure the systems to their own security requirements. Instead, they seem to have fallen victim to the belief that when legacy support is required, one must simply graft it onto the new product and allow all systems to negotiate down to the lowest common denominator. This is an example of learned helplessness from the designer and implementer standpoints within a vendor.

Copyrighted material

NOT MICROSOFT ALONE

Lest the reader think I'm picking on Microsoft, I offer the following equal-opportunity (and potentially offending) observations.

During this time frame (the mid- to late 1990s), Microsoft was taking the stance in marketing and media that its systems were more secure than Unix. The majority of the servers on the Internet were Unix systems, and Microsoft was trying to break into this market. It was well known that numerous security vulnerabilities had been found in the various Unix variants that made up the vast majority of systems on the public Internet. Little research, however, had been performed on the security of Microsoft's Windows NT 4.0 from an Internet perspective. This was due in no small part to the fact that NT 4.0 systems were such a small fraction of the systems on the Net.

Microsoft's stance was, in essence, "we are secure because we are not Unix." But it took until the Vista release of the Windows operating system for Microsoft to really show strong and modern security practices in an initial OS offering. Vista has had its own issues, but less on the security front than other factors. So, when NT 4.0 was novel, Microsoft picked on Unix, citing their long list of security issues at the time. The shoe went on the other foot, and people now cite the litany of Microsoft security issues to date. Now that Microsoft actually offers an operating system with many strong security components, will there be anyone to pick on? Enter Apple.

Apple Computer has seemingly taken a similar marketing tack as Microsoft did historically. Whereas Microsoft essentially claimed that they were secure because they were not Unix, Apple's marketing and user base is stating that its OS X platform is more resistant to attacks and viruses essentially because it is not Windows. Having had a good look around the kernel and userland space of OS X, I can say that there are many security vulnerabilities (both remote and local) still waiting to be pointed out and patched. Apple appears to be in a honeymoon period similar to Microsoft's first NT offering: Apple is less targeted because it has a relatively tiny market share. But both its market share and, predictably, the amount of attack focus it receives seems to be increasing....

Naïveté As the Client Counterpart to Learned Helplessness

As we've seen, the poor security choice made by Microsoft in backward compatibility might have involved a despondent view (justified or not) of their customers' environment, technical abilities, and willingness to change. I attribute another, even larger, security breach in our current networks to a combination of learned helplessness on the part of the vendor and naïveté on the part of the customer. A long trail of audits has made it clear that major manufacturers of network switches have intentionally designed their switches to "fail open" rather than closed. Switches are designed to move packets between systems at the data-link layer. *Failing closed,* in this case, means that a device shuts down and stops functioning or otherwise ceases operation in a "secure" fashion. This would result in data no longer passing

Copyrighted material

through the system in question. Conversely, *failing open* implies that the system stops performing any intelligent functions and just blindly forwards all packets it receives out of all of its ports.‡

In essence, a switch that fails open turns itself into a dumb hub. If you're out to passively sniff network traffic that is not intended for you, a dumb hub is just what you want. A properly functioning switch will attempt to send traffic only to the appropriate destinations.

Many organizations assume that passive network sniffing is not a viable threat because they are running switches. But it is entirely common nowadays to connect a sniffer to a switched LAN and see data that is not destined for you—often to the extreme surprise of the networking group at that organization. They don't realize that the vendor has decided to avoid connectivity disruptions at all costs (probably because it correctly fears the outrage of customer sites whose communications come to a screeching halt), and therefore make its switches revert to a dumb broadcast mode in the event that a switch becomes confused through a bug, a security attack, or the lack of explicit instructions on what to do with certain packets. The vendor, in other words, has quietly made a decision about what is best for its customer.

I would like to believe that the customer would be in a better position to determine what is and what is not in her best interest. While it might be a good idea for a switch to fail open rather than shut down an assembly line, there are situations where switches are used to separate important traffic and segregate internal domains and systems. In such cases it might be in the best interest of the customer if the switch fails closed and sends an alarm. The customer should at least be provided a choice.

Here we have both learned helplessness on the vendor's part and naïveté on the consumer's part. The learned helplessness comes from the vendor's cynicism about its ability to educate the customer and get the customer to appreciate the value of having a choice. This is somewhat similar to the previous discussion of legacy system compatibility solutions. The vendor believes that providing extra configurability of this kind will just confuse the customer, cause the customer to shoot herself in the foot, or generate costly support calls to the vendor.

The naïveté of the client is understandable: she has bought spiffy-looking systems from well-established vendors and at the moment everything seems to be running fine. But the reasonableness of such naïveté doesn't reduce its usefulness to an adversary. *Must a system's security be reduced by an attempt to have it always work in any environment?* Are protocols blinded deliberately to allow legacy version systems to interact at weaker security levels? If a system gets confused, will it revert to acting as a dumb legacy device? These situations can often be traced back to learned helplessness.

‡ This is the opposite of electrical circuits, where failing closed allows current to flow and failing open breaks the circuit.

Copyrighted material

Confirmation Traps

Hobbit (friend and hacker[§] extraordinaire) and I were having dinner one August (back around 1997) with an executive and a senior engineer[‖] from Microsoft. They wanted to know how we were able to find so many flaws in Microsoft products so readily. I believe, with imperfect memory, that we replied truthfully and said we would input random garbage into the systems. This is a straightforward bug- and security-testing technique, sometimes known as "fuzzing," and has now been documented in major computer science publications. However, fuzzing was not widely practiced by the "hacker" community at the time.

We told the engineer we were surprised how often Windows would go belly-up when confronted with fuzzed input. We followed up by asking what sort of robustness testing they performed, as it would seem that proper QA would include bad input testing and should have identified many of the system and application crashes we were finding.

The engineer's response was that they performed exhaustive usability testing on all of their products, but that this did not include *trying* to crash the products. This response shone a light on the problem. While Microsoft made efforts to ensure a good user experience, they were not considering adversarial users or environments.

As an example, teams that developed Microsoft Word would test their file parsers against various acceptable input formats (Word, Word Perfect, RTF, plain text, etc.). They would not test variations of the expected formats that could be created by hand but could never be generated by a compatible word processor. But a malicious attacker will test these systems with malformed versions of the expected formats, as well as quasi-random garbage.

When we asked the senior Microsoft representatives at dinner why they did not send malicious data or provide malformed files as input to their product's testing, the answer was, "Why would a user want to do that?" Their faces bore looks of shock and dismay that anyone would intentionally interact with a piece of software in such a way as to intentionally try to make it fail.

They never considered that their applications would be deployed in a hostile environment. And this view of a benign world probably sprang from another psychological trait that malicious attackers can exploit: *confirmation traps*.

An Introduction to the Concept

Microsoft's product testing was designed to *confirm* their beliefs about how their software behaved rather than *refute* those beliefs. Software architects and engineers frequently suffer

[§] The word "hacker" is being used in the truest and most positive sense here.

[‖] Sometimes it seems it is cheaper to hire a key inventor of a protocol and have him "reinvent" it rather than license the technology. One of the people responsible for Microsoft's "reimplementation" of DCE/RPC into SMB/CIFS was the engineer present at the dinner.

Copyrighted material

from this blind spot. In a 1968 paper, Peter Wason pointed out that "obtaining the correct solution necessitates a willingness to attempt to falsify the hypothesis, and thus test the intuitive ideas that so often carry the feeling of certitude."[#] He demonstrated confirmation traps through a simple mental test.

Find some people and inform them that you are conducting a little experiment. You will provide the participant with a list of integers conforming to a rule that he is supposed to guess. To determine the rule, he should propose some more data points, and you will tell him whether each of his sets of points conform to the unspoken rule. When the participant thinks he knows what the rule is, he can propose it.

In actuality, the rule is simply *any three ascending numbers*, but you will keep this to yourself.

The initial data points you will provide are the numbers 2, 4, and 6.

At this point, one of the participants might offer the numbers 8, 10, and 12. You should inform her that 8, 10, 12 does indeed conform to the rule. Another participant might suggest 1, 3, and 5. Again, you would confirm that the series 1, 3, and 5 conforms to the rule.

People see the initial series of numbers 2, 4, and 6 and note an obvious relationship: that each number is incremented by two to form the next number. They incorporate this requirement—which is entirely in their own minds, not part of your secret rule—into their attempts to provide matching numbers, and when these sequences conform, the confirmation pushes them further down the path of confirming their preconceived belief rather than attempting to refute it.

Imagine the secret rule now as a software rule for accepting input, and imagine that the participants in your experiment are software testers who believe all users will enter sequences incremented by two. They won't test other sequences, such as 1,14, and 9,087 (not to mention −55, −30, and 0). And the resulting system is almost certain to accept untested inputs, only to break down.

Why do confirmation traps work? The fact is that we all like to be correct rather than incorrect. While rigid logic would dictate trying to test our hypotheses—that all inputs must be even numbers, or must be incremented by two—by proposing a series that does not conform to our hypothesis (such as 10, 9, 8), it is simply human nature to attempt to reinforce our beliefs rather than to contradict them.

"Does a piece of software work as expected?" should be tested not just by using it the way you intend, but also through bizarre, malicious, and random uses. But internal software testing rarely re-creates the actual environments and inputs to which software will be subjected, by regular end users and hostile adversaries alike.

[#] "Reasoning About a Rule," Peter Wason, *The Quarterly Journal of Experimental Psychology*, Vol. 20, No. 3. 1968.

Copyrighted material

The Analyst Confirmation Trap

Consider an intelligence analyst working at a three-letter agency. The analyst wants to create valid and useful reports in order to progress up the career ladder. The analyst culls information from multiple sources, including the previous reports of analysts in her position. The analyst then presents these reports to her superior. While this might seem straightforward, it entails a potential confirmation trap. Before her superiors were in the position to review her work, it is quite likely that *they* were the prior analysts that created some of the reports the current analyst used as background. In other words, it is not uncommon that the input to a decision was created by the people reviewing that decision.

It should be apparent that the analyst has a proclivity to corroborate the reports that were put together by her boss rather than to attempt to challenge them. She might fall into line quite consciously, particularly if she is trying to make a career in that community or organization, or do it unconsciously as in Wason's example with three ascending numbers. At the very least, the structure and information base of the agency creates a strong potential for a self-reinforcing feedback loop.

I have personally witnessed two cases where people became cognizant of confirmation traps and actively worked to ensure that they did not perpetuate them. Not surprisingly, both cases involved the same people that brought the intelligence analyst scenario to my attention and who confirmed my suspicions regarding how commonly this error is made in intelligence reports.

Stale Threat Modeling

During a previous presidency, I acted as an advisor to a key group of people in the Executive Office. One of my important tasks was to express an opinion about a briefing someone had received about cyber capabilities (both offensive and defensive) and which areas of research in those briefings were valid or had promise. I would often have to point out that the initial briefings were woefully inaccurate in their modeling of adversaries and technologies. The technology, tactics, and capabilities being presented were not even close to representative of the techniques that could be mustered by a well-financed and highly motivated adversary. Many of the techniques and tactics described as being available only to competent nation-state adversaries were currently run-of-the-mill activities for script kiddies and hobbyists of the day.

The briefings did try to understand how cyber threats were evolving, but did so unimaginatively by extrapolating from historical technology. Technology had progressed but the models had not, and had been left far behind reality. So the briefings ended up regurgitating scenarios that were possibly based in accurate generalizations at one point in the past, but were now obsolete and inaccurate. This is endemic of confirmation traps. And as it turned out, the briefings I had been asked to comment on had come about due to situations similar to the aforementioned analyst confirmation trap.

Copyrighted material

Rationalizing Away Capabilities

As the success of the L0pht in breaking security and releasing such tools as L0phtCrack became well known, the government developed a disturbing interest in our team and wanted to understand what we were capable of. I reluctantly extended an invitation to a group from the White House to visit and get a briefing. Now, mind you, the L0pht guys were not very comfortable having a bunch of spooks and government representatives visiting, but eventually I and another member were able to convince everyone to let the "govvies" come to our "secret" location.

At the end of the night, after a meeting and a dinner together, we walked the government delegation out to the parking lot and said our goodbyes. We watched them as they walked toward their cars, concerned to make sure all of them actually drove away. So our paranoia spiked as we saw them stop and chat with each other.

I briskly walked over to the huddle and interrupted them with an objection along the lines of: "You can't do that! You can tell all the secrets you want once you are back in your offices, but we just let you into our house and extended a lot of trust and faith in doing so. So I want to know what it is you were just talking about!" It's amazing that a little bit of alcohol can provide enough courage to do this, given the people we were dealing with. Or perhaps I just didn't know any better at the time.

I think this stunned them a bit. Everyone in their group of about five high-level staff looked at one member who had not, up to that point, stood out in our minds as the senior person (nice operational security on their part). He gazed directly back at me and said, "We were just talking about what you have managed to put together here."

"What do you mean?" I pressed.

He replied, "All of the briefings we have received state that the sort of setup with the capabilities you have here is not possible without nation-state-type funding." I responded that it was obvious from what we had showed them that we had done it without any money (it should be noted that it is a great oversight to underestimate the capabilities of inquisitive people who are broke). "We were further wondering," he said, "if any governments have approached you or attempted to 'hire' you." So in my typical fashion I responded, "No. Well, at least not that I'm aware of. But if you'd like to be the first, we're willing to entertain offers...."

Even with this poor attempt at humor, we ended up getting along.

But despite the fear on both sides and the communication problems that resulted from our radically different viewpoints, the government team left understanding that our exploits had truly been achieved by a group of hobbyists with spare time and almost no money.

The visitors were the people who received reports and briefings from various three-letter agencies. They were aware of how the career ladder at these agencies could be conducive to confirmation biases. Assured by officials that our achievements required funding on a scale that could only be achieved by specific classes of adversaries, they took the bold step of

Copyrighted material

searching us out so that they might refute some of the basic beliefs they had been taught. They went so far as to visit the dingy L0pht and ended up modifying their incorrect assumptions about how much effort an adversary might really need to pull off some pretty terrifying cyber-acts.

Unfortunately, there are not as many people as one might like who are either able or willing to seek out uncomfortable evidence to challenge assumptions. When testing software and systems, it is important to consider the environment in which engineers, developers, and testers might be working and the preconceived notions they might bring. This is particularly important in regards to what their application might be asked to do or what input might be intentionally or unexpectedly thrust at them.

Functional Fixation

Functional fixation is the inability to see uses for something beyond the use commonly presented for it. This is similar to the notion of first impressions—that the first spin applied to initial information disclosure (e.g., a biased title in a newspaper report or a presentation of a case by a prosecutor) often permanently influences the listener's ongoing perception of the information.

When someone mentions a "hammer," one normally first thinks of a utilitarian tool for construction. Few people think first of a hammer as an offensive weapon. Similarly, a flame-thrower elicits images of a military weapon and only later, if at all, might one think of it as a tool to fight wildfires through prescribed burning tactics that prevent fires from spreading.

Functional fixation goes beyond an understanding of the most common or "default" use of a tool. We call it fixation when it leaves one thinking that one knows the *only* possible use of the tool.

Consider a simple quarter that you find among loose change in your pocket. If someone asks you how to use it, your first response is probably that the coin is used as a medium of exchange. But, of course, people use coins in many other ways:

- A decision-maker
- A screwdriver
- A projectile
- A shim to keep a door open
- An aesthetic and historic collectible

Ignoring these alternative functions can surprise you in many ways, ranging from offers to buy your old coins to a thunk in the head after you give a quarter to a young child.

Copyrighted material

Vulnerability in Place of Security

Now that you have a general understanding of functional fixation, you might be wondering how it relates to computer and network security.

Many people think of security products such as vulnerability scanners and anti-virus software as tools that increase the security of a system or organization. But if this is the only view you hold, you are suffering from functional fixation. Each of these technologies can be very complex and consist of thousands of lines of code. Introducing them into an environment also introduces a strong possibility of new vulnerabilities and attack surfaces.

As an example, during the early years of vulnerability scanners, I would set up a few special systems on the internal networks of the company that I worked for. These systems were malicious servers designed to exploit client-side vulnerabilities in the most popular vulnerability scanners at the time. Little did I realize that client-side exploitation would become such a common occurrence in malware infection years later.

As one example, the ISS scanner would connect to the finger service on a remote system to collect remote system information. However, the scanning software had a classic problem in one of its security tests: the program did not check the length of the returned information and blindly copied it into a fixed-size buffer. This resulted in a garden-variety buffer overflow on the program's stack. Knowing this about the scanner, and knowing the architecture of the system the scanner was running on, I set up malicious servers to exploit this opportunity.

When the company I was employed by would receive their annual audit, as a part of evaluation the auditors would run network vulnerability scans from laptops they brought in and connected to the internal network. When the scanner would eventually stumble across one of my malicious servers, the scanning system itself would be compromised through vulnerabilities in the scanning software.

This often resulted in humorous situations, as it gave the executives of the company some ammunition in responding to the auditors. Since the compromised auditor system had usually been used for engagements across multiple clients, we could confront them with audit information for other companies that were now exposed by the auditors' systems. The executives could justifiably claim that vulnerabilities found on our internal systems (living behind firewalls and other defensive technologies) were not as severe a risk to the corporation as disclosure of sensitive information to competitors by the auditors themselves—made possible by the "security software" they used. Functional fixation might cause one to forget to check the security of the security-checking software itself.

Modern anti-virus software, unfortunately, has been found to include all sorts of common programming vulnerabilities, such as local buffer overflows, unchecked execution capabilities, and lack of authentication in auto-update activities. This security software, therefore, can also become the opening for attackers rather than the defense it was intended for.

Copyrighted material

The preceding examples are straightforward examples of functional fixation and can be attributed to the same naïveté I discussed in the section on learned helplessness. However, there are more subtle examples as well.

Sunk Costs Versus Future Profits: An ISP Example

One of the greatest hampers to security springs from negative perceptions of security requirements at a high corporate level. Some of these represent functional fixation.

Several months before the historic Distributed Denial of Service (DDoS) attacks that temporarily shut down major service providers and commercial entities (including eBay, CNN, Yahoo!, and others) on the Internet,* I had the opportunity to analyze backbone router configurations for a Tier 1 ISP. The majority of the IP traffic that transited these core routers was TCP traffic, in particular HTTP communications. A much smaller percentage was UDP, and well below that, ICMP. I was surprised to discover that the routers lacked any controls on traffic other than minimal filters to prevent some forms of unauthorized access to the routers themselves. But when I suggested that the core router configurations be modified toward the end of protecting the ISP's customers, the expression of surprise shifted to the company's executives, who immediately told me that this was not an option.

Two schools of thought clashed here. The ISP did not want to risk reducing the throughput of their core routers, which would happen if they put any type of nontrivial packet filtering in place. After all, an ISP is in the business of selling bandwidth, which customers see as throughput. Router behavior and resulting throughput can be negatively impacted when the systems moving packets from point A to point B have to spend any extra time making decisions about how to handle each packet.

Furthermore, neither the ISP nor its customers were suffering any adverse effects at the time. The managers could understand that there might be an attack against their own routers, but were willing to wait and deal with it when it happened. To spend money when there was no problem might be wasteful, and they would probably not have to spend any more money on a future problem than they would have to spend now to proactively keep the problem from happening. Attacks on customers were not their problem.

On my side, in contrast, although there had not been a widespread instance of DDoS at this point in time (in fact, the phrase DDoS had yet to be coined), I was aware of the possibility of network resource starvation attacks against not only the ISP's routers but also the customers behind them. I knew that attacks on customers would be hard to diagnose and difficult to react to quickly, but I entirely failed to convince the ISP. In fact, I had to concede that from a business standpoint, their reasons for not wanting to further secure their systems was somewhat logical.

* "Clinton fights hackers, with a hacker," CNN, February 15, 2000 (*http://web.archive.org/web/20070915152644/http://archives.cnn.com/2000/TECH/computing/02/15/hacker.security/*).

Copyrighted material

(The problem of security as a cost rather than a revenue generator is also examined in Chapter 12, *Oh No, Here Come the Infosecurity Lawyers!*, by Randy V. Sabett.)

Some time after the wide-scale DDoS attacks, I was honored to find myself sitting at the round table in the Oval Office of the White House only a few seats down from President Clinton. The meeting had been called to discuss how government and industry had handled the recent DDoS situation and what should be done going forward.

And once again, I was surprised. The main concern expressed by executives from the commercial sector was that the attacks might prompt the government to come in and regulate their industry. They seemed uninterested in actually understanding or addressing the technical problem at hand.

Then it started to dawn on me that the ISPs were functionally fixated on the notion that government intervention in these sorts of matters is likely to negatively impact revenue. This was the same fixation that I had witnessed when interacting with the large ISPs months earlier in regards to placing packet filters on their core routers: that security costs money and only prevents against future potential damage. They never considered ways that implementing security could *create* revenue.

After the meeting, I reengaged the executive of the large ISP I had previously dealt with. I told him that I understood why he made the security decisions he had and asked him to give me an honest answer to a question that had been on my mind lately. I asked him to suppose I had not approached him from a security standpoint. Instead, suppose I had pointed out that the ISP could negotiate committed access rates, use them to enforce caps on particular types of traffic at particular rates, take these new certainties to better plan utilization, and ultimately serve more customers per critical router. Further, they could use such a scheme to provide different billing and reporting capabilities for new types of services they could sell. The filtering and measurement would prevent inappropriate bandwidth utilization by the client, but any useful traffic the client found to be blocked or slowed down could be satisfied by negotiating a different service level.

But as a side effect, the same filtering would dramatically reduce inappropriate bandwidth utilization by external acts of malice. Would this, I asked, have been a better approach?

The answer was a resounding *yes*, because the company would view this as an opportunity to realize more revenue rather than just as an operational expense associated with security posturing.

I learned from this that I—along with the vast majority of practitioners in my field—suffered from the functional fixation that security was its own entity and could not be viewed as a by-product of a different goal. As so often proves to be the case, architecting for efficiency and well-defined requirements can result in enhanced security as well.

Copyrighted material

Sunk Costs Versus Future Profits: An Energy Example

Part of my career has involved examining in detail the backend control systems at various electric utilities and, to a somewhat lesser extent, oil company backend systems. I assessed how they were protected and traced their interconnections to other systems and networks. It was surprising how the oil and electric industries, while using such similar systems and protocols, could be operated and run in such widely disparate configurations and security postures.

To put it politely, the electric company networks were a mess. Plant control systems and networks could be reached from the public Internet. General-purpose systems were being shared by multiple tasks, interleaving word processing and other routine work with critical functions that should have been relegated to specialized systems to prevent potential interference or disruption of operations. It appeared in several cases that systems and networks had been put together on a whim and without consideration of optimal or even accurate operations. Implementers moved on to the next job as soon as things worked at all. Many plant control networks, plant information networks, and corporate LANs had no firewalls or chokepoints. From a security standpoint, all this combined to create the potential for malicious interlopers to wreak serious havoc, including manipulating or disrupting the physical components used in the production and transmission of power.

Conversely, the few offshore oil systems that I had looked at, while utilizing similar SCADA systems, were configured and operated in a different fashion. Plant control and information networks were strictly segregated from the corporate LAN. Most critical systems were set correctly to have their results and status handled by a librarian system that then pushed the information out in a diode fashion to higher analysis systems. Concise and efficient network diagrams resulted in crisp and clean implementations of SCADA and DCS systems in the physical world, including restriction of access that resulted in effective security. In many cases the components were custom systems designed and configured to perform only specific functions.†

The contrast between the electric and oil organizations intrigued and worried me. As fate would have it, I was in the position to be able to call a meeting about this subject with some high-ranking technical people from electric companies, oil companies, and government (think spook) agencies.

The first salient aspect that surprised me from the meeting was that the people from the electric utilities and their electric utility oversight and clearinghouse organizations did not refute my statements regarding the poor—or completely missing—security on their networks and systems. This surprised me because the electric companies were publicly denying that they had

† It is important to note that I analyzed only a subset of all the oil and electric systems out there. The differences are put forth here for comparison purposes to help illustrate functional fixation and how it affects corporate views of *security*. The oil industry has its fair share of incorrectly configured systems and environments, as do almost all large industries. Similarly, there are probably some well-configured electric company plant information and control networks…somewhere.

Copyrighted material

any cyber-system risk. In our meeting they pointed out some examples where security had been implemented correctly—but they acknowledged that these examples were exceptions and not the norm.

My second surprise came when the oil companies stated that they did not go about designing their systems from a security perspective at all, and that although security was important, it was not the business driver for how things were configured. The primary driver was to have an edge against their direct competitors.

If company A could make a critical component operate at 5% greater efficiency than company B, the increased operational capacity or reduction in overhead rewarded company A over time with large sums of money. Examples of how to increase such efficiency included:

- Forced separation and segregation of systems to prevent critical systems from incurring added latency from being queried by management and reporting requests
- Utilizing special-purpose systems designed to accomplish specific tasks in place of general-purpose nonoptimized systems

These efficiencies benefited security as well. The first created strong, clean, and enforceable boundaries in networks and systems. The second produced systems with smaller surface areas to attack.

Enforceable network and system boundaries are an obvious effect, but the case of smaller surface areas deserves a brief examination. Imagine that you have a general-purpose system in its default configuration. The default configuration might have several services already configured and running, as well as many local daemons executing to assist user processing. This allows the system to be deployed in the largest number of settings with minimal reconfiguration required. The systems' vendor prefers such systems with broad capabilities because they make installation easier.

However, this doesn't mean that the default setting is *optimal* for the majority of consumers, just that it is *acceptable*. In the default setting, each of the running services is an attack surface that may be exploited. Similarly, client applications may be compromised through malicious input from compromised or falsified servers. The more services and client applications that are running on the system, the greater the attack surface and the greater the likelihood that the system can be remotely or locally compromised.

Having a large attack surface is not a good thing, but the drawback of generality examined by the oil companies was the systems' suboptimal performance. For each running program, which includes server services as well as local applications, the kernel and CPU devotes processing time. If there are many running applications, the system has to time-slice among them, a kernel activity that in itself eats up resources.

However, if there are few running applications, each one can have a greater number of CPU slices and achieve greater performance. A simple way to slim down the system is to remove superfluous services and applications and optimize the systems to run in the most

Copyrighted material

stripped-down and dedicated fashion possible. Another way is to deploy systems dedicated to specific functions without even the capability of running unrelated routines. These tactics had been used by the oil companies in the offshore rigs I had examined in order to maximize performance and thus profits.

Why hadn't the electric utilities gone through the same exercise as the oil companies? At first, electric companies were regulated monopolies. Where these companies did not need to be competitive, they had no drive to design optimized and strictly structured environments.

One would be tempted to assume that deregulation and exposure of electric companies to a competitive environment would improve their efficiency and (following the same path as oil companies) their security. However, the opposite occurred. When the electric companies were turned loose, so to speak, and realized they needed cost-cutting measures to be competitive, their first steps were to reduce workforce. They ended up assigning fewer people to maintain and work on the same number of local and remote systems (often through remote access technologies), focusing on day-to-day operations rather than looking ahead to long-term needs. This is usually a poor recipe for efficiency or security.

The story of the oil companies confirms the observation I made in the previous section about the ISP. Most organizations think of security as a sunk cost, insofar as they think of it at all. Security approached in this fashion will likely be inadequate or worse. If, however, one focuses on optimizing and streamlining the functionality of the networks and systems for specific business purposes, security can often be realized as a by-product. And once again, security professionals can further their cause by overcoming their functional fixation on security as a noble goal unto itself worth spending large sums on, and instead sometimes looking at sneaking security in as a fortuitous by-product.

Summary

In this chapter, I have offered examples of classic security failures and traced them beyond tools, practices, and individual decisions to fundamental principles of how we think. We can improve security by applying our resources in smarter ways that go against our natural inclinations:

- We can overcome learned helplessness and naïveté by ensuring that initial decisions do not shut off creative thinking.
- We can overcome confirmation traps by seeking inputs from diverse populations and forcing ourselves to try to refute assumptions.
- We can overcome functional fixation by looking for alternative uses for our tools, as well as alternative paths to achieve our goals.

All these ventures require practice. But opportunities to practice them come up every day. If more people work at them, this approach, which I'm so often told is unusual, will become less curious to others.

Copyrighted material

CHAPTER TWO

Wireless Networking: Fertile Ground for Social Engineering

Jim Stickley

By now, everyone has heard the security concerns about wireless devices. They have been an area of concern for many security professionals since the original Wi-Fi release in 2000. As early as 2001, the standard Wired Equivalent Privacy (WEP) access protocol, designed to keep unwanted users from accessing the device, was discovered to have fundamental flaws that allowed security to be bypassed within a couple of minutes. Although security was greatly increased in 2003 with the release of Wi-Fi Protected Access (WPA), most paranoid system administrators still had their doubts. Sure enough, with time new exploits were discovered in WPA as well. Although it is not nearly as dangerous as WEP, it left many administrators feeling justified in their concerns.

However, while one camp has remained skeptical, others have seen the operational benefits that come with wireless and have embraced the technology. For example, handheld devices carried throughout a department store allow employees to accomplish inventory-related tasks while communicating directly with the organization's servers. This can save a tremendous amount of time and increase customer service satisfaction. Wi-Fi has reinvigorated the use of public spaces from cafés to parks around the world. Unfortunately, several attack scenarios remain largely unknown and could feed an epidemic of corporate and personal identity theft.

This chapter begins with a story of how I, a professional security researcher, probed wireless security flaws in the wild and discovered the outlines of the threat they present. Then I'll return to the state of Wi-Fi and the common ways it undermines organizational security.

Copyrighted material

Easy Money

Here's an everyday attack scenario. You're on a layover at a major airport in the United States. As you scan the departure monitors checking for your gate, your eyes focus on the words every traveler dreads: "Delayed." Just like that, you have become one of the many refugees who will be spending the next six hours enjoying all the comforts and amenities of the airport.

You head over to your gate and start searching for an electrical plug to boost up your laptop's dying battery. I have done this search many times, slowly walking the whole area trying to spot the plug that might be tucked behind a row of seats or on the backside of a pole. You can always spot the guy searching for this elusive plug as he walks by, staring mainly at what looks to be your feet while trying not to be obvious. I assume it was probably similar to the caveman's quest for fire. Everyone wants it, only a few can find it, and once you have it you become extremely protective of it. In fact, on more than one occasion when others have come near, I have grunted and beaten my chest to show dominance.

Now, assuming you are the alpha male who found the plug, you pop open your laptop, plug it in, and immediately start searching for wireless access. Most airports, hotels, coffee shops, and even parks now offer wireless access service. You simply turn on your laptop, click the wireless access icon, and up pops one or more access points from which to choose. As you scan through the list you see an access point titled "T-Mobile." It turns out this particular airport has partnered with this hotspot service, so you select it without giving it a second thought. A couple of seconds later, you open a web browser. Instead of your home page, you are automatically redirected to the T-Mobile page, where you are given the option to sign in using your existing T-Mobile account or create a new one.

Since you don't have an account, you click to create a new one, only to find that the price is $9.99 for a day. While that's not a horrible price, you did notice there were a couple of other wireless access points available, so you decide to quickly check whether any of them happen to be free. You click on the wireless icon again and see a list of three other wireless access points. Two of them are locked and require the correct key to access them, but one titled WiFly is open. You select WiFly, and this time the page is redirected to the WiFly login page offering access for just $1.99. Pleased that you just saved eight bucks, you pull out your credit card and fill out the online form. You click Submit and, voilà, you are now browsing the Internet.

With nothing else to do, you decide to check your email via the online web interface. You type in the URL to the website and press Enter. Immediately an error message pops up stating there is a problem with the website's security certificate. A security certificate is used when you browse to any site that offers encryption. You will recognize that a site is using an encrypted session because the web link starts with *https://* instead of *http://*.

In addition, you will see the closed lock in the status bar on your web browser that indicates the page is encrypted. However, the pop-up error message indicates that the security certificate

Copyrighted material

was not issued by a trusted certificate authority and that the website you are visiting doses not match the certificate.

You now have the choice to either close the page or continue to the website. You think about it for a second and assume maybe you went to the wrong page, so you choose to close it. You open a new browser and try again, and the same error message pops up. Realizing that you are at the correct page and that something else is wrong, you make the assumption that this probably has something to do with the service provider you are using at the airport, and so you click to continue. The page that comes up looks normal, and so you log in to check your email. You continue to browse over the next several hours, and while that error pops up a few more times, everything else seems to be fine.

Finally your plane arrives, you pack up your laptop, leave your electrical outlet to one of the other cavemen who has waited patiently nearby to make his move, and head off to your final destination.

A few weeks pass and you are back at home paying bills. You open your credit card statement and discover with a shock that your credit card, which previously had a balance of a couple hundred dollars, has now been maxed out. With some concern, you ask your wife if she may have gone on a shopping spree. To your relief, she hasn't, and yet still there are all these charges. Of course, by now you have completely forgotten about that day back at the airport where you chose to submit your credit card information to that cheap wireless access company. Unfortunately for you, it turns out that this cheap wireless access company was really an identity thief.

Setting Up the Attack

As part of my job, I am hired to go out and put security scams to the test. I have performed this particular attack numerous times throughout the United States. In every case I have gained access to credit card information. Although the scam may seem complicated, in fact the ease with which it can be performed is what makes it so worrying.

Before going to the venue where I want to launch my attack, I create a credible, professional-looking login page for a fictional company such as WiFly. It offers a form for people to fill out with their credit card information.

Upon reaching the venue, I open a garden-variety laptop and purchase the real Internet access offered at that location. In locations lacking an Internet provider, I just use my mobile device to gain access via cellular. Even if the speed is slow, it really doesn't matter because by the time the victims are using the access, they will already have been scammed.

Next, I set up a wireless access device connected to my laptop. Depending on the location, I may set up a large antenna to cast a wider net.

Copyrighted material

Finally, I run software I wrote that passes traffic from the computers of unsuspecting victims through my wireless access device, then through my laptop, and ultimately out to the Internet through the connection I paid for. With everything in place, I am the spider waiting for the fly.

Eventually the victims begin to arrive. They choose to connect to the low-cost wireless access point, hit the web page, submit their credit card information, and in a snap they become (should I so choose) another identity-theft statistic.

A Cornucopia of Personal Data

Obviously, gaining access to the credit card information is useful to an identity thief, but there are even more concerns with this type of attack.

Remember the security warning that popped up about the certificate? The warning popped up because any traffic being encrypted was actually being decrypted at my laptop, not at the final destination as the user assumes. In other words, they're running a secure, encrypted connection just as they want—except the encryption is using my certificate and I can trivially decrypt the data again. As the man in the middle, I can decrypt users' data, record everything, and then reencrypt it and pass it along to its final destination. I could record usernames, passwords, email messages, and other potentially confidential information that the victim assumed was being passed securely to a trusted destination.

Even a small slice of your personal networking traffic can open a chink for serious identity attacks. Say, for example, that I gain access only to your email account. I now start watching all email messages that you receive. Think about all the crazy information that gets passed through email.

If you don't use a web-based email service, you might think there's a safe barrier between your email account and your web accounts. But most websites offer a "forgot your login" option. Simply give them your email address and they will email you your password. By checking someone's incoming email long enough for telltale strings, I can generally gain access to the login name and password for anything, including online banking accounts.

A good identity thief knows that the keys to the kingdom are sometimes spread around. Sometimes you have to collect data from several sources before you have enough. Like a puzzle, each piece of data you obtain can then be used to go after another piece of data. In some tests, I put up a wireless access point without charging anything. In these cases I gave up the opportunity to sniff the user's credit card information right away, but I attracted many more users who probably felt even safer because they didn't have to give me that information. My only goal in these tests was to record all the data that passed through. Within a few hours I recorded everything from login names and passwords to online purchases that included names, addresses, and credit card information.

Often, people who live in apartment buildings will attempt to get on the Internet without buying their own service by jumping on an open wireless device they discover, under the

Copyrighted material

assumption that a neighbor put it up. They think they are getting a free ride, but in reality they are getting scammed day in and day out by the thief who put up that access point for the sole purpose of ripping people off.

Precisely because wireless access points are so prevalent and so much a part of our landscape, most people don't think about where that wireless is coming from. At a shopping mall, you can feel reasonably confident that a fine watch you buy from a jewelry store is real. However, if you were driving down the street and saw a guy selling watches out of the trunk of his car, you're probably not going to make the same assumption. It's fairly easy to see the difference between the two and know what looks safe and what looks like a scam. But with wireless access, you have no idea who is offering the goods. Even if the name says T-Mobile and the login page looks exactly like a real T-Mobile login page, how do you know that this particular site is run by T-Mobile? Anyone can put up a wireless access point and put up any page they like.

A Fundamental Flaw in Web Security: Not Trusting the Trust System

One of the major reasons for the success of my attack revolves around the handling of security certificates by both users and servers. Most people don't pay that much attention to security warnings that show up when browsing the Internet. There are two main reasons for this.

Some people just don't understand what these warnings mean. I like to use the example of my parents. They are intelligent people at the age of retirement who, like the rest of the world, have embraced the Internet experience. However, they have never been trained professionally about the Internet and its risks, nor do they work for a corporation that has engineers dedicated to making sure employees understand what all the security warnings mean. Instead they have felt their way through the process and have just enough knowledge to pay their bills online and check their stock portfolio. Something like a digital certificate makes about as much sense to them as a proton accelerator.

On the other hand, I find technically savvy people who have a comprehensive understanding not only of digital certificates but also of man-in-the-middle attacks. One might think these people would never fall for such a scam, but on the contrary, I have found that even these people are quick to fall victim. The reason is that—unlike my parents, who don't understand anything—the experts understand it so well that they rationalize what is taking place.

For example, when the security alert pops up their first assumption is that the administrator for the WiFly site has dropped the ball and didn't renew her expired certificate. Or they assume that because the site is being offered in a coffee shop or hotel, the person who set it up wasn't well trained technically and simply set up the certificate improperly. In addition, you will often come across intranets where the security certificates have long since expired, but IT tells the employees to just ignore the security warnings and use the site. This type of conditioning teaches average users that the certificates are not important, whereas the more advanced users,

Copyrighted material

while aware of the certificates' importance, simply become desensitized to the point where they just don't pay attention.

When I interview victims after these attacks, a very common answer for their behavior was that they have just stopped paying attention to all the security-warning pop ups. Many seem to be jaded by operating systems (the worst offender in this area being Microsoft's Vista) that present you with so many security warnings every day that they become white noise.

Establishing Wireless Trust

Despite the risks of using wireless access points, they are obviously very convenient and I'm not suggesting everyone stop using them. Society just needs to find ways to reduce the risks without reducing the convenience.

The most obvious thing a user can do to protect himself is to pay attention to security alerts. If you are using a wireless access point and receive a warning that the security certificate has a problem, you need to stop immediately. Although it's true that some website administrators make mistakes and don't update their certificates, the risks of continuing are far greater than bailing out.

Of course, this is also a strong argument for better management of certificates by organizations. The digital certificate is one of the few things a corporation can do to give an end user any sort of confidence in the site's security. If the certificate is not properly maintained, it causes skepticism about the rest of the security.

If the site requires a credit card number, another simple trick you can use to check the authenticity of the site is to submit a bogus credit card number. If the site is legitimate, the site will check the card number and notify you that the transaction failed. If it is a malicious site, it is not going to run the card, and will just accept whatever you give it. There are no guarantees with this trick, but it is better than nothing.

My last recommendation is to avoid using public access points to conduct confidential transactions. The coffee shop might not be the best location for logging in and checking the balance on your bank account. At the airport, logging in to sell some stocks should probably be put on hold. Being a full-time traveler myself, I understand that sometimes you have no choice, and you may be stuck having to pass confidential information to whatever wireless service you can find. If that is the case and you have even the slightest concerns about what you have entered, change your login credentials on any website you visited while using the wireless site immediately upon getting back to a trusted location. Again, it's not a guarantee that you didn't already become a victim, but it can't hurt.

Adapting a Proven Solution

Although being paranoid about your information security can help, it doesn't truly eliminate the risk of these types of attacks. For that to happen, the industry needs to solve the trust

Copyrighted material

relationship. When I open my wireless device software and search for available access points, I need to feel confident that a hub called "T-Mobile" really does belong to T-Mobile.

To do this, the access point and the client need to be able to exchange verifiable information that securely identifies both of them. Although coordinating the implementation of this mutual trust in the different systems may be difficult, the technology itself is rather simple and has been around for years. The same certificate technology that has been used on websites could be put to use on the wireless access devices.

The solution would work something like this. The user opens a wireless client and receives a list of available access points. The client would check the digital certificate assigned to each device. Each certificate would be based on the displayed name, the company name, and a hardware identifier, as well as the certificate authority who signed the certificate. As the devices are displayed via the wireless client, they indicate whether they are valid (see Figure 2-1).

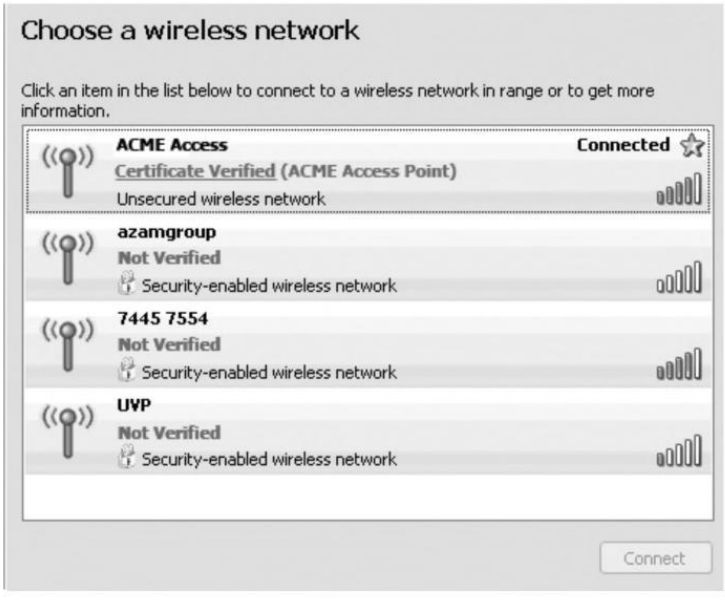

FIGURE 2-1. Hypothetical display of certified wireless hubs

While there are numerous ways that the Certificate Authorities could be established, I imagine that the easiest way would be to use the same Certificate Authorities that are trusted for web certificates. Using an already established group would make this particular area of development much easier, and with the infrastructure already in place, the crossover development would be reduced.

I realize that some malicious hubs will post bogus certificates and some users will never pay attention to the difference between a validated and nonvalidated access point. But as people

Copyrighted material

become more educated about the solution, and as the software evolves to help point people in the right direction, the risk of these types of attacks would be greatly reduced.

Wireless Gone Wild

Having charted out the next frontier of wireless attacks and ways to combat them, I'll take a step back to examine the well-known problems with Wi-Fi security and the social conditions of its use that make it an everyday danger.

Network users and administrators who are used to Ethernet-based LANs have trouble grasping the relative uncontrollability of wireless. A LAN uses very insecure protocols (sniffing and altering traffic, masquerading as another system, and carrying out denial-of-service attacks are all trivial), but the physical limitations of the cable are forgiving; it's difficult to tap into the cable and attach a rogue system. Wireless deliberately removes this layer of physical security we so often take for granted, and allows traffic to spill out, even across such physical boundaries as walls and fences.

As I mentioned at the beginning of this chapter, administrators originally used WEP (if they were cautious enough to use any security) to secure access points. The main problem with WEP was that a hacker could simply snoop packets that were in the air and extract the keys to gain access. Numerous tools were created to allow even the most novice hackers to perform these attacks.

WPA was introduced to resolve the security shortcomings of WEP by closing the loophole that allowed the key to be extracted rapidly from snooped packets. For the moment, wireless engineers were happy.

Of course, the joy was short-lived when it was discovered that passphrases used for WPA's Pre-Shared Key (PSK) could still be discovered. WPA starts with an exchange of packets, known as a *handshake*, that verifies the client to the access point. Because data is encrypted, no one can use it to derive the key to break into the system, even by monitoring traffic near the hub and recording the entire handshake. However, the hacker can store that recorded data and perform a password grinding or a brute-force attack on it, which involves taking common words from an extremely large word list and encrypting each one to find which one matches the recorded data.

For a hacker to be successful, a few conditions need to be in place. The most obvious is that the hacker must be able to receive the wireless signal from the access point. I have come across a number of organizations that have thought of this as a primary form of defense. In some cases, the administrator has buried the access point deep inside a facility with the idea that the signal would be too weak to pass its walls. Although it's true that the average laptop using an internal wireless device would not be able to pick up the signal, any determined hacker will own a far more powerful antenna that can pick up a much weaker signal.

Copyrighted material

Next, the hacker needs to be monitoring the beginning of the session between the client and the wireless access point. Simply monitoring the traffic once the session is already established is of no use. Although this might sound like this requirement greatly reduces the odds of a hacker gaining that initial handshake information, in reality it doesn't help much at all. It turns out that a number of tools have been created that are designed to end a wireless session by faking the client and sending a command to terminate the session. Upon disconnect, the client will generally attempt to reconnect. At this point the hacker will be watching the traffic and now has the initial handshake.

The last major criterion is the strength of the passphrase itself. My tests have turned up access points which such simple phrases as *tootired* or *bicycles*. Password-cracking software can discover these in mere minutes. A WPA passphrase can range from 8 to 63 characters, but I find that most people generally end up using only 8 or 9. Of course, the best possible passphrase someone could deploy would be 63 characters, mixing letters, numbers, and punctuation. So why doesn't every administrator do this?

The main reason is that whatever passphrase you choose for your access point needs to be typed into the wireless client on every computer that will be connecting. The support costs for a really crazy passphrase become a nightmare, as people mistype characters or forget the passphrase altogether. So instead, administrators go to the opposite extreme and make the passphrase as easy as possible. Although there is no perfect solution, my suggestion is to find a happy medium. A passphrase such as "This1was900yearsold!!!" is relatively easy to remember and would be far more difficult to crack than standard dictionary words.

Wireless As a Side Channel

Worrying about bogus wireless access points when you're on the road is one thing, but system administrators tasked with securing their networks have even more to be concerned about.

Some organizations decided long ago that the risk of Internet web use is just too great for their network, and therefore have blocked access to web surfing completely. In the past this solution seemed to work well, but more recently, with the proliferation of open wireless access points popping up everywhere, the threat surfaces anew. Users have discovered that they can bring a USB wireless device to work or use the existing wireless in their laptops and then log on to another, nearby organization's wireless network.

Using others' wireless connections became popular with hackers about five years ago with the advent of *warchalking*. This term referred to the marks left by hackers on the sides of buildings that had open wireless networks. Subsequently arriving hackers could log into these networks from their laptops on the street.

As wireless became more popular, less technically savvy users started putting this technique to use. The problem administrators are now facing is that these users do not understand the potential risk they are placing upon their own organization's network by using this newfound access.

Copyrighted material

Any user bypassing the main security infrastructure of her own organization to access the Internet through a secondary device is now at the mercy of whatever security the other organization implements. The lack of security already shown at this company in their wireless network is generally not a sign of good things to come. Viruses and worms that might be blocked from entry into your organization have a new avenue through this wireless access point.

Some users set up their laptop or computer so they are plugged into their local company's network and subsequently connect by wireless to the other organization. This design creates a potential conduit between the two networks that directly compromises the entire network security at the more secure organization and negates the majority of its precautions.

Hackers are also aware of how corporate users use open wireless connections to gain Internet access. For that reason, hackers have started setting up bogus wireless access points near sensitive sites, attempting to obtain corporate information.

Their attack is basically simple. Place a wireless access point in a building with Internet access. In most cases, this is extremely easy because most small companies have no real controls on the network and do not know when an employee has installed such a device. Next, boost the signal strength with a modified antenna to reach the largest possible audience. Then, write a small program that watches for any activity on the wireless device. As soon as there is activity, notify the hacker, who begins to attack the computer that has logged onto the wireless network. If the computer is vulnerable, the hacker will allow the user to continue to use the Internet access while the hacker silently gains access. In most cases, spyware and trojans are quickly loaded onto the unsuspecting user's computer.

Once a system is compromised, the sky is the limit regarding the types of information the hacker can obtain. Systems connected to the organization's internal network as well as the hacker's access point are pure gold. Although some routing issues come into play when first attempting to access the user's network, even low-level hackers are able to bypass that problem within a couple of minutes.

What about the corporation's internal network monitoring? Depending on what kind of security has been put into place, they may never know what is happening until long after the damage has been done.

What About the Wireless Access Point Itself?

When I mention TJX, what pops into your mind? Odds are, if you have even casually caught the mainstream news over the past two years, the first thing you think of is the department store credit card numbers that were stolen from this company. It was discovered in December of 2006 that over the past two years, hackers had breached their network and systematically downloaded a minimum of 45.7 million credit card numbers—and there is speculation that the number is probably closer to 200 million. (The TJX breach is covered in detail in Chapter 3, *Beautiful Security Metrics*, by Elizabeth A. Nichols.) While TJX continues to lick its wounds

Copyrighted material

from the fallout and experts are predicting that the total cleanup costs will tip the scales at a billion dollars, it turns out that many more organizations are operating day to day with the exact same security flaws: unprotected wireless access points.

In November 2007, the Motorola AirDefense division, which offers security and compliance monitoring for wireless networks, released a study examining the security of wireless devices at 3,000 stores nationwide. The study revealed that 25% of the devices were not using any encryption at all, and another quarter of the rest were using only the old, vulnerable WEP connection protocol.

It's frightening to still find such sloppy security years after the well-publicized TJX case. One quarter of the stores tested had less security than TJX, while a quarter of the remaining stores mustered only an easily bypassed security matching that of TJX.

Organizations that decide to take advantage of the convenience of wireless need to make sure they not only understand all the risks involved, but also diligently maintain the security necessary to support these devices. TJX, when it first deployed its wireless hubs, had implemented the security available at that time. Unfortunately, that security became quickly outdated. Had the company simply taken the time to upgrade to a properly deployed WPA design, it's probable that most of us would never have heard of TJX.

Still, Wireless Is the Future

From hotels and airports to corporate office buildings and supermarkets, the demand for wireless access continues to grow. Much as with the Internet itself, security risks will continue to be exposed. How organizations, administrators, and even average users respond to these security threats remains the question. Using open wireless access points is risky, and users need to be aware of these risks and respond accordingly. In addition, wireless access points can allow for major security breaches when not properly secured, as TJX discovered the hard way. If you are going to use newer technologies, you must be aware of all potential ramifications.

Copyrighted material

Copyrighted material

CHAPTER THREE

Beautiful Security Metrics

Elizabeth A. Nichols

When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is a meager and unsatisfactory kind; it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced to the state of science.

—William Thomson, Lord Kelvin, 1883

The revolutionary idea that defines the boundary between modern times and the past is the mastery of risk; the notion that the future is more than a whim of the gods and that men and women are not passive before nature. Until human beings discovered a way across that boundary, the future was a mirror of the past or the murky domain of oracles and soothsayers who held a monopoly over knowledge of anticipated events.

—Peter Bernstein, 1996

THE TWO QUOTES THAT START THIS CHAPTER CAPTURE THE ESSENCE OF BEAUTY in measurement and its primary deliverable: metrics. Lord Kelvin's message is that there is no science without metrics. Peter Bernstein's statement is about risk, which is conceptually related to security. Dr.

Copyrighted material

Bernstein states that metrics free you from the morass of being a prisoner of the past or, even worse, dependent upon fortune tellers—certainly key objectives of science, in general.

For these reasons in themselves, metrics are beautiful—at a conceptual level. What about in practice? In particular, what about the application of metrics for protecting individual, corporate, national, and global interests, especially where those interests rely upon an Information Technology (IT) infrastructure? The purpose of this chapter is to explore the beauty of applied security metrics. I'll start by using the medical field as an analogy. Then I'll turn to the domain covered by this book, computer security, and discuss metrics through examples. To do so, I'll dissect a small collection of highly visible and devastating security incidents. One is in the general area of defending against ATM fraud. Two additional contexts concern well-publicized incidents at two individual public companies: Barings Bank and TJX.

Security Metrics by Analogy: Health

Medical research has used metrics to advance both the science and practice of human health. Health, like security, is intangible. You can feel it but you can't touch it. Health, like security, is all about achieving the *absence* of something—namely the absence of failures in mental or physical well-being for health care, and the absence of failures in confidentiality, integrity, or availability for security. Many security practitioners cite these characteristics—intangibility and "proving a negative"—as reasons why a quantitative, rigorous, analytic approach to IT security is hard or impossible. Medical research is a shining counterexample. Therefore, some observations about metrics as applied to the field of medicine will be useful as a lead-in to our discussion on security metrics by example.

Let's begin by looking at the big questions—stated here as objectives for enlightenment that beautiful security metrics should deliver:

1. How secure am I?
2. Am I better off now than I was this time last year?
3. Am I spending the right amount of dollars, effort, or time?
4. How do I compare with my peers?
5. Could that happen to me?

The "that" in question 5 refers to any recent (usually negative) incident.

The first four questions were initially posed as a challenge for security metrics by Dr. Daniel E. Geer, Jr., a very prominent security "metrician"* whose doctorate is in epidemiology—a cornerstone of metrics used in public health care research. I added the fifth question.

* The term *metrician* is not a real word. I, and others, have coined it to refer to someone who performs quantitative analysis to identify requirements, models, and visualizations associated with metrics.

Copyrighted material

Dr. Geer says that he first heard questions 1–4 from the CFO of a large, public financial services company. In describing the experience, he recounts that there was a fair amount of disdain when answers were not forthcoming—providing yet more evidence that security as a discipline needs metrics to get the respect it deserves.

Mapping the previous security questions to what you might ask your medical doctor requires almost no rewording. Moreover, like security, one's health is a complex function of many factors: threats (such as germs and disease), vulnerabilities (such as genetic or environmental conditions), and the impact of their joint occurrence. Preventive health care, like IT security, aims to identify and implement measures that are designed to reduce the probability that a threat will be able to exploit one or more vulnerabilities. Medicine, like IT security operations, aims to identify treatments that reduce the impact when a threat manages to exploit a vulnerability.

Unreasonable Expectations

In both medicine and IT security, the previous questions are not answerable. In scientific terms, they are (charitably put) ill-formed. The first problem is definitions. What is the definition of being healthy or secure? Is a hangnail enough to render a person unhealthy? Or, analogously, is a single unpatched server enough to render a company's IT infrastructure insecure?

The second problem is context. How did the hangnail get there and is it really a problem? Does the patient have a history or set of characteristics that suggests anything? Does he chew his fingernails? If the hangnail is on the hand of a concert pianist, the impact will be different from the impact of one afflicting a nonmusical office worker. Or, for a vulnerability in an IT asset, the criticality, use, and connectedness (or popularity) of the asset should be considered.

The third problem is uncertainty. A hangnail has some nonzero probability of leading to gangrene and loss of limb or death; or, analogously, a single vulnerability has a nonzero probability of leading to a breach and catastrophic loss or a damaged reputation. Most people know enough about hangnails and their associated risks to treat them appropriately. In contrast, many executives have no idea how to think rationally about vulnerabilities in IT assets. They look to their CISO and CIO who (they perceive) are letting them down.

Why is this? The answer lies in factors that are in part cultural and in part due to lack of scientific rigor. Medical research is currently better at overcoming both factors than is IT security research.

Data Transparency

Let's first look at culture. The medical community has strong and institutionalized data-sharing mechanisms that are critical prerequisites for researchers to learn more and disseminate knowledge. Consider what happens when an emergent health threat is identified with suspected related deaths or illness. Investigators are summoned and inquiries are conducted

Copyrighted material

by experts from organizations with a wide range of specialties: the attending physician, consulting specialists, nurses, orderlies, immunologists from the Centers for Disease Control and Prevention (CDC), the insurer, drug manufacturers, and the hospital. Their findings may well be examined by journalists and politicians. They will also be widely discussed by health care providers and incorporated into medical school courses and other practitioner education to spread awareness quickly and accurately.

In stark contrast, let's look at the behavior of banks that operate 9ATM networks. In the late 1980s, New York banks discovered that printing the customer's full account number on ATM tickets enabled attacks, leading to substantial loss of money for both customers and banks. A thief could stand in an ATM line and obtain a legitimate customer's PIN by observing manual entry. The thief could then use both the PIN and account number (from discarded printed tickets) to steal money. For many years thereafter, non-New York banks continued to print account numbers on ATM tickets—many out of ignorance. As late as 1992, one bank in the UK was pilloried by the press for continuing this practice. While embarrassing, this publicity had the undeniable benefit that the bank immediately discontinued the outdated and insecure practice.

The value of data sharing is well understood by government and regulatory authorities. The Department of Homeland Security (DHS) has mandated Information Sharing and Analysis Centers targeted at industries such as Financial Services, Telecommunication, Power, and Manufacturing. Starting in early 2002 with California SB 1386 and spreading now to over 38 states, state laws require disclosure to a customer when personal data has been breached. This has resulted in a valuable online database called the Data Loss Database (DataLossDB), which security researchers are now beginning to mine for metrics about data loss size and occurrence by industry, type, and source. In particular, the trends regarding breach size and frequency, as well as insider versus externally perpetrated breaches, can now be characterized with metrics computed from accurate and (relatively) complete hard data.

Reasonable Metrics

Medicine's reasonable approach to measuring health gives patients a reasonable guideline for assessing their own health while helping them set reasonable expectations. Although patients do ask, "How healthy am I?" they know that it is essentially an unanswerable question, and most medical practitioners give a somewhat circumspect response like, "Everything looks good today." If pressed, the doctor may embellish this with "Your vital signs are normal. All your test results are normal. You are not overweight, but you should stop smoking"—or something similar. They give a summary that captures a list of unweighted positive and negative facts, all directly or indirectly derived from metrics. In some cases, they use medical terms such as cholesterol and osteopenia and are prepared to provide layman's definitions, if requested.

The point is that doctors don't attempt to give one a patient an all-encompassing score. Often, the list is organized into organ systems: cardiovascular, lungs, skin, nervous system, etc. Each

Copyrighted material

of these systems has its associated specialists that the patient can be referred to for focused, in-depth testing and diagnosis.

And, equally importantly, patients find such answers acceptable. Unlike the CFO in Dr. Geer's meeting, they don't expect a quick, deterministic score.

Moreover, the field of medicine has also developed an impressive collection of metrics that security metricians can learn from.

First, medical research calls for collecting "patient history," "vital signs," and "basic biometrics" to frame a discussion about health. Patient history provides extensive general context that is typically represented as a collection of yes or no answers, like a survey: "Have you ever had measles?" or "Did your mother, father, sister, or brothers have high blood pressure?"

Medical vital signs for most people include temperature, blood pressure, pulse, and respiratory rate. Basic biometrics are height, weight, and pain. For newborns, a special set of vital signs called the Apgar score was designed. This metric was devised in 1952 by Virginia Apgar, a pediatric anesthesiologist, as a simple, somewhat repeatable method to quickly and accurately assess the health of a newborn child immediately after birth. A baby is evaluated at one minute and at five minutes after birth on five criteria: appearance, pulse, irritability reflex, activity, and respiration. The Apgar score is obtained by assigning a value of 0, 1, or 2 to each of the five criteria and computing the total. A perfect score is 10. Along with guidance about assigning values for each criterion, there is also guidance on how to interpret the final score. A score over 7 is viewed as normal. Scores below 3 are critically low.

Second, medical research endorses relative as opposed to absolute measurements. The "pain" biometric mentioned earlier is a great example. Interestingly, pain is measured on a scale of 0 to 5 or a scale of 0 to 10, where a 0 means no pain and a 5 or 10 means the most pain the patient has ever felt. It is considered good practice for the physician to allow the patient to pick which scale (0–5 or 0–10) to use. Equally important is that this metric is measured relative to the patient's prior experience. There is no absolute scale. Since a given patient can't have experienced the pain of others, this approach is the only sensible option.

Third, medical research embraces the concept of "good enough." Positive vital signs and biometrics are probably neither necessary nor sufficient for someone to be healthy. A person can be sick with good scores or, conversely, be well with bad scores. But the latter situation is unusual and, in the spirit of "good enough," measuring vital signs and biometrics is still viewed as a "best practice." Moreover, through statistical analysis, medical research has produced levels of confidence that tell how likely it is that an exception will occur.

Finally, and probably most significantly, metrical researchers base their guidance on vast repositories of patient data, collected over decades. The values for medical vital signs have been empirically determined and validated by analyzing this data. This data is available because (1) patients readily agree to being measured and (2) patients agree to share those measurements. Analysis of freely shared patient data allows researchers to identify normal values with distributions that describe the likelihood that an individual will exhibit values different from

Copyrighted material

the norm and how to interpret the differences. Also, they can determine values for different populations, such as newborns, children, teenagers, adults, and the elderly. Analysis can identify other influences on normal values, such as gender, nationality, race, recent activity or lifestyle, whether the person is overweight, and more. Note that individual patient data is never revealed to either the doctors or patients during consultations, just the derived metrics: namely averages, percentile ranges, highs, lows, and standard deviations.

Culturally, the security profession needs to change in ways that emulate the medical profession. Security must develop a system of vital signs and generally accepted metrics that it understands and are "good enough." These metrics need not be deterministic nor absolute. They can be relative, perhaps even subjective at times. They can have a nonzero chance, but acceptably low and measurable, probability of being false negatives or false positives. IT security practitioners must embrace not only being regularly measured but also sharing data instead of perpetuating the cloak of secrecy that dominates their current behavior. If they don't change, mandatory fiats, such as the California SB 1386 mandate, will force them to share.

With these changes, security researchers can apply rigorous science to back up their analysis. Only then can they free themselves from being viewed with a level of credibility normally ascribed to soothsayers and fortune tellers.

Security Metrics by Example

Metrics have clearly helped medical practitioners by providing both a framework for quantifying the health of an individual or population and a collection of guidelines to communicate that state to nonexperts. Can metrics do the same for the security of an enterprise? As mentioned earlier, voluntary data sharing in this field is rare, but sometimes, when the consequences are sufficiently dire, information cannot be suppressed and the results of the ensuing investigations become public. In this section, we analyze two situations where catastrophic security incidents occurred and discuss how an effective security metrics program might have alleviated or even eliminated suffering and loss.

Barings Bank: Insider Breach

Let us first look at a breach with the most dire of consequences: bankruptcy. The breach was actually a succession of breaches perpetrated by one individual, Nick Leeson, over a period of four years that resulted in the collapse of Barings Bank and its ultimate sale to the ING Group for one pound sterling in 1995.

The players

Barings Bank was Britain's oldest merchant bank, founded in 1762. It had a long and distinguished history, helping to finance the Louisiana Purchase, the Napoleonic Wars, the Canadian Pacific Railway, and the Erie Canal. The British government used Barings to liquidate

Copyrighted material

assets in the United States and elsewhere to finance the war effort during World War II. Princess Diana was the great-granddaughter of one of the Barings family. Barings was Queen Elizabeth's personal bank.

Born in 1967 as a working class son of a plasterer, Nick Leeson's life is a tale of rags to riches to rags, and possibly back to somewhat lesser riches. He failed his final math exam and left school with few qualifications. Despite this, he managed to land a job as a bank clerk, which led to a succession of additional jobs with other banks, ending up with Barings in the early 1990s. At Barings and earlier, Leeson worked in back-office operations, but shortly after joining Barings, he applied for and received a transfer to the Far East to help straighten out a back-office problem in Jakarta. His work there was visible and successful. In 1992, he was appointed to the position of general manager of Barings Securities (Singapore) Limited (BSS) with the authority to hire both traders and back-office staff to start up a futures trading operation on the SIMEX (today's Singapore Exchange). Although it was not his job to trade but, rather, to manage, he quickly took the necessary exam, passed it, and began trading on SIMEX along with his team of traders. Consequently, at the time of the breaches, he held three roles: general manager, trader, and de facto head of back-office operations (due to his extensive past experience). Separation of duties is a fundamental tenet of any trading operation, yet it was absent at the BSS operating unit of Barings for four years, only to be discovered too late.

How it happened

Leeson and his team of traders had the authority to perform two types of trades: futures and option orders for external clients and for other Barings business units, and price arbitrage to leverage price differences between futures traded on SIMEX and Japan's Osaka Securities Exchange (OSE).

Officially, Barings (and Leeson as its designated representative) adopted a trading strategy called a "short straddle." This involves selling (going short) both a call option and a put option on some stock, interest rate, index, or other underlying instrument—in Leeson's case, the Nikkei 255 index (among others). For this discussion let's assume that some number of shares of a publicly traded stock is the underlying instrument.

If you are not familiar with options trading, a few definitions are in order. There are two basic types of options: *puts* and *calls*. Both puts and calls are contracts between two parties—a seller and a buyer.

Put options provide a right to sell. A buyer of a put has the option, at her sole discretion, to sell a certain number of shares of stock (N) for a strike price (S) at a time T in the future. The seller of a put option is making a commitment to buy a certain number of shares of stock (N) for a strike price (S) at a time (T) in the future. For one put option, the values of N, S, and T are the same for both the buyer and seller of the put option. The price of the option ($Qput$) is what the buyer pays to the seller for the contract (i.e., the option).

Copyrighted material

Calls are essentially a mirror of puts. Call options provide a right to buy. The buyer of a call has the option, at his sole discretion, to purchase a certain number of shares of stock for a strike price at a time in the future. The seller of a call option is making a commitment to sell a certain number of shares of stock for a strike price at a time in the future. As with put options, the parameters *N*, *S*, and *T* all apply and are the same for one contract between the buyer and seller of a call. The price of the option (*Qcall*) is what the buyer pays to the seller.

A *short straddle* is a trading strategy in which a trader—say, Nick Leeson—**sells** matching put and call options. Note that a *long straddle* is a strategy in which the trader **buys** matching put and call options. For both of these trading strategies, one can build a model that projects trading profits (*Y*) as a function of *N*, *S*, *T*, and the current stock price per share (*X*). Since Nick was authorized to follow a short straddle strategy, let's look at short straddles in detail.

On day one, Nick sells two options: a put and a call for *N* shares, strike price *S*, and expiration date *T*. Nick receives:

Q = Qcall + Qput

from the buyers of his two options. So Nick is starting out with a profit equal to *Q*. As you will see from our analysis, this is the best Nick can do. At time day 1 plus *T*, when the two options expire, there are three possible cases:

Case 1: X = S (the current stock price X is equal to the strike price S at time T)

If the current price and strike price are the same, then neither option will be exercised and Nick will not have to buy or sell any stock. He keeps his profit *Q* and enjoys the fleeting taste of victory.

Case 2: S < X (the current stock price X is greater than the strike price S at time T)

In this case, the holder of the call option, who has a right to buy at *S* (a lower price), will do so. The holder of the put option, with the right to sell at *S*, will not exercise this option because she can sell at the higher current price. Thus, Nick has to purchase *N* shares at *X* and sell *N* shares at *S*, yielding an overall profit equal to:

Q – N * (X – S)

This is OK as long as *N* * (*X* – *S*) is less than *Q*. It is easy to imagine a circumstance where this would *not* be the case. Large *N* or large (*X* – *S*) will put Nick in the hole.

Case 3: X < S (the current stock price X is less than the strike price S at time T)

In this case, the holder of the put option, who has a right to sell at *S* (a higher price), will do so. The holder of the call option, with the right to buy at *S*, will not exercise this option, because he can buy at a lower current price. Thus, Nick has to purchase *N* shares at *S* and sell *N* shares at *X*, yielding an overall profit equal to:

Q – N * (S – X)

This is OK as long as *N* * (*S* – *X*) is less than *Q*. It is easy to imagine a circumstance where this would *not* be the case. Large *N* or large (*S* – *X*) will put Nick in the hole.

Copyrighted material

We can compute the break-even value for the current stock price X at time T by solving the equations shown earlier with the profit set to zero:

$0 = Q - N * (S - X)$ where $S > X$
$0 = Q - N * (X - S)$ where $X > S$

This yields:

$X = S - Q/N$ for $X < S$
$X = S + Q/N$ for $X > S$

To summarize, the model for short straddles can be expressed as a graph as shown in Figure 3-1. This graph shows how profit on the y-axis varies in response to changes in the current stock price as reflected on the x-axis.

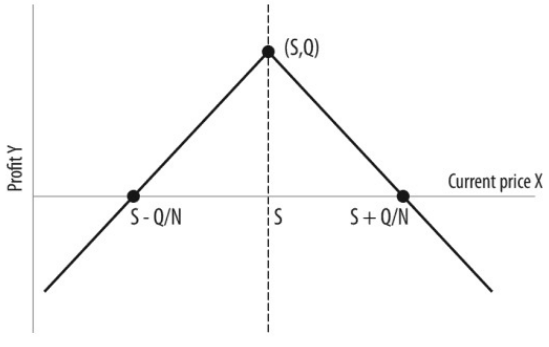

FIGURE 3-1. Short straddle profit model: profit Y is expressed as a function of current stock price X at option expiration

This is not a pretty picture at all. To be sure, it is a strategy that will yield (limited) profits in a nonvolatile market but potentially huge losses if the market becomes volatile. As Figure 3-1 shows, for large values of N or for large differences between X and S (high price volatility), the profits become increasingly negative.

Almost immediately after beginning to trade, Leeson began taking unauthorized and speculative positions in Nikkei 255 futures, Nikkei stock options, and Japanese Government Bonds. Using his influence over the back-office staff, he reported his profits from the authorized straddles in the proper accounts while hiding his unauthorized activities and losses in an old error account (Account #88888) that was an artifact of his previous back-office project in Jakarta. Table 3-1† shows some financial metrics that characterize the situation. All values are in millions of Great Britain Pounds (GBP).

† Source: Report of the Board of Banking Supervision Inquiry into the Circumstances of the Collapse of Barings, Ordered by the House of Commons, Her Majesty's Stationery Office, 1995.

Copyrighted material

TABLE 3-1. Barings Bank: some financial metrics

Year	Reported	Actual	Cumulative
End 1993	+8.83 M	−21 M	−23 M
End 1994	+28.529 M	−185 M	−208 M
End 1995	+18.567 M	−619 M	−827 M

Up until January of 1995, Leeson was primarily unskilled and unscrupulous. But on January 17, 1995, he became monumentally unlucky as well. A major earthquake in Kobe, Japan caused the Nikkei 225 average to plunge. Extreme volatility continued for over a month. Whereas a prudent trader would have taken his losses, Leeson did the opposite: he started an aggressive buying campaign of futures in the Nikkei 225 index, as shown in Figure 3-2.‡

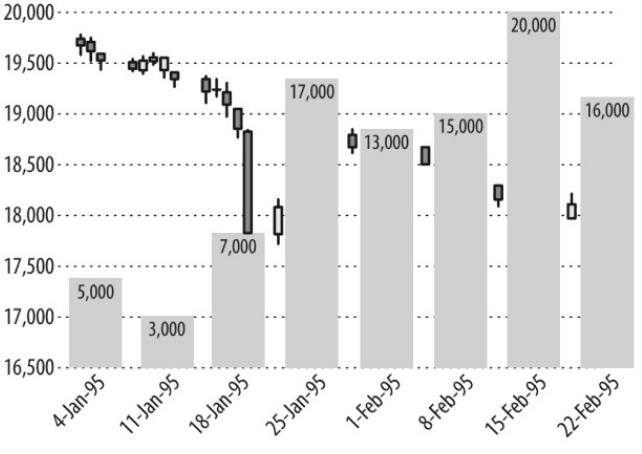

FIGURE 3-2. Purchases by Leeson during crisis

The x-axis reflects time, beginning just before the Kobe earthquake on January 17, 1995, and continuing until just before Barings went into receivership on February 27, 1995. The lefthand y-axis and the bars with whiskers show the Nikkei 225 high, low, and closing averages on the OSE. Before the earthquake, the Nikkei 225 index was trading at a price between 19,000 and 19,500. After the Kobe earthquake, that price plummeted and then wildly swung between as low as 17,400 and as high as 18,800. This translates to high differences between X and S in the short straddle model.

The gray bars in Figure 3-2 are plotted on a scale of Millions of U.S. Dollars ($M) that represent the long futures positions of Barings' traded by Leeson. This scale ranges from a low of about

‡ See *http://www.asiafunclub.com/japan/finance/stock_market/nikkei_1995.shtml* and Zang, Peter G., "Barings bankruptcy and financial derivatives," 1995, *World Scientific*, p. 137.

Copyrighted material

$3B to a high of more than $20B, as opposed to the scale represented in the left y-axis that represents the Nikkei index, ranging from a low of 16,500 to 20,000. Numbers are provided for each bar to indicate the associated value in $M. So, the height of each bar reflects Leeson's commitment to purchase Nikkei index shares at a future date for a specific price. Leeson's maximum commitment during this period was $20 billion—more than 32 times Baring's then-reported capital of approximately $615 million. This translates to high values for N in the short straddle model.

The end was swift, painful, and very public.

To be sure, the numbers and charts in this section reflect financial metrics, not security metrics. A key characteristic of beautiful security metrics is that they are linked somehow to financial metrics, or at least some meaningful measure of the performance of a business. What are some security metrics that might have provided the red flags needed to catch the problem before the situation became unsalvageable? We first analyze what when wrong and then look at a "what-if" scenario that assumes Barings had a metrics program.

What went wrong

In reviewing the context and sequence of events described, we can make the following observations:

- Leeson did not have a clearly defined role with targeted system privileges. In fact, he simultaneously was wearing three hats: trader, back-office manager, and general manager.
- Separation of duties between those who define, enforce, and follow policies was absent.
- There was no formal process for reviewing Leeson's access and entitlements for performing back-office and trading functions—specifically gaining access to IT applications.
- No controls were in place to ensure that reviews were performed when Leeson's responsibilities and functions changed.

An important role to which security metrics are well suited is monitoring and performance measurement for controls. In the Barings case, many critical controls were absent. In the next section, we leverage some inspiration from medical research to show how security metrics can be used for both detecting an absence of critical controls and monitoring ongoing effectiveness and efficiency.

Barings: "What if..."

As discussed earlier, medical research has defined a collection of facts that a physician can collect to evaluate a person that she is meeting for the first time and of whom she has no prior knowledge. The first step is to take a history—we have all done that many times. This history is a series of simple questions that provide an overview of patient status. Turning to companies in general and Barings in particular, what would be an appropriate analogy to taking a medical

Copyrighted material

history? The answer is a "security history." This is a series of high-level questions that provide a general characterization of the organization's implementation of controls that protect the security of IT assets. Here are some sample questions:

- Does the organization have formally documented information security policies and procedures?
- Are employees required to follow these policies and procedures?
- Does the organization have an identity and access management process that governs the management of user access and entitlements?
- Is a separation of duties enforced between individuals that authorize access and those that enable such access to IT assets?
- Is a formal review process conducted whenever an employee's job or responsibilities change?
- Is system and application activity instrumented via mechanisms such as logfiles?
- Are activity logs regularly analyzed?

The number of questions in a security history may number in the hundreds. Unlike a patient history, it may take days or even weeks for an organization to complete it, but the principle is the same. It highlights the top-level issues and is intended to give the big picture as well as provide a road map for further investigation and ultimately the collection of more targeted metrics.

The IT security market, as of 2008, is starting to develop guidelines for security histories. Unlike the medical profession, where recommended medical history questions are available, there is little organization or structure in the IT security space. But this is changing. Some organizations are emerging to provide requirements:

- The Payment Card Industry (PCI) has issued a set of requirements required from its members to trust that a company will appropriately protect customer credit card data.
- The Santa Fe Group in partnership with BITS has created the Financial Institution Shared Assessments Program to boost the effectiveness and efficiency of assessing a company's IT security posture.
- Companies such as the Unified Compliance Framework (*http://www.unifiedcompliance.com*) are developing specifications that harmonize requirements as adumbrated by regulations, standards, best practices, and more.

These efforts have high potential for generating a level of consensus "security history" similar to that which the medical profession currently enjoys for medical history.

Barings' answers to these sample (admittedly targeted) questions, had anyone asked, would have been telling. However, it is reasonable to conjecture that the Barings senior management in the U.K. did not realize how bad things were at BSS operations in the Far East. In fact, had they asked, it is certainly possible that Leeson would have lied in filling out his answers to a

Copyrighted material

questionnaire. A subjective questionnaire that reflects what people *say or think* needs to be backed up with *hard facts and data*. Thus, like a medical history, this type of security history must be augmented with metrics. The history is merely a quick and somewhat subjective precursor that can point to areas where more in-depth investigation should begin.

Consider the following what-if scenario: What if a present-day Barings had decided to obtain a Shared Assessments security rating? Or what if a present-day Barings had undergone a mandatory PCI audit as required for maintaining a good-standing status with payment card companies such as Visa and American Express?

Right away, Barings' lax controls for access to trading and back-office IT systems would likely have been determined. Auditors might have discovered a lack of centralized controls. They might have discovered that there were no regular reviews of employee entitlements—especially when employees changed jobs. In light of these findings, Barings would be faced with a low Shared Assessments rating, which in turn would impact their reputation, or they would potentially lose their ability to process payment card data, thereby losing significant revenue.

These last two business impacts would likely have gotten the attention of senior Barings management. Perhaps this would have prompted them to initiate a project to treat the problem. But how much should they commit to spend, and what should they adopt as realistic objectives or measures of success? It is easy to imagine them asking:

- "How did our competitors do?"
- "How sophisticated are their identity and access management controls?"

Unlike health care, the IT security profession has no help to offer here. Individuals whose medical data is used for research have similar risks that can materialize in several forms—higher insurance costs or no insurance coverage, difficulty finding employment, credit problems, and possibly litigation if they infect others. But individuals do, despite these risks, share their data. Why? One reason may be a general confidence that their data is appropriately anonymized prior to its release to researchers. Addressing this concern in relation to corporate security, the Center for Internet Security (CIS)[§] has announced a cross-enterprise benchmark metrics project. Associated with this project, my company PlexLogic is launching a service, called MetricsCenter, that can anonymously collect the data needed to compute the CIS benchmark metrics. The MetricsCenter technology leverages research in anonymization, much of which was motivated by the privacy requirements for medical research.[‖]

Of course, like Barings in the early 1990s, many companies may not be collecting metrics, so there are none to report. This is like an individual choosing never to visit a physician: not prudent.

[§] See *http://www.cisecurity.org* for more information.

[‖] See *http://www.plexlogic.com* and *http://www.metricscenter.org* for more information.

Copyrighted material

As a result, the Barings executives cannot use metrics (others or their own) to define the scale and scope of their "treatment." Since they have no choice (they need and want a good security rating), Barings must launch an internal project to "treat" their "sick" identity and access management controls based upon intuition. Let's assume that, this time, they decide to implement a metrics program along with the treatment program.

The good news is that Barings' project can benefit greatly from metrics, and their implementation is entirely within the control of internal captive resources. In particular, there is no reliance upon others to collect and share data.

Barings: Some security metrics

So, in our what-if scenario, let's assume that Barings is well along in their project to institute improved controls for identity and access management. Let's further assume that they have embraced security metrics and are busy collecting, analyzing, and reporting results. Note that while the access management improvements are slowly advancing toward the BSS business unit in Singapore, it is certainly possible for Leeson to be blissfully trading away. But his days are numbered. Let's take a look at some generic metrics that will have a high likelihood of exposing his transgressions:

Coverage metrics

These characterize how far a solution has been implemented, as well as inconsistencies in deployment. Here are some examples, in the context of the Barings scenario:

- Percentage of applications or basic services—such as email, websites, or shared storage—whose access is managed by an authoritative system. The closer this value is to 100%, the more complete control has been concentrated in one or a few well-managed places, thereby facilitating accountability. To calculate this metric, one would compare two lists: one from the authoritative systems and one compiled from configuration files and activity logs for the target application or service. The percentage of overlap of these two lists is what this metric measures. In Leeson's case, the metric would expose the existence of secret accounts that the applications recognized for the purpose of granting access but were not officially managed by an authoritative system.
- Percentage of login accounts that are explicitly linked to a valid employee or contractor. Any deviation of this metric from 100% should be examined carefully and ultimately justified. A large class of accounts that are not linkable to current employees or contractors would be terminated employees—some of whom may have less than positive attitudes toward their former employer and would almost certainly have "inside" information. Again, this metric is calculated by comparing two lists: the first from an authoritative access control system and the second from the corporation's human resources or employee roster. The amount of overlap of these two lists is what this metric measures. It would expose that Leeson was using a former employee's account or a bogus account to gain unauthorized access.

Copyrighted material

- Percentage of accounts or of employees whose access and entitlements are formally reviewed at least once per year, reviewed once per quarter, or have never been reviewed. The percentage of accounts or employees never reviewed should be as close to zero as possible. This metric is often derived from service ticketing systems that keep records of reviews conducted. The result is calculated by dividing the number of conducted reviews by the total number of active accounts. Leeson and his accounts likely were never reviewed, with the worst possible consequences. Leeson was promoted after his first project in Singapore, but his access credentials were never reviewed. As with the other metrics, his existence as an unreviewed employee would prevent this metric from reaching its target.
- Percentage of group overlap. In access control systems, it is common practice to define groups of accounts that share the same entitlements. For example, a group called Options Traders could be defined to possess all of the entitlements needed to grant access to a set of trading applications, or another group called Trade Administration could be defined to possess all of the entitlements needed to account for the exchange of money between buyers, sellers, exchanges, and other parties. The percentage of group overlap between the Options Traders and the Trade Administration group should ideally be zero. It may be less of a red flag if all of the accounts in that intersection have recently been reviewed (e.g., within one quarter). In Leeson's case, his account would have been in that group intersection with no record of any review for over 12 quarters.

Like measuring temperature or pulse rate, none of these metrics are difficult to understand, and none require advanced mathematical computations. Also, they are probably not definitive. It is, after all, unrealistic for any company to achieve 100% perfection in these metrics, just as a person with less than perfect temperature or pulse rate may have a good reason for the deviation (e.g., they just ran up 10 flights of stairs). Also, these metrics are values derived at a single point in time, which gives only a snapshot of an organization's status. When regularly and repeatedly computed, the time series of measurements can form a valuable data set that will support further, potentially more sophisticated analysis.

Time-series metrics are based on a set of values obtained at regular or irregular time intervals. Dynamic changes in a system over time, as reflected by a time series, show whether system state is improving or degrading—and how rapidly. As examples, consider what we would see if we collected all of the previously described metrics monthly or quarterly. Metrics derived from one business unit within a single organization such as Barings could potentially be used to forecast how rapidly other business units could reasonably be expected to roll out new controls. Perhaps we could even project how long it would take to discover and end Leeson's unauthorized access to back-office functions.

Investment metrics

Similarly, if one were tracking the costs associated with a program to improve access controls, one could derive a cost per unit of improvement for any or all of the previous

Copyrighted material

metrics. If the rate of spending is constant and the rate with which a metric approaches its target (e.g., 100%) is slowing, the cost per unit of improvement is increasing. Using forecasting models, one can project how much it will cost to achieve alternate control targets (say, 95%, 96%, etc.). Of course, in Leeson's case, it would be (with the advantage of hindsight) easy to justify any expenditure to get BSS's identity and access management systems under control.

Treatment effect metrics

To derive treatment effect metrics, run a time-series metric to compare performance before and after a "treatment" is applied. Using basic statistical analysis, one can test the null hypothesis, which in this case would say that any detected change is due merely to chance and that the "treatment" made no difference.

As an example, suppose that before initiating the access control project for a designated business unit, Barings decided to collect some metrics about access-related security incidents. Metrics such as the Mean Time Between Security Incidents (MTBSI) or Mean Time to Repair Security Incidents (MTTRSI) would characterize the frequency and impact of observed access security incidents. These baseline values (the "before" time series) could later be compared with the same measures taken after access control was deployed (the "after" time series). Was there improvement? Was it significant (not just by chance)? A statistically significant result would give you confidence that the improvement can be attributed to the deployment of access control.

With "before" and "after" time series data, Barings would have hard facts with which to assess the effectiveness of its treatment. Additionally, it would have hard data to present to auditors and thus either pass an initial audit or clear a negative finding. As more results are gathered in the fullness of time, the statistically derived levels of confidence and quantification of the treatment effects will improve. Finally (this can be only a conjecture in the absence of data), they might have been able to avoid the big access control incident that ended it all.

It is well and good to collect measurements, compute metrics, and analyze results, but these activities are of little value unless one also communicates the findings to the right audience at the right time for driving better decisions. For Barings, it is safe to say that senior executives would not be likely to read detailed reports about access control incidents or, for that matter, any of the detailed technical metrics that we just described. What they might look at is a dashboard with a list of general areas and associated color codes: say, red (bad), yellow (caution), or green (OK). The managers in charge of each area would assign the color codes based upon their interpretation of metrics such as those I just described.

The managers have a tightrope to walk when they are responsible for tracking and reporting security vulnerabilities. Reporting a problem simultaneously brings both the potential benefit of evoking the funding to resolve the problem and the reputational damage for allowing it to happen in the first place. This reputational downside is actually similar to some (but not all) health care situations. Metrics bypass the dilemma by forcing managers to share information

Copyrighted material

with their bosses, who can then evaluate their performance in isolation or as compared with their peers. It's ironic that these same bosses often refuse to share information outside of their department or outside of their company because they fear being evaluated against their peers as members of the same market ecosystem.

Certainly, Nick Leeson's account and its contribution to metrics such as those described would have merited at least a yellow rating for access management controls in the BSS business unit or in the Far East location. The presence of a persistent yellow rating, with no improvement, would hopefully have precipitated an executive decision to commission some type of investigation, if for no other reason than to be able to annotate the yellow with statements that the accounts had been reviewed and no problems were discovered.

It is a safe bet that any sort of review would have curtailed or ended Leeson's unauthorized activities. And, again, it is interesting to note that no understanding of complex derivatives, arbitrage, or higher mathematics would have been necessary.

TJX: Outsider Breach

The second example for security metrics in this chapter looks at the biggest case of payment card theft ever recorded (as of 2008). This is a very recent breach. In fact, events and information about it are still coming to light as this chapter is being written in October 2008. Because of the currency of this case, it is often difficult to assert what is proven and what is yet to be proven. Some of the following narrative is based upon unverified allegations from law enforcement and others close to the case. However, this does not diminish the value of the lessons I draw, because I am using the case to illustrate the value of security metrics.

The breach (or really, breaches) is believed to have started in July 2005 in a parking lot outside a Marshalls discount clothing store near St. Paul, Minnesota. Breaches continued for 18 months until December 2006, when TJX took action. TJX was actually not the first to detect the massive egress of credit card and other customer personal data from their network. Rather, it was credit card holders and issuers who began to see strange transactions on their credit card bills as early as November 2005. It wasn't until December 2006 that TJX became suspicious enough to hire forensics experts and launch an investigation. Although TJX has survived the breach, it may cost the company as much as $1 billion over five years to cover all of their expenses, such as consultants, IT security upgrades, legal fees, and potential liability to payment card companies and banks. Costs to affected individuals and partners doing business with TJX will take a long time to determine.

The players

TJX Companies, Inc., with 2007 revenues of over $17 billion, is the largest off-price apparel and home fashions retail store chain. The company is parent to eight businesses, 2,500 stores, and 130,000 employees. TJX businesses in the U.S. currently consist of T.J. Maxx, Marshalls, HomeGoods, and A.J. Wright.

Copyrighted material

As of summer 2008, 11 men have been charged with not only the T.J. Maxx and Marshalls thefts, but also hitting an additional eight retailers in the U.S.: Office Max, Barnes & Noble, Sports Authority, Boston Market, DSW Shoe Warehouse, Dave & Buster's, Forever 21, and B.J.'s Wholesale Club. Ten are in custody; one is at large. The accused ringleader is Albert Gonzalez, age 27, of Miami, Florida. The others come from Estonia, the Ukraine, China, and Belarus, as well as the U.S.

At the time of his arrest, Gonzalez had in his possession a Glock 27 firearm, a 2006 BMW, a condominium in Florida, and $1.65 million in cash. He had a high school education, was single, and lived with his parents. In 2003, Gonzalez was arrested on fraud charges in New Jersey and was subsequently recruited by the Secret Service to help an investigation into a major computer hacking operation. In 2004, he was credited with helping to take down a message-board operation that was used as a clearinghouse to buy and sell credit card numbers and other personal information. He was subsequently assigned to the TJX case. His assignment was to be an informant and pass information learned about the hackers to the Secret Service. However, the Secret Service now alleges that he was a double agent, passing information he learned from the Secret Service investigators to the hackers. Gonzalez's Secret Service consulting position allegedly paid him $6,000 per month.

How it happened

Let us first look at how events unfolded from the public perspective. The following time line was published by the *Boston Globe* on October 25, 2007:[#]

> **2007 Jan. 17:** TJX Cos. says it suffered an unauthorized intrusion into its computer systems, potentially compromising customer credit and debit card data as far back as 2003. The company said it knows of no misuse of data.
>
> **2007 Jan. 18:** Credit card companies, banks, and customers begin to report fraudulent use of credit and debit card numbers that had been stored in the TJX system. Thousands of cards are canceled.
>
> **2007 Feb. 21:** TJX reports that hackers may have gained access to its computers in 2005, a year earlier than it previously thought.
>
> **2007 March 28:** TJX puts the number of credit and debit card numbers stolen by hackers at 45.7 million, but says about 75 percent of those were expired or had their data masked.
>
> **2007 July:** Florida police obtain guilty pleas from several individuals in Florida for using credit card numbers taken from TJX to purchase $8 million in gift cards and electronics at Wal-Mart and Sam's Club.
>
> **2007 Aug. 14:** In a corporate filing, TJX puts the total cost of the data breach at $256 million.

[#] See *http://www.boston.com/business/globe/articles/2007/10/25/tjx_timeline*.

Copyrighted material

2007 Sept. 21: TJX reaches a tentative settlement with customers affected by the breach, offering store vouchers to those affected and planning to hold a three-day "customer appreciation" sale. The settlement is later amended to offer affected customers the choice of vouchers or cash.

2007 Sept. 25: Canadian privacy officials fault TJX for failing to adopt adequate safeguards to protect customer information.

2007 Oct. 23: Court filings in a case brought by banks against TJX say the number of accounts affected by the thefts topped 94 million.

This time line shows the progression of escalating revelations regarding the impact of the breach, both in terms of numbers affected and ensuing litigation (all interesting metrics in their own right). As of October 2007, the criminals had not been caught, but there was an active investigation involving multiple law enforcement organizations in multiple states and countries. Almost a year later, in August 2008, arrests and indictments were announced.

Here is a time line based upon a technology perspective:

1999: Wireless LANs born: IEEE 802.11b completed and approved.

2001: Hack to crack WEP protection of wireless communication demonstrated.

2002 April: IEEE 802.1X ratified to address port-level security using Extensible Authentication Protocol (EAP), which works with RADIUS servers, already in wide commercial use. It was intended to standardize security for wired network ports, but was found to be applicable to wireless networking as well.

2002 July: Widespread publication of IEEE 802.11b security weaknesses with detailed recommendations to obtain enterprise-class strength.

2004 June: IEEE 802.11i ratified as an amendment to the wireless IEEE 802.11 standard to provide a standard specification for security mechanisms and to supersede the previous security specification.

2005 June: TJX use of WEP protection still prevalent at retail stores.

It is now alleged by law enforcement that Gonzalez and others practiced a popular and low-budget technique called *wardriving* to uncover open wireless networks. Wardriving involves cruising public areas (in this case, the parking lots of malls and retail stores) in cars equipped with PCs and a good antenna. The antenna discovers wireless networks. The PC runs wireless client software that attempts a connection if the network has open access, or records transmitted packet streams if the network has implemented some form of access security. If the access security is weak, the packet streams can be analyzed using readily available software to extract all the information needed to break in. Once in, the hackers can access network-connected customer databases and begin collecting credit card numbers and other personal information.

Copyrighted material

To understand how this works in a bit more detail, let us look at a few key facts about wireless networks. We will also provide Consensus Good Practices, recommendations that have been developed by various industry groups and standards organizations:

- The IEEE 802.11b standard specifies a Service Set Identifier (SSID) that differentiates wireless access points (independent networks). As an example, a hotel called the Good Night's Rest might name its WAP GoodNR for convenience. WAPs operating in default mode broadcast the SSID every few seconds in what are known as "Beacon Frames." While this makes it easy for authorized users to discover and connect to a network, it also makes the same thing easy for unauthorized users.

  CONSENSUS GOOD PRACTICE: A company should set the value of all SSIDs to something obscure. A setting such as "TJX-Miami" is not a good choice. Additionally, it is best to turn off SSID broadcasts.

- To authenticate a user who wishes to connect to a network, IEEE 802b specifies a protocol called Wireless Equivalent Privacy (WEP). Most WAPs ship with a default setting for open authentication, which means that WEP is disabled and there is no authentication for clients who wish to connect.

  CONSENSUS GOOD PRACTICE: A company should change the default setting for authentication to disallow open access for all clients.

- When WEP is enabled, each individual client is configured with a secret key that is shared with its associated WAP. A prospective network client begins by sending an association request to the WAP, whereupon the WAP replies by sending a string of challenge text to the client. The client then applies an algorithm called RC4 to encrypt the challenge text and then sends it back to the WAP. The WAP authenticates that the client of the challenge text was encrypted correctly.

  CONSENSUS GOOD PRACTICE: Key management for WEP is a nightmare. To change the shared key, the configurations for each and every WAP and client must be updated.

- The WEP protocol allows an unauthorized listener to capture two of the three variables in the authentication equation: the challenge text and the RC4-encrypted response. With enough sample pairs, a hacker can discover the shared key using well-known mathematics embodied in freely downloadable software.

  CONSENSUS GOOD PRACTICE: WEP is not adequate protection for a wireless network. Good practices for achieving strong wireless security include the following:

  — Turn off WEP

  — Select obscure SSIDs

  — Isolate wireless subnets with routers and firewalls

  — Use 802.1X for key management

Apparently TJX, as well as other breached retailers, did not follow guidelines issued as early as July 2002. The hackers leveraged this fact to gain access to the company network. As a result,

Copyrighted material

they could acquire customer information in transit within a store's network, as well as wirelessly transmitted employee credentials. Using the employee credentials, the hackers gained access to both in-store servers and servers at TJX headquarters in Massachusetts, which stored repositories of customer data.

What went wrong

Unlike the Barings Bank failure, which was an internal breach attributable to an egregious lapse in access management processes, the TJX failure was a breach perpetrated by outside attackers that succeeded due to lapses in network and system management processes. People close to the ongoing investigation allege that the outside attackers found WAPs with default configurations, WAPs with self-evident SSIDs and open access, and WAPs configured to use WEP. These WAPs were the low-hanging fruit that let the hackers gather enormous quantities of valuable data with an incredibly small investment.

So, given our perfect 20/20 hindsight, what did TJX do wrong? The technical lapses are easy to list:

- Inadequate network configuration control
- Inadequate network usage surveillance
- Inadequate server configuration and usage monitoring
- Inadequate data protection, particularly data egress

At a deeper level, of course, the technical failures were symptoms of mismanagement:

- Ignorance of or indifference to basic security practices, such as password protection
- Ignorance about wireless network operation, compounded by failure to follow industry alerts and news about common vulnerabilities
- Lack of unified policies and lack of communication with remote sites about recommended policies
- Refusal to pursue evidence of trouble when reported
- Lack of concern in general for security at all times: during installation, during staff training, and during ongoing use when monitoring should have been taking place

The wireless standards committees and manufacturers also have to shoulder some of the blame for the problem, but that is a wider discussion that I can't attempt in this article.

> **NOTE**
>
> **It's not just TJX that is guilty. Nine other retail companies were hit by the same thieves using the same hacking techniques and exploiting the same holes in their IT systems' security.**

Copyrighted material

TJX: "What if..."

What if TJX had sophisticated sensors that could continually monitor the millions of transactions and the (likely) terabytes of data both stored and in transit within their IT infrastructure? What if they could regularly summarize the telemetry (no human can read it all) and embody it in a collection of metrics, covering a spectrum of aggregation criteria, for regular review? Medical research has produced sophisticated sensors that yield vast quantities of data that can be made available to both researchers and practitioners. It is certainly possible.

The key to successfully creating "smart sensors" in medicine (along with data sharing) is that medical practitioners who deal directly with patients work closely with researchers who, in turn, work closely with vendors that manufacture sensors. As a result, when a medical diagnostic tool is released, it not only can sense data but can also summarize it and provide very specific guidance as to how to interpret results.

As an example, consider equipment that performs bone density or DEXA scans for osteopenia. First, the manufacturer provides information about what the equipment does, how to operate it, when it should be used, and how to interpret results. In addition, medical researchers provide information about risk factors for osteopenia, various treatments (e.g., diet, medication), current research (including medical trials), support groups, references, and additional products. DEXA scan results are often presented as a color-coded picture of a patient's skeleton, where color represents bone density. It is a complete package.

With the current state of security products, one gets software and a set of manuals that are all about the technology and how to operate it. Papers covering recommended usage and the significance of events, which would be the main vehicle for delivering a diagnosis and interpretation, are, for many security vendors (and their investors), the least interesting part of the product. Indeed, system administrators in the field also have a poor record of reading and understanding such documents. The patented gizmo that lies deep within a sensor or software platform, and which end users will never understand, seems to be what is valued most highly—especially in startups, where a lot of the innovation resides. Security vendors, in general, lack the domain expertise to deliver a complete solution, thereby leaving a gap between what their products deliver and what end users can effectively use.

Security metrics have the potential to bridge some of this gap. By providing definitions that specify both sources of raw (sensor) data as well as how to interpret results, metrics can significantly add to the value of the existing wealth of security management products. The Security Metrics Catalog project, jointly sponsored by SecurityMetrics.org and MetricsCenter.org, is designed to provide a centralized, searchable repository of metrics definitions.*

Let's look at a few security metrics that TJX would have found useful.

* See *http://www.securitymetrics.org* and *http://www.metricscenter.org* for more information.

Copyrighted material

TJX: Some security metrics

The problem of data breaches, unlike other computer security issues, gives us the unique opportunity to look at some public metrics, on both global and local levels. Global metrics capture industry-wide experience, whereas local metrics capture the experience of a single enterprise.

Global metrics. Global metrics need global data to drive them. California's SB 1386 mandate and similar legislation require the reporting, starting in July 2003, of public breaches that affect individual personally identifiable information. The Open Software Foundation, a group of volunteers, tracks publicly reported security incidents, using press accounts as their primary source of information. Their results are posted at the Attrition website† in the form of the Data Loss Data Base (DataLossDB) repository, which physically is a comma-delimited file that is easily ingested by most statistics software. As of July 15, 2008, the Open Security Foundation has assumed hosting responsibilities from Attrition.org for the DataLossDB. Maintenance of the DataLossDB will be a community-driven project at a new location, *http://datalossdb.org*.

Using the DataLossDB as a source, one can derive a simple metric, "Number of Events per Quarter," and generate the time-series plot shown in Figure 3-3.

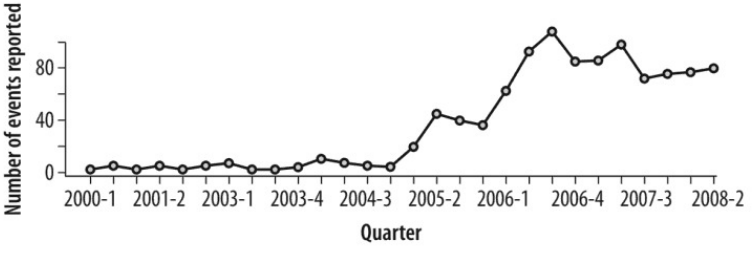

FIGURE 3-3. Incidence of data breaches over first years of mandated reports

The scope and scale of the mandate to report breaches has grown from just California in mid-2003 to an increasing number of states over the past five years. Clearly, the trend in the number of breaches reported is up, but it does seem to be flattening out. No doubt some of the early growth during 2005 can be attributed to new mandates in more jurisdictions. People who otherwise might not have disclosed an event are now required to do so by law. In Figure 3-4 we'll "zoom in" to view monthly data, starting in 2005.

The monthly data shows a fair amount of volatility. If we fit a line (using a least-squares model) to the observations, we can see that the trend is increasing slightly. One can use such a line to make projections into the future, but precision is not the goal. The metrics diagrammed (namely the slope and intercept of the dotted line) paint a clear picture that incidents are

† See *http://datalossdb.com* and the DataLossDB database.

Copyrighted material

increasing in frequency and therefore it is fair to say that the situation is getting worse, not better.

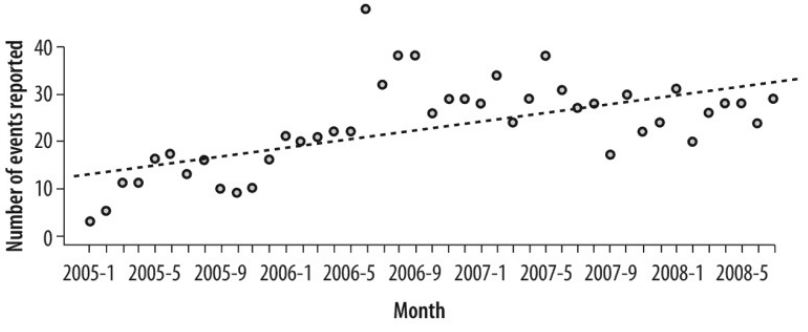

FIGURE 3-4. Incidence of data breaches over recent years

The TJX breach was perpetrated by outsiders. It is widely believed that insider breaches are increasing faster than outsider breaches. The DataLossDB data allows us to investigate this question with hard facts and data. Since DataLossDB tags each breach with an "insider" or "outsider" label, we can create a new plot, similar to the previous one but breaking event frequency down into these two subsets. This calculation produces the graph in Figure 3-5, again starting in 2005.

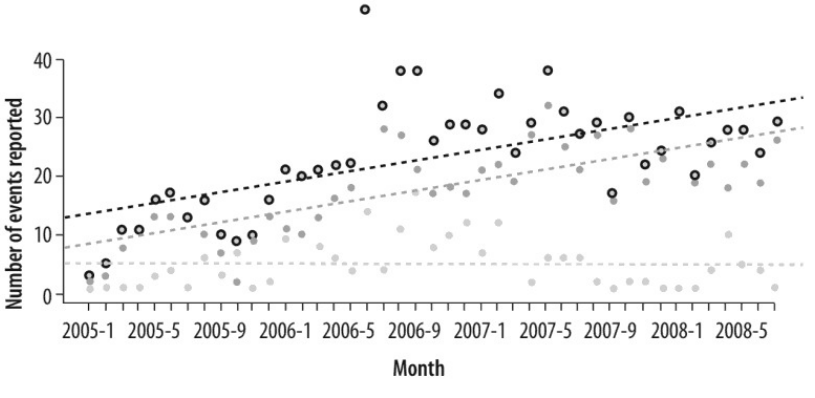

FIGURE 3-5. Breakdown of data breaches

The solid dark gray dots represent the number of insider breaches, and the solid light gray dots represent the number of outsider breaches. The black donuts just replicate the total of the two, as plotted in the earlier trend graph. The dark gray, light gray, and black dotted lines are least-squares fits for the outside, inside, and total frequencies, respectively. It is easy to see that

Copyrighted material

outside breaches are increasing, whereas the incidence of inside breaches are essentially flat, so the increase in total breach frequency is due entirely to the increase in outside breach frequency—at least according to the data in the DataLossDB.

Another aspect of a breach is impact. How far-reaching is the amount of data compromised by each incident? It turns out that the Attrition volunteers record data about each breach that provides a good measure of impact: the total number of individuals affected by the breach. The name for this value is *TotalAffected*. Of course, for most events this is just the best estimate that the breached company can provide, but it is much better than nothing. Again, we are not necessarily pursuing precision here. So Figure 3-6 shows a box plot that reflects the several "distribution" metrics for *TotalAffected* on the $\log_{10}$ y-axis, scaled across all breaches for each year starting in 2005 and going through August 11, 2008. In truth, since 2008 is not complete at the time of this writing, one could debate whether we should include 2008 in the graph. Our rationale is that the current year is always of great interest. But it is important to note that the data shown reflects only 7.5 months.

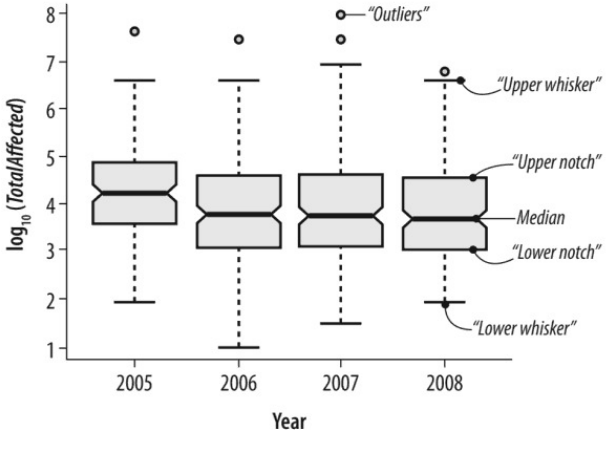

FIGURE 3-6. Compared severity of data breaches

The median is the value of $\log_{10}$(*TotalAffected*) for which half of the years' observations are either higher (or lower). The upper and lower notches approximate the third and first quadrant observations, respectively. The upper and lower whiskers reflect values that encompass a 95% confidence interval around the medians of two distributions. In this case, a distribution covers one year of reported breaches, so Figure 3-6 characterizes four distributions, one each for 2005, 2006, 2007, and 2008 (only 7.5 months). The high degree of overlap in all the distributions indicates that the number of individuals affected per breach varies by two orders of magnitude (two notches on the $\log_{10}$ vertical axis), from about 5,000 on the low end to 100,000 on the high end.

Copyrighted material

It would be interesting to determine the sensitivity of the *TotalAffected* value to the length of time it takes for a breach to be discovered. In the TJX case, that length of time was approximately 18 months. According to the DataLossDB, the *TotalAffected* value for the TJX breach is 94 million. Note that the callout labeled "Outlier" points to the bubble that represents TJX's breach.‡ Unfortunately, the DataLossDB does not track a value for breach time to discover.

Local metrics. Before leaving the topic of TJX, let us very briefly identify local metrics worth tracking, with a focus on internal operations. Perhaps a regular review of these metrics would have inspired TJX IT security operations first to detect and then to protect some of the holes that enabled the breach we have discussed.

Within the TJX IT infrastructure, vulnerability, patch management, and configuration management systems are examples of proactive security systems whose objective is to ensure that certain classes of weaknesses are discovered and repaired before hackers have a chance to exploit them. In performing their work, these systems create vast quantities of data that can be used to derive metrics that characterize how well they are meeting objectives. Two sample classes of metrics include:

Scan coverage

How much of the corporate network is actually being reviewed? Typically, this metric is expressed as a percentage, and is often broken down by location, owning business unit, type of device, type of vulnerability, or severity. Clearly, the goal is to achieve measurements as close as possible to 100%—i.e., total coverage.

It appears that coverage metrics for TJX WAPs were almost or totally lacking. TJX neglected to review the configuration of any of their wireless access points.

Available tools also complicate monitoring. Often, due to specialization, security management products cover only one type of target, such as routers, WAPs, servers, or even servers running a specific operating system. As is often the case, in 2002, wireless access points were delivered to market well in advance of the enterprise-class scanners capable of discovering vulnerabilities in their configuration. It is certainly possible that TJX was a victim of that gap. This is not to say that they shouldn't have known better.

Mean Time to Repair (MTTR)

How long (on average) does it take to remove a flaw that has been discovered (broadly defined) in the IT infrastructure? This metric is expressed in some unit of time: hours, days, weeks, etc. In general, the lower the value of this metric, the less time hackers will have to exploit vulnerabilities.

Although this metric is clearly valuable as a leading indicator of risk, it requires several prerequisites.

‡ $\log_{10}(94{,}000{,}000)$ is approximately 7.97.

Copyrighted material

First, sensors need to be installed to detect faults such as vulnerabilities, bad configurations, unpatched software, etc. Second, when a vulnerability is found, records must be kept that track the time it was discovered and the time it was validated as fixed. It is not clear that TJX had any mechanism in place to detect deployed wireless access points, much less poorly configured ones. So they were never fixed. MTTR equaled infinity.

If one can't prevent a breach, the next best strategy is to detect it quickly. It took TJX 18 months and hundreds of thousands of credit card numbers to become suspicious. What are some detection metrics that could have helped?

The thieves who stole the credit card numbers left numerous artifacts of their work:

Sensitive data (e.g., credit card numbers or SSNs) in outbound network traffic
: This is easily detected and can be blocked within minutes by existing commercially available tools generically called data usage monitors.

Unexplained account activity
: Having intercepted account credentials from authorized users, the hackers logged into accounts, searched databases, extracted data, created files, and performed file transfers to move their ill-gotten gains off to servers that they controlled. All of this activity is detectable via database management tools and, often, application logs.

Metrics that aggregate quantities based upon the data collected by data usage monitors, database management systems, and application logs would reflect the magnitude of activity that was required for the hackers to marshal and export their stolen materials. These numbers would stand out in any time series.

More Public Data Sources

Finally, it is worth noting that there are an increasing number of additional public sources for global threat information. This is progress that should not go unacknowledged:

Symantec DeepSight Threat Management Service[§]
: Provides views of the global security threat landscape by capturing and processing security log data from routers, firewalls, IDSs, and other devices. Limited metrics are available, aggregated by dimensions such as location, severity, and type. A sample report for July 2008 is available at *http://eval.symantec.com/mktginfo/enterprise/white_papers/b-whitepaper_internet_security_threat_report_xiii_04-2008.en-us.pdf*.

iDefense Labs (http://labs.idefense.com/)
: A subsidiary of VeriSign, iDefense Labs monitors threat as well as hacker activity and reports the results of its analyses to subscribers and the public.

§ *http://www.symantec.com/business/services/overview.jsp?pcid=hosted_services&pvid=deepsight_early_warning_services*

Copyrighted material

SANS Internet Storm Center (http://isc.sans.org/)
: This is a free and open service that uses a platform called DShield to allow users of firewalls to share intrusion information. The Internet Storm website extracts simple metrics and publishes reports that identify top ports attacked, top sources of attacks, country reports, and others.

Additional sources of interest are:[‖]

- *http://www.gocsi.com* for industry survey results
- *http://antiphishing.org* for phishing, malware, and data-theft data and statistics
- *http://nvd.nist.gov* for vulnerability data and statistics
- *http://www.commtouch.com/Site/ResearchLab/statistics.asp* for spam data and statistics
- *http://www.postini.com/stats* for spam data and statistics
- Symantec Internet Security Threat Reports (ISTR), issued annually
- *http://www.webroot.com* for spyware data and statistics
- *http://www.counterpane.com* for polity violations (e.g., IDS, IPS, etc.)
- *http://www.qualys.com/research/rnd/vulnlaws* for interesting analysis on vulnerability prevalence, persistence, and "half-life"

Summary

Security metrics are a critical prerequisite for turning IT security into a science instead of an art. The major ideas I have discussed in this chapter can be summarized as follows:

Metrics can make a difference
: Metrics can and do facilitate better awareness and better decision-making. They force a structure around the analysis of complex situations and shine a light on broken processes or anomalous activity. Once executives and managers know about something, they are usually capable of identifying a solution and monitoring its effects when enacted. In the cases of Barings and TJX, major failures might have been averted or significantly mitigated if only a metrics program were in place to provide critical transparency.

One size does not fit all
: Metrics that matter must address situations of critical interest as well as reflect a high degree of domain knowledge about the situations that they measure. In the discussion in "Security Metrics by Example" on page 38, we spent as much time discussing the business context as we did the metrics that might have demonstrated their beauty by helping to avert disaster.

[‖] Many thanks to Dr. D. E. Geer, Jr. for providing pointers to these sources.

Copyrighted material

You get what you give

If companies do not begin sharing more data about their security successes and failures, IT security management will be doomed to a morass of soothsaying, oracles, and lack of credibility. In the discussion in "Security Metrics by Example" on page 38, there was little data available for us to actually compute and present internal metrics. The DataLossDB data is far better than nothing, but, as a project staffed by volunteers, one can't expect it to expand its scope and scale without a corresponding increase in investment.

If, on the other hand, a security data-sharing culture takes hold, the IT security management discipline can grow up, become a science, and deliver quantitative metrics to help decision-makers invest in the infrastructure that is the foundation of our increasingly digital world.

Security metrics is an interdisciplinary pursuit

There is much to be learned from other disciplines, such as medical research. Security metricians must also be knowledgeable in IT security, in mathematics and statistics, in the visualization of quantitative information, in scalable software implementation, and in the business that the metrics will serve.

Security metrics is in its infancy and, in my opinion, has been stuck there for too many years. This chapter is about what I believe is required to give it a kick start. Toward that end, we first looked to the relative success of a more mature discipline with a lot of similarities, and then we applied what we learned to some noteworthy IT security failures.

Copyrighted material

Copyrighted material

CHAPTER FOUR

The Underground Economy of Security Breaches

Chenxi Wang

THE LATEST STATISTIC FROM NETCRAFT PUTS TODAY'S INTERNET at 185,497,213 sites. Though the absolute number suffered some loss lately due to the late economic downturn, the Internet growth at mid-2008 was measured at over 130,000 sites per day! It is estimated that the worldwide Internet user population will reach 500 million some time soon. The Internet is fast becoming one of the most significant markets in our modern economy.

Not surprisingly, just like its physical counterpart, the Internet is fostering one of the biggest underground economies.

As one might expect, this cyber underground has one main goal: *money*. The actors in this economy employ a wide array of digital ammunitions—including malware, botnets, and spam—to help them achieve this goal.

Unlike the physical world, where behavior can be held in check in most places by laws and regulations, the laws that govern the digital universe are, for all intents and purposes, ill-defined and poorly enforced. As a result, the cyber underground flourishes. In recent years, cyber attacks have graduated from the ad hoc, script-kiddie attacks to large-scale, organized crimes.

The 2007 CSI/FBI Study offers these statistics:

- The average annual loss due to computer crime jumped from $168,000 in 2004 to $350,424 in 2006. This figure represents a three-year high since 2004.

Copyrighted material

- For the first time in history, financial fraud overtook virus attacks as the largest cyber source for financial losses.

The cyber underground is, without a doubt, a global, thriving economy whose actors can reap handsome benefits from their activities and investments. The increasingly organized nature of the market and its growing sophistication mark a move toward automation and optimization, ultimately yielding higher financial gains for the criminals.

So how does this underground economy work? How do the various actors interact with each other and conduct illicit activities? This chapter presents high-level results of my investigation into these topics.

I'll end this article with some suggestions for ways we could disrupt the cyber underground or mitigate its destructive effects. But I'm not a law enforcement professional, so finding solutions is not my role. Rather, I hope to initiate an open discussion that will spur a community effort to combat the rise of this underground economy.

The Makeup and Infrastructure of the Cyber Underground

Perverse as it may sound, the cyber underground is a thriving community, with all kinds of collaboration and trading taking place every minute. The members of that community rarely work alone; it is a common practice for different parties to exchange assets (e.g., data and malware) to achieve mutually beneficial goals or shorten the time to launch an attack.

The cyber underground breaks down into an assortment of different actors, which I loosely classify as follows:

Malware producers
: Much of the malware for purchase today is of production quality: highly functional, easy to use, and effective. A professional malware writer goes through production cycles similar to those of a legitimate software producer, including testing, release, and frequent updates once out in the field.

Resource dealers
: These actors profit by selling computing or human resources. Computing resources often come from a botnet comprised of infected machines that can execute commands given remotely as part of an attack. Human resources represent actual hackers, residing in all corners of the world, waiting to be mobilized. A resource dealer's existence depends on the ability to tap into the massive botnet pool, and as such they are constantly on the lookout to amass more botnet resources. Their main mission is the creation, maintenance, and expansion of botnets.

Information dealers
: An information dealer sells valuable information—such as customer data, bank accounts, and security vulnerabilities—for a profit. Their main goal, therefore, is to gather more information of that nature. An information dealer is sometimes the customer of malware

Copyrighted material