SYNGRESS

Network and System Security

John R. Vacca

Network and System Security

This page intentionally left blank

Network and System Security

Editor

John R. Vacca

ELSEVIER

AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO
Syngress is an imprint of Elsevier

SYNGRESS

Syngress is an imprint of Elsevier
30 Corporate Drive, Suite 400, Burlington, MA 01803, USA
The Boulevard, Langford Lane, Kidlington, Oxford, OX5 1GB, UK
Network and System Security

Library of Congress Cataloging-in-Publication Data
Vacca, John R.
Network and system security / by John R. Vacca.
p. cm.
Includes bibliographical references and index.
ISBN 978-1-59749-535-6 (alk. paper)
1. Computer networks—Security measures. I. Title.
TK5105.59.V34 2010
005.8—dc22
2009052077

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

ISBN: 978-1-59749-535-6

Printed in the United States of America
10 11 12 13 10 9 8 7 6 5 4 3 2 1

This book is dedicated to my wife Bee

Contents

This page intentionally left blank

Foreword

Everyone wants to be connected. The use of computer networks has become almost universal. Where you find a computer you now generally find a network. However, without security, electronic communications hold little value and computer networks present significant security challenges, including protecting against network attacks, establishing physical control, and preventing unauthorized access. Security professionals and application developers, along with IT and network staff in all types of organizations, all need to do their part in assuring that network and system security issues are addressed.

This book provides an extensive analysis of network and system security practices, procedures, and technologies. Design issues and architectures are also expertly covered. But this book goes beyond theory and analysis to explain numerous implementation issues. This book is written for people that need to cut through the confusion about network security and get down to adoption and deployment. The book starts with the basic concepts and takes readers through all of the necessary learning steps to enable them to effectively secure computer networks and information systems.

Michael Erbschloe
Computer & Network Security Consultant

This page intentionally left blank

Acknowledgements

There are many people whose efforts on this book have contributed to its successful completion. I owe each a debt of gratitude and want to take this opportunity to offer my sincere thanks.

A very special thanks to my Senior Acquisitions Editor, Rick Adams, without whose continued interest and support would not have made this book possible. Associate Editor, David Bevans, who provided staunch support and encouragement when it was most needed. Thanks to my project manager, Andre Cuello; Copyeditor, Melissa Revell, whose fine editorial work has been invaluable. Thanks also to my marketing manager, Andrea Dierna, whose efforts on this book have been greatly appreciated. Finally, thanks to all of the other people at Syngress (an imprint of Morgan Kaufmann Publishers/Elsevier Science & Technology Books), whose many talents and skills are essential to a finished book.

Thanks to my wife, Bee Vacca, for her love, her help, and her understanding of my long work hours. Also, a very very special thanks to Michael Erbschloe for writing the foreword. Finally, I wish to thank all the following authors who contributed chapters that were necessary for the completion of this book: John R. Mallery, Scott R. Ellis, Michael A. West, Tom Chen, Patrick J. Walsh, Gerald Beuchelt, Mario Santana, Jesse Walker, Xinyuan Wang, Daniel Ramsbrock, Xuxian Jiang, Bill Mansoor, Pramod Pandya, Chunming Rong, Prof. Erdal Cayirci, Gansen Zhao, Laing Yan, Peng Liu, Thomas F. LaPorta and Kameswari Kotapati.

This page intentionally left blank

About the Editor

John Vacca is an information technology consultant and best selling author based in Pomeroy, Ohio. Since 1982, John has authored 65 books. Some of his most recent books include: *Computer And Information Security Handbook (Morgan Kaufman 2009); Biometric Technologies and Verification Systems* (Elsevier 2007); *Practical Internet Security* (Springer 2006); *Optical Networking Best Practices Handbook* (Wiley-Interscience 2006); *Guide to Wireless Network Security* (Springer 2006); *Computer Forensics: Computer Crime Scene Investigation*, *2nd Edition* (Charles River Media 2005); *Firewalls: Jumpstart for Network And Systems Administrators* (Elsevier 2004); *Public Key Infrastructure: Building Trusted Applications and Web Services (*Auerbach 2004); *Identity Theft* (Prentice Hall\PTR 2002); *The World's 20 Greatest Unsolved Problems* (Pearson Education 2004); and more than 600 articles in the areas of advanced storage, computer security and aerospace technology. John was also a configuration management specialist, computer specialist, and the computer security official (CSO) for NASA's space station program(Freedom) and the International Space Station Program, from 1988 until his early retirement from NASA in 1995.

This page intentionally left blank

Contributors

Michael Erbschloe (FOREWORD), Teaches Information Security courses at Webster University in St. Louis, Missouri.

John R. Mallery (CHAPTER 1), BKD, LLP, Twelve Wyandotte Plaza, 120 West 12th Street, Suite 1200, Kansas City, Missouri 64105-1936

Scott R. Ellis (CHAPTER 2), Forensics and Litigation Technology, RGL Forensics, 33 N. Dearborn Street, Suite 1310, Chicago IL, 60602

Michael A. West (CHAPTER 3), Independent Technical Writer, 636 Fig Tree Lane, Martinez, California 94553

Tom Chen (CHAPTER 4), Swansea University, Singleton Park, Swansea SA2 8PP, Wales, UK

Patrick J. Walsh (CHAPTER 4), eSoft Inc., 295 Interlocken Blvd., Suite 500, Broomfield, Colorado 80021

Gerald Beuchelt (CHAPTER 5), Independent Security Consultant, 13 Highland Way, Burlington, MA 01803

Mario Santana (CHAPTER 6), Terremark, 3200 Main St, Dallas, TX. 75226

Jesse Walker (CHAPTER 7), Intel Corporation, 2211 NE 25th Avenue, Hillboro, OR 97124

Xinyuan Wang (CHAPTER 8), Department of Computer Science, George Mason University, 4400 University Drive, MSN 4A4, Fairfax, VA 22030

Daniel Ramsbrock (Co-Author) (CHAPTER 8), Department of Computer Science, George Mason University, 4400 University Drive, MSN 4A4, Fairfax, VA 22030

Xuxian Jiang (Co-Author) (CHAPTER 8), Department of Computer Science, North Carolina State University, 890 Oval Drive, Campus Box 8206, Raleigh, NC 27695-8206

Bill Mansoor (CHAPTER 9), Information Systems Audit and Control Association (ISACA), 95 Bloomfield Lane, Rancho Santa Margarita, CA 92688-8741

Pramod Pandya (CHAPTER 10), Department of Information Systems and Decision Sciences, California State University, Fullerton, CA 92834

Chunming Rong (CHAPTER(S) 11, 13), Professor, Ph.D., Chair of Computer Science Section, Faculty of Science and Technology, University of Stavanger, N-4036 Stavanger, NORWAY

Prof. Erdal Cayirci (Co-Author) (CHAPTER(S) 11, 13), University of Stavanger, N-4036 Stavanger, NORWAY

Gansen Zhao (Co-Author) (CHAPTER(S) 11, 13), University of Stavanger, N-4036 Stavanger, NORWAY

Laing Yan (Co-Author) (CHAPTER(S) 11, 13), University of Stavanger, N-4036 Stavanger, NORWAY

Kameswari Kotapati (CHAPTER 12), Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802

Peng Liu (CHAPTER 12), College of Information Sciences and Technology, The Pennsylvania State University, University Park, PA 16802

Thomas F. LaPorta (CHAPTER 12), Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802

Chunming Rong (CHAPTER(S) 11, 13), Professor, Ph.D., Chair of Computer Science Section, Faculty of Science and Technology, University of Stavanger, N-4036 Stavanger, NORWAY

Prof. Erdal Cayirci (Co-Author) (CHAPTER(S) 11, 13), University of Stavanger, N-4036 Stavanger, NORWAY

Gansen Zhao (Co-Author) (CHAPTER(S) 11, 13), University of Stavanger, N-4036 Stavanger, NORWAY

Laing Yan (Co-Author) (CHAPTER(S) 11, 13), University of Stavanger, N-4036 Stavanger, NORWAY

Introduction

Organizations today are linking their systems across enterprise-wide networks and virtual private networks (VPNs), as well as increasing their exposure to customers, competitors, browsers and hackers on the Internet.

According to industry analysts, NAC is now the "Holy Grail" of network security, but NAC isn't the sole contributor to the booming security market. According to industry analysts, hackers are inventing new ways to attack corporate networks, and vendors are just as quickly devising ways to protect against them. Those innovations will continue to push the security market higher.

First, there's a real need for enterprise-class security for handheld devices, especially wireless client devices, such as Wi-Fi VoIP handsets. Second, as the next step in perimeter security, network IPS is beginning to make the transition from niche security technology to core network infrastructure. And, finally, enterprises are fed up with viruses, spyware and malware, and are willing to make significant investments to put a stop to them. Industry analysts have identified the following trends in the burgeoning security market:

- Software, hardware appliances and security routers are the preferred security for most respondents and will continue to be through 2010. Secure routers show the most growth.
- Fifty percent of respondents have purchased wireless LAN security products, while 31% said they will buy or are considering buying WLAN security.
- The need to block viruses and the fear of hackers are prompting respondents to buy security products and services en masse.
- Increased service reliability is the most important payback respondents expect from managed security service. Respondents also thought organizations should focus on core competencies, have access to more advanced technology and have access to better expertise.

In this book, you will learn how to analyze risks to your networks and the steps needed to select and deploy the appropriate countermeasures to reduce your exposure to physical and network threats. This book will enhance the skills and knowledge of practitioners and IT professionals who need to identify and counter some fundamental security risks and requirements. Practitioners and IT professionals will learn some advanced network security skills pertaining to network threat identification and prevention. They will also examine Internet security threats and measures (audit trails IP sniffing/spoofing etc. . . .) and learn how to implement advanced security policies and procedures. In addition, in this book, you will also learn how to:

1. Secure UNIX and Linux systems from internal and external threats
2. Establish authenticated access to local and remote resources
3. Avoid potential security loopholes by limiting super user privileges
4. Protect UNIX file systems
5. Configure tools and utilities to minimize exposure and detect intrusions
6. Tackle security problems by swapping out insecure software components
7. Add tools and services to increase security
8. Create, document and test continuity arrangements for your organization
9. Perform a risk assessment and Business Impact Assessment (BIA) to identify vulnerabilities
10. Select and deploy an alternate site for continuity of mission-critical activities
11. Identify appropriate strategies to recover the infrastructure and processes
12. Test and maintain an effective recovery plan in a rapidly changing technology environment
13. Detect and respond to vulnerabilities that put your organization at risk using scanners
14. Employ real-world exploits and evaluate their effect on your systems
15. Analyze the results of vulnerability scans
16. Assess vulnerability alerts and advisories
17. Build a firewall to protect your network
18. Install and configure proxy-based and stateful-filtering firewalls
19. Provide access to HTTP and FTP services on the Internet
20. Implement publicly accessible servers without compromising security
21. Protect internal IP addresses with NAT and deploy a secure DNS architecture
22. Identify security threats to your data and IT infrastructure
23. Recognize appropriate technology to deploy against these threats
24. Adapt your organization's information security policy to operational requirements and assess compliance
25. Effectively communicate information security issues

In addition, you will also gain the skills needed to secure your UNIX and Linux platforms. You will learn to use tools and utilities to assess vulnerabilities, detect configurations that threaten information assurance and provide effective access controls.

You will also learn to identify vulnerabilities and implement appropriate countermeasures to prevent and mitigate threats to your mission-critical processes. You will learn techniques for creating a business continuity plan (BCP) and the methodology for building an infrastructure that supports its effective implementation.

Knowledge of vulnerability assessment and hacking techniques allows you to detect vulnerabilities before your networks are attacked. In this book, you will learn to configure and use vulnerability scanners to detect weaknesses and prevent network exploitation. You will also acquire the knowledge to assess the risk to your enterprise from an array of vulnerabilities and to minimize your exposure to costly threats.

Organization of This Book

The book is composed of 13 contributed chapters by leading experts in their fields; as well as, 10 Appendices, including an extensive glossary of computer security terms and acronyms at the back.

Contributor John Mallery (Chapter 1, "Building a Secure Organization"), begins by setting the stage for the rest of the book by showing insight on where to start building a secure organization. It seems logical that any business, whether a commercial enterprise or a not-for-profit business, would understand that building a secure organization is important to long-term success. When a business implements and maintains a strong security posture, it can take advantage of numerous benefits. An organization that can demonstrate an infrastructure protected by robust security mechanisms can potentially see a reduction in insurance premiums being paid. A secure organization can use its security program as a marketing tool, demonstrating to clients that it values their business so much that it takes a very aggressive stance on protecting their information. But most important, a secure organization will not have to spend time and money identifying security breaches and responding to the results of those breaches.

Security breaches can cost an organization significantly through a tarnished reputation, lost business, and legal fees. And numerous regulations, such as the Health Insurance Portability and Accountability Act (HIPAA), the Graham-Leach-Bliley Act (GLBA), and the Sarbanes-Oxley Act, require businesses to maintain the security of information. Despite the benefits of maintaining a secure organization and the potentially devastating consequences of not doing so, many organizations have poor security mechanisms, implementations, policies, and culture.

The steps to achieving security mentioned in this chapter are only the beginning. They should provide some insight into where to start building a secure organization.

Next, contributor Scott R. Ellis (Chapter 2, "A Cryptography Primer,") provides an overview of cryptography. He shows how communications may be encrypted and transmitted. Man is a warrior creature, a species that ritually engages in a type of warfare where the combat can range from the subtlety of inflicting economic damage, or achieving economic superiority and advantage, to moving someone's chair a few inches from sitting distance or putting rocks in their shoes, to the heinousness of the outright killing of our opponents. As such, it is in our nature to want to prevent others who would do us harm from intercepting private communications (which could be about them!). Perhaps nothing so perfectly illustrates this fact as the art of cryptography. It is, in its purpose, an art form entirely devoted to the methods whereby we can prevent information from falling into the hands of those who would use it against us—our enemies. In essence, computer-based cryptography is the art of creating a form of communication that embraces the following precepts:

- Can be readily understood by the intended recipients
- Cannot be understood by unintended recipients
- Can be adapted and changed easily with relatively small modifications, such as a changed pass phrase or word

Furthermore, reports that AES is not as strong as it should be are likely, at this time, to be overstated and inaccurate, because anyone can present a paper that is dense and difficult to understand and claims to achieve the incredible. It is unlikely that, any time in the near or maybe not-so-near future (this contributor hedges his bets), AES will be broken using multivariate quadratic polynomials in thousands of dimensions. Mathematica is very likely one of the most powerful tools that can solve quadratic equations, and it is still many years away from being able to perform this feat.

Then, contributor Michael West (Chapter 3, "Preventing System Intrusions") discusses how to prevent system intrusions, where an unauthorized- penetration of a computer in your enterprise occurs or an address in your assigned domain. The moment you establish an active Web presence, you put a target on your company's back. And like the hapless insect that lands in the spider's web, your company's size determines the size of the disturbance you create on the Web—and how quickly you're noticed by the bad guys. How attractive you are as prey is usually directly proportionate to what you have to offer a predator. If yours is an ecommerce site whose business thrives on credit card or other financial information or a company with valuable secrets to steal, your "juiciness" quotient goes up; you have more of value there to steal. And if your business is new and your Web presence is recent, the assumption could be made that perhaps you're not yet a seasoned veteran in the nuances of cyber warfare and, thus, are more vulnerable to an intrusion.

Unfortunately for you, many of those who seek to penetrate your network defenses are educated, motivated, and quite brilliant at developing faster and more efficient methods of quietly sneaking around your perimeter, checking for the smallest of openings. Most IT professionals know that an enterprise's firewall is ceaselessly being probed for weaknesses and vulnerabilities by crackers from every corner of the globe. Anyone who follows news about software understands that seemingly every few months, word comes out about a new, exploitable opening in an operating system or application. It's widely understood that no one—not the most savvy network administrator or the programmer who wrote the software—can possibly find and close all the holes in today's increasingly complex software.

Bugs exist in applications, operating systems, server processes (daemons), and clients. System configurations can also be exploited, such as not changing the default administrator's password or accepting default system settings, or unintentionally leaving a hole open by configuring the machine to run in a nonsecure mode. Even Transmission Control Protocol/Internet Protocol (TCP/IP), the foundation on which all Internet traffic operates, can be exploited, since the protocol was designed before the threat of hacking was really widespread. Therefore, it contains design flaws that can allow, for example, a cracker to easily alter IP data.

Once the word gets out that a new and exploitable opening exists in an application (and word *will* get out), crackers around the world start scanning sites on the Internet searching for any and all sites that have that particular opening. Making your job even harder is the fact that many openings into your network can be caused by your employees. Casual surfing of porn sites can expose the network to all kinds of nasty bugs and malicious code, merely by an employee visiting the site. The problem is that, to users, it might not seem like such a big deal. They either don't realize or don't care that they're leaving the network wide open to intrusion.

Preventing network intrusions is no easy task. Like cops on the street—usually outnumbered and under equipped compared to the bad guys—you face an enemy with determination, skill, training, and a frightening array of increasingly sophisticated tools for hacking their way through your best defenses. And, no matter how good your defenses are today, it's only a matter of time before a tool is developed that can penetrate them. If you know that ahead of time, you'll be much more inclined to keep a watchful eye for what "they" have and what you can use to defeat them.

Your best weapon is a logical, thoughtful, and nimble approach to network security. You have to be nimble—to evolve and grow with changes in technology, never being content to keep things as they are because "Hey, they're working just fine." Today's "just fine" will be tomorrow's "What the hell happened?"

Stay informed. There is no shortage of information available to you in the form of white papers, seminars, contract security specialists, and online resources, all dealing with various aspects of network security.

Have a good, solid, comprehensive, yet easy-to-understand network security policy in place. The very process of developing one will get all involved parties thinking about how to best secure your network while addressing user needs. When it comes to your users, you simply can't overeducate them where network security awareness is concerned. The more they know, the better equipped they'll be to act as allies against, rather than accomplices of, the hoards of crackers looking to steal, damage, hobble, or completely cripple your network.

Do your research and invest in good, multipurpose network security systems. Select systems that are easy to install and implement, are adaptable and quickly configurable, can be customized to suit your needs of today as well as tomorrow, and are supported by companies that keep pace with current trends in cracker technology.

Contributors Tom Chen and Patrick Walsh (Chapter 4, "Guarding Against Network Intrusions") continue by showing how to guard against network intrusions, by understanding the variety of attacks from exploits to malware to social engineering. Virtually all computers today are connected to the Internet through dialup, broadband, Ethernet, or wireless technologies. The reason for this Internet ubiquity is simple: Applications depending on the network, such as email, Web, remote login, instant messaging, and VoIP, have become essential to the computing experience. Unfortunately, the Internet exposes computer users to risks from a wide variety of possible attacks. Users have much to lose—their privacy, valuable data, control of their computers, and possibly theft of their identities. The network enables attacks to be carried out remotely, with relative anonymity and low risk of traceability.

The nature of network intrusions has evolved over the years. A few years ago, a major concern was fast worms such as Code Red, Nimda, Slammer, and Sobig. More recently, concerns shifted to spyware, Trojan horses, and botnets. Although these other threats still continue to be major problems, the Web has become the primary vector for stealthy attacks today.

Next, contributor Gerald Beuchelt (Chapter 5, "UNIX and Linux Security") discusses how to scan for vulnerabilities; reduce denial-of-service (DoS) attacks; deploy firewalls to control network traffic; and, build network firewalls. When **Unix** was first booted on a PDP-8 computer at Bell Labs, it already had a basic notion of user isolation, separation of kernel and user memory space, and process security. It was originally conceived as a multiuser system, and as such, security could not be added on as an afterthought. In this respect, **Unix** was different from a whole class of computing machinery that had been targeted at single-user environments.

The examples in this chapter refer to the Solaris operating system and Debian-based Linux distributions, a commercial and a community developed operating system. Solaris is freely available in open source and binary distributions. It derives directly from AT&T System V R4.2 and is one of the few operating systems that can legally be called **Unix**. It is distributed by Sun Microsystems, but there are independent distributions built on top of the open source version of Solaris.

Then, contributor Mario Santana (Chapter 6, "Securing Linux and UNIX Operating Systems") presents an introduction to securing UNIX in general and Linux in particular, presenting some historical context and describing some fundamental aspects of the secure operating system architecture. As an operating system designed to be flexible and robust, **Unix** lends itself to providing a wide array of host- and network-based services. **Unix** also has a rich culture from its long history as a fundamental part of computing research in industry and academia. **Unix** and related operating systems play a key role as platforms for delivering the key services that make the Internet possible.

For these reasons, it is important that information security practitioners understand fundamental **Unix** concepts in support of practical knowledge of how **Unix** systems might be securely operated. This chapter is an introduction to **Unix** in general and to Linux in particular, presenting some historical context and describing some fundamental aspects of the operating system architecture. Considerations for hardening **Unix** deployments will be contemplated from network-centric, host-based, and systems management perspectives. Finally, proactive considerations are presented to identify security weaknesses to correct them and to deal effectively with security breaches when they do occur.

Especially when duties are appropriately separated, unannounced forced vacations are a powerful way to bring fresh perspectives to security tasks. It's also an effective deterrent to internal fraud or mismanagement of security responsibilities.

Contributor Jesse Walker (Chapter 7, "Internet Security") continues by showing you how cryptography can be used to address some of the security issues besetting communications protocols. The Internet, and all its accompanying complications, has become integral to our lives. The security problems besetting the Internet are legendary and have been daily annoyances to many users. Given the Net's broad impact on our lives and the widespread security issues associated with, it is worthwhile understanding what can be done to improve the immunity of our communications from attack.

The Internet can serve as a laboratory for studying network security issues; indeed, we can use it to study nearly every kind of security issue. Walker will pursue only a modest set of questions related to this theme. The goal of this chapter is to understand how cryptography can be used to address some of the security issues besetting communications protocols. To do so, it will be helpful to first understand the Internet architecture. After that, he will survey the types of attacks that are possible against communications. With this background he will be in a position to understand how cryptography can be used to preserve the confidentiality and integrity of messages.

Walker's goal is modest. It is only to describe the network architecture and its cryptographic-based security mechanisms sufficiently to understand some of the major issues confronting security systems designers and to appreciate some of the major design decisions they have to make to address these issues.

This chapter also examines how cryptography is used on the Internet to secure protocols. It reviews the architecture of the Internet protocol suite, as even what security means is a function of the underlying system architecture. Next, it reviews the Dolev-Yao model, which describes the threats to which network communications are exposed. In particular, all levels of network protocols are completely exposed to eavesdropping and manipulation by an attacker, so using cryptography properly is a first-class requirement to derive any benefit from its use. Walker also shows you that effective security mechanisms to protect session-oriented and session establishment protocols are different, although they can share many cryptographic primitives. Cryptography can be very successful at protecting messages on the Internet, but doing so requires pre-existing, long-lived relationships. How to build secure open communities is still an open problem; it is probably intractable because a solution would imply the elimination of conflict between human beings who do not know each other.

Next, contributors Xinyuan Wang, Daniel Ramsbrock and Xuxian Jiang (Chapter 8, "Internet Security: The Botnet Problem in Internet Security,") describe the botnet threat and the countermeasures available to network security professionals. This chapter describes the botnet threat and the countermeasures available to network security professionals. First, it provides an overview of botnets, including their origins, structure, and underlying motivation. Next, the chapter describes existing methods for defending computers and networks against botnets. Finally, it addresses the most important aspect of the botnet problem: how to identify and track the botmaster in order to eliminate the root cause of the botnet problem.

Botnets are one of the biggest threats to the Internet today, and they are linked to most forms of Internet crime. Most spam, DDoS attacks, spyware, click fraud, and other attacks originate from botnets and the shadowy organizations behind them. Running a botnet is immensely profitable, as several recent high-profile arrests have shown. Currently, many botnets still rely on a centralized IRC C&C structure, but more and more botmasters are using P2P protocols to provide resilience and avoid a single point of failure. A recent large-scale example of a P2P botnet is the Storm Worm, widely covered in the media.

A number of botnet countermeasures exist, but most are focused on bot detection and removal at the host and network level. Some approaches exist for Internet-wide detection and disruption of entire botnets, but we still lack effective techniques for combating the root of the problem: the botmasters who conceal their identities and locations behind chains of stepping-stone proxies.

The three biggest challenges in botmaster traceback are stepping stones, encryption, and the low traffic volume. Even if these problems can be solved with a technical solution, the trace must be able to continue beyond the reach of the Internet. Mobile phone networks, open wireless access points, and public computers all provide an additional layer of anonymity for the botmasters.

Short of a perfect solution, even a partial traceback technique could serve as a very effective deterrent for botmasters. With each botmaster that is located and arrested, many botnets will be eliminated at once. Additionally, other botmasters could decide that the risks outweigh the benefits when they see more and more of their colleagues getting caught. Currently, the economic equation is very simple: Botnets can generate large profits with relatively low risk of getting caught. A botmaster traceback solution, even if imperfect, would drastically change this equation and convince more botmasters that it simply is not worth the risk of spending the next 10–20 years in prison.

Then, contributor Bill Mansoor (Chapter 9, "Intranet Security") covers internal security strategies and tactics; external security strategies and tactics; network access security; and, Kerberos. Thus, the onus of preventing embarrassing security gaffes falls squarely on the shoulders of IT security chiefs (CISOs and security officers). These CISOs, are sometimes hobbled by unclear mandates from government regulators and lack of sufficient budgeting to tackle the mandates.

It is true that the level of Internet hyperconnectivity among generation X and Y users has mushroomed lately, and the network periphery that we used to take for granted as a security shield has been diminished, to a large extent, because of the explosive growth of social networking and the resulting connectivity boom. However, with the various new types of incoming application traffic (VoIP, SIP, and XML traffic) to their networks, security administrators need to stay on their toes and deal with these new protocols by implementing newer tools and technology. One recent example of new technology is the application-level firewall for connecting outside vendors to intranets (also known as an XML firewall, placed within a DMZ) that protects the intranet from malformed XML and SOAP message exploits coming from outside sourced applications.

So, with the myriad security issues facing intranets today, most IT shops are still well equipped to defend themselves if they assess risks and, most important, train their employees regarding data security practices on an ongoing basis. The problems with threat mitigation remain largely a matter of meeting gaps in procedural controls rather than technical measures. Trained and security-aware employees are the biggest deterrent to data thefts and security breaches.

Contributor Dr. Pramod Pandya (Chapter 10, "Local Area Network Security,") continues by discussing network design and security deployment; and, ongoing management and auditing. Securing available resources on any corporate or academic data network is of paramount importance because most of these networks connect to the Internet for commercial or research activities. Therefore, the network is under attack from hackers on a continual basis, so network security technologies are ever evolving and playing catch-up with hackers. Around 20 years ago the number of potential users was small and the scope of any activity on the network was limited to local networks only. As the Internet expanded in its reach across national boundaries and as the number of users increased, potential risk to the network grew exponentially. Over the past 10 years, ecommerce-related activities such as online shopping, banking, stock trading, and social networking have permeated extensively, creating a dilemma for both service providers and their potential clients, as to who is a trusted service provider and a trusted client on the network. Of course, this being a daunting task for security professionals, they have needed to design security policies appropriate for both the servers and their clients. The security policy must be a factor in the clients' level of access to the resources. So, in whom do we place trust, and how much trust?

Securing network systems is an ongoing process in which new threats arise all the time. Consequently, firewalls, NIDS, and intrusion prevention systems are continuously evolving technologies. In this chapter, Pandya's focus has been and will be wired networks. However, as wireless data networks proliferate and seamlessly connect to the cellular voice networks, the risk of attacks on the wired networks is growing exponentially.

In addition, the responsibility for the design and implementation of network security, should be headed by the chief information officer (CIO) of the enterprise network. The CIO has a pool of network administrators and legal advisers to help with this task. The network administrators define the placing of the network access controls, and the legal advisors underline the consequences and liabilities in the event of network security breaches. We have seen cases of customer records such as credit card numbers, Social Security numbers, and personal information being stolen. The frequency of these reports have been on the increase in the past years, and consequently this has led to a discussion on the merits of encryption of stored data. One of the most quoted legal requirements on the part of any business, whether small or big, is the protection of consumer data under the Health Insurance Portability and Accountability Act (HIPAA), which restricts disclosure of health-related data and personal information.

Next, contributors Chunming Rong, Erdal Cayirci, Gansen Zhao and Laing Yan (Chapter 11, "Wireless Network Security") present an overview of wireless network security technology; how to- design wireless network security, plan for wireless network security; install and deploy wireless network security, and maintain wireless network security; information warfare countermeasures: the wireless network security solution; and, wireless network security solutions and future directions. With the rapid development of technology in wireless communication and microchips, wireless technology has been widely used in various application areas. The proliferation of wireless devices and wireless networks in the past decade shows the widespread of wireless technology.

Wireless networks is a general term to refer to various types of networks that are wireless, meaning that they communicate without the need of wire lines. Wireless networks can be broadly categorized into two classes based on the structures of the networks: wireless ad hoc networks and cellular networks. The main difference between these two network classes is whether a fixed infrastructure is present.

Three of the well-known cellular networks are the GSM network, the CDMA network, and the 802.11 wireless LAN. The GSM network and the CDMA network are the main network technologies that support modern mobile communication, with most of the mobile phones and mobile networks that are built based on these two wireless networking technologies and their variants. As cellular networks required fixed infrastructures to support the communication between mobile nodes, deployment of the fixed infrastructures is essential. Further, cellular networks require serious and careful topology design of the fixed infrastructures before deployment, because the network topologies of the fixed infrastructures are mostly static and will have a great impact on network performance and network coverage.

Then, contributors Peng Liu, Thomas F. LaPorta, and Kameswari Kotapati (Chapter 12, "Cellular Network Security"), address the security of the cellular network; educate readers on the current state of security of the network and its vulnerabilities; outline the cellular network specific attack taxonomy, also called *three dimensional attack taxonomy*; discuss the vulnerability assessment tools for cellular networks; and, provides insights as to why the network is so vulnerable, and why securing it can prevent communication outages during emergencies.

In recent years, cellular networks have become open public networks to which end subscribers have direct access. This has greatly increased the threats to the cellular network. Though cellular networks have vastly advanced in their performance abilities, the security of these networks still remains highly outdated. As a result, they are one of the most insecure networks today—so much so that using simple off-the-shelf equipment, any adversary can cause major network outages affecting millions of subscribers.

In this chapter, Liu, LaPorta, and Kotapati, address the security of the cellular network. They educate readers on the current state of security of the network and its vulnerabilities. They also outline the cellular network specific attack taxonomy, also called the *three-dimensional attack taxonomy*. They then discuss the vulnerability assessment tools for cellular networks. Finally, they provide insights as to why the network is so vulnerable and why securing it can prevent communication outages during emergencies.

Cellular networks are high-speed, high-capacity voice and data communication networks with enhanced multimedia and seamless roaming capabilities for supporting cellular devices. With the increase in popularity of cellular devices, these networks are used for more than just entertainment and phone calls. They have become the primary means of communication for finance-sensitive business transactions, lifesaving emergencies, and life-/mission-critical services such as E-911. Today these networks have become the lifeline of communications.

A breakdown in the cellular network has many adverse effects, ranging from huge economic losses due to financial transaction disruptions; loss of life due to loss of phone calls made to emergency workers; and communication outages during emergencies such as the September 11, 2001, attacks. Therefore, it is a high priority for the cellular network to function accurately.

It must be noted that it is not difficult for unscrupulous elements to break into the cellular network and cause outages. The major reason for this is that cellular networks were not designed with security in mind. They evolved from the old-fashioned telephone networks that were built for performance. To this day, the cellular network has numerous well-known and unsecured vulnerabilities providing access to adversaries. Another feature of cellular networks is network relationships (also called *dependencies*) that cause certain types of errors to propagate to other network locations as a result of regular network activity. Such propagation can be very disruptive to the network, and in turn it can affect subscribers. Finally, Internet connectivity to the cellular network is another major contributor to the cellular network's vulnerability because it gives Internet users direct access to cellular network vulnerabilities from their homes.

To ensure that adversaries do not access the network and cause breakdowns, a high level of security must be maintained in the cellular network. However, though great efforts have been made to improve the cellular network in terms of support for new and innovative services, greater number of subscribers, higher speed, and larger bandwidth, very little has been done to update the security of the cellular network. Accordingly, these networks have become highly attractive targets to adversaries, not only because of their lack of security but also due to the ease with which these networks can be exploited to affect millions of subscribers.

In this chapter, the contributors analyze the security of cellular networks. Toward understanding the security issues in cellular networks, the rest of the chapter is organized as follows. They present a comprehensive overview of cellular networks with a goal of providing a fundamental understanding of their functioning. Next, they present the current state of cellular network security through an in-depth discussion on cellular network vulnerabilities and possible attacks. In addition, they present the cellular network specific attack taxonomy. Finally, they present a review of current cellular network vulnerability assessment techniques and conclude with a discussion.

Next to the Internet, the cellular network is the most highly used communication network. It is also the most vulnerable, with inadequate security measures making it a most attractive target to adversaries that want to cause communication outages during emergencies. As the cellular network is moving in the direction of the Internet, becoming an amalgamation of several types of diverse networks, more attention must be paid to securing these networks. A push from government agencies requiring mandatory security standards for operating cellular networks would be just the momentum needed to securing these networks.

Of all the attacks discussed in this chapter, cascading attacks have the most potential to stealthily cause major network disoperation. At present there is no standardized scheme to protect from such attacks. EndSec is a good solution for protecting from cascading attacks, since it requires every data item to be signed by the source service node. Because service nodes are unlikely to corrupt data items they are to be accounted for by their signatures, the possibility of cascading attacks is greatly reduced. EndSec has the added advantage of providing end-to-end security for all types of signaling messages. Hence, standardizing EndSec and mandating its deployment would be a good step toward securing the network.

Both Internet and PSTN connectivity are the open gateways that adversaries can use to gain access and attack the network. Because the PSTN's security is not going to be improved, at least its gateway to the core network must be adequately secured. Likewise, since neither the Internet's design nor security will to be changed to suit the cellular network, at least its gateways to the core network must be adequately secured.

So, because the cellular network is an amalgamation of many diverse networks, it has too many vulnerable points. Hence, the future design of the network must be planned to reduce the number of vulnerable networks points and reduce the number of service nodes that participate in servicing the subscriber, thereby reducing the number of points from which an adversary may attack.

Finally, contributors Chunming Rong, Erdal Cayirci, Gansen Zhao and Laing Yan (Chapter 13, "RFID Security") describe the RFID tags and RFID reader and back-end database in detail. Radio frequency identification (RFID) systems use RFID tags to annotate and identify objects. When objects are processed, an RFID reader is used to read information from the tags attached to the objects. The information will then be used with the data stored in the back-end databases to support the handling of business transactions. Generally, an RFID system consists of three basic components: RFID tags, RFID readers, and a back-end database.

- *RFID tags or RFID transponders.* These are the data carriers attached to objects. A typical RFID tag contains information about the attached object, such as an identifier (ID) of the object and other related properties of the object that may help to identify and describe it.
- *The RFID reader or the RFID transceiver.* These devices can read information from tags and may write information into tags if the tags are rewritable.
- *Back-end database.* This is the data repository responsible for the management of data related to the tags and business transactions, such as ID, object properties, reading locations, reading time, and so on.

John R. Vacca
Editor-in-Chief
jvacca@frognet.net
http://www.johnvacca.com

This page intentionally left blank

CHAPTER 1

Building a Secure Organization

John Mallery
BKD, LLP

It seems logical that any business, whether a commercial enterprise or a not-for-profit business, would understand that building a secure organization is important to long-term success. When a business implements and maintains a strong security posture, it can take advantage of numerous benefits. An organization that can demonstrate an infrastructure protected by robust security mechanisms can potentially see a reduction in insurance premiums being paid. A secure organization can use its security program as a marketing tool, demonstrating to clients that it values their business so much that it takes a very aggressive stance on protecting their information. But most important, a secure organization will not have to spend time and money identifying security breaches and responding to the results of those breaches.

As of September 2008, according to the National Conference of State Legislatures, 44 states, the District of Columbia, and Puerto Rico had enacted legislation requiring notification of security breaches involving personal information [1]. Security breaches can cost an organization significantly through a tarnished reputation, lost business, and legal fees. And numerous regulations, such as the Health Insurance Portability and Accountability Act (HIPAA), the Gramm–Leach–Bliley Act (GLBA), and the Sarbanes–Oxley Act, require businesses to maintain the security of information. Despite the benefits of maintaining a secure organization and the potentially devastating consequences of not doing so, many organizations have poor security mechanisms, implementations, policies, and culture.

1. Obstacles to Security

In attempting to build a secure organization, we should take a close look at the obstacles that make it challenging to build a totally secure organization.

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00001-4

Security Is Inconvenient

Security, by its very nature, is inconvenient, and the more robust the security mechanisms, the more inconvenient the process becomes. Employees in an organization have a job to do; they want to get to work right away. Most security mechanisms, from passwords to multifactor authentication, are seen as roadblocks to productivity. One of the current trends in security is to add whole disk encryption to laptop computers. Although this is a highly recommended security process, it adds a second login step before a computer user can actually start working. Even if the step adds only one minute to the login process, over the course of a year this adds up to four hours of lost productivity. Some would argue that this lost productivity is balanced by the added level of security. But across a large organization, this lost productivity could prove significant.

To gain a full appreciation of the frustration caused by security measures, we have only to watch the Transportation Security Administration (TSA) security lines at any airport. Simply watch the frustration build as a particular item is run through the scanner for a third time while a passenger is running late to board his flight. Security implementations are based on a sliding scale; one end of the scale is total security and total inconvenience, the other is total insecurity and complete ease of use. When we implement any security mechanism, it should be placed on the scale where the level of security and ease of use match the acceptable level of risk for the organization.

Computers Are Powerful and Complex

Home computers have become storehouses of personal materials. Our computers now contain wedding videos, scanned family photos, music libraries, movie collections, and financial and medical records. Because computers contain such familiar objects, we have forgotten that computers are very powerful and complex devices. It wasn't that long ago that computers as powerful as our desktop and laptop computers would have filled one or more very large rooms. In addition, today's computers present a "user-friendly" face to the world. Most people are unfamiliar with the way computers truly function and what goes on "behind the scenes." Things such as the Windows Registry, ports, and services are completely unknown to most users and poorly understood by many computer industry professionals. For example, many individuals still believe that a Windows login password protects data on a computer. On the contrary—someone can simply take the hard drive out of the computer, install it as a slave drive in another computer, or place it in a USB drive enclosure, and all the data will be readily accessible.

Computer Users Are Unsophisticated

Many computer users believe that because they are skilled at generating spreadsheets, word processing documents, and presentations, they "know everything about computers."

These "power users" have moved beyond application basics, but many still do not understand even basic security concepts. Many users will indiscriminately install software and visit questionable Web sites despite the fact that these actions could violate company policies. The "bad guys"—people who want to steal information from or wreak havoc on computers systems—have also identified that the average user is a weak link in the security chain. As companies began investing more money in perimeter defenses, attackers look to the path of least resistance. They send malware as attachments to email, asking recipients to open the attachment. Despite being told not to open attachments from unknown senders or simply not to open attachments at all, employees consistently violate this policy, wreaking havoc on their networks. The "I Love You Virus" spread very rapidly in this manner. More recently, phishing scams have been very effective in convincing individuals to provide their personal online banking and credit-card information. Why would an attacker struggle to break through an organization's defenses when end users are more than willing to provide the keys to bank accounts? Addressing the threat caused by untrained and unwary end users is a significant part of any security program.

Computers Created Without a Thought to Security

During the development of personal computers (PCs), no thought was put into security. Early PCs were very simple affairs that had limited computing power and no keyboards and were programmed by flipping a series of switches. They were developed almost as curiosities. Even as they became more advanced and complex, all effort was focused on developing greater sophistication and capabilities; no one thought they would have security issues. We only have to look at some of the early computers, such as the Berkeley Enterprises Geniac, the Heathkit EC-1, or the MITS Altair 8800, to understand why security was not an issue back then [2]. The development of computers was focused on what they could do, not how they could be attacked.

As computers began to be interconnected, the driving force was providing the ability to share information, certainly not to protect it. Initially the Internet was designed for military applications, but eventually it migrated to colleges and universities, the principal tenet of which is the sharing of knowledge.

Current Trend Is to Share, Not Protect

Even now, despite the stories of compromised data, people still want to share their data with everyone. And Web-based applications are making this easier to do than simply attaching a file to an email. Social networking sites such as SixApart provide the ability to share material: "Send messages, files, links, and events to your friends. Create a network of friends and share stuff. It's free and easy . . ." [3] In addition, many online data storage sites such as DropSend [4] and FilesAnywhere [5] provide the ability to share files. Although currently in the beta state of development, Swivel [6] provides the ability to upload data sets for analysis and comparison. These sites can allow proprietary data to leave an organization by bypassing security mechanisms.

Data Accessible from Anywhere

As though employees' desire to share data is not enough of a threat to proprietary information, many business professionals want access to data from anywhere they work, on a variety of devices. To be productive, employees now request access to data and contact information on their laptops, desktops, home computers, and mobile devices. Therefore, information technology (IT) departments must now provide the ability to sync data with numerous devices. And if the IT department can't or won't provide this capability, employees now have the power to take matters into their own hands.

Previously mentioned online storage sites can be accessed from both the home and office or anywhere there is an Internet connection. Though it might be possible to block access to some of these sites, it is not possible to block access to them all. And some can appear rather innocuous. For many, Google's free email service Gmail is a great tool that provides a very robust service for free. What few people realize is that Gmail provides more than 7 GB of storage that can also be used to store files, not just email. The Gspace plug-in [7] for the Firefox browser provides an FTP-like interface within Firefox that gives users the ability to transfer files from a computer to their Gmail accounts. This ability to easily transfer data outside the control of a company makes securing an organization's data that much more difficult.

Security Isn't About Hardware and Software

Many businesses believe that if they purchase enough equipment, they can create a secure infrastructure. Firewalls, intrusion detection systems, antivirus programs, and two-factor authentication products are just some of the tools available to assist in protecting a network and its data. It is important to keep in mind that no product or combination of products will create a secure organization by itself. Security is a process; there is no tool that you can "set and forget." All security products are only as secure as the people who configure and maintain them. The purchasing and implementation of security products should be only a percentage of the security budget. The employees tasked with maintaining the security devices should be provided with enough time, training, and equipment to properly support the products. Unfortunately, in many organizations security activities take a back seat to support activities. Highly skilled security professionals are often tasked with help-desk projects such as resetting forgotten passwords, fixing jammed printers, and setting up new employee workstations.

The Bad Guys Are Very Sophisticated

At one time the computer hacker was portrayed as a lone teenager with poor social skills who would break into systems, often for nothing more than bragging rights. As ecommerce has evolved, however, so has the profile of the hacker.

Now that there are vast collections of credit-card numbers and intellectual property that can be harvested, organized hacker groups have been formed to operate as businesses. A document released in 2008 spells it out clearly: "Cybercrime companies that work much like real-world companies are starting to appear and are steadily growing, thanks to the profits they turn. Forget individual hackers or groups of hackers with common goals. Hierarchical cybercrime organizations where each cybercriminal has his or her own role and reward system are what you and your company should be worried about." [8]

Now that organizations are being attacked by highly motivated and skilled groups of hackers, creating a secure infrastructure is mandatory.

Management Sees Security as a Drain on the Bottom Line

For most organizations, the cost of creating a strong security posture is seen as a necessary evil, similar to purchasing insurance. Organizations don't want to spend the money on it, but the risks of not making the purchase outweigh the costs. Because of this attitude, it is extremely challenging to create a secure organization. The attitude is enforced because requests for security tools are often supported by documents providing the average cost of a security incident instead of showing more concrete benefits of a strong security posture. The problem is exacerbated by the fact that IT professionals speak a different language than management. IT professionals are generally focused on technology, period. Management is focused on revenue. Concepts such as profitability, asset depreciation, return on investment, realization, and total cost of ownership are the mainstays of management. These are alien concepts to most IT professionals.

Realistically speaking, though it would be helpful if management would take steps to learn some fundamentals of information technology, IT professionals should take the initiative and learn some fundamental business concepts. Learning these concepts is beneficial to the organization because the technical infrastructure can be implemented in a cost-effective manner, and they are beneficial from a career development perspective for IT professionals.

A Google search on "business skills for IT professionals" will identify numerous educational programs that might prove helpful. For those who do not have the time or the inclination to attend a class, some very useful materials can be found online. One such document provided by the Government Chief Information Office of New South Wales is *A Guide for Government Agencies Calculating Return on Security Investment* [9]. Though extremely technical, another often cited document is *Cost-Benefit Analysis for Network Intrusion Detection Systems,* by Huaqiang Wei, Deb Frinke, Olivia Carter, and Chris Ritter [10].

Regardless of the approach that is taken, it is important to remember that any tangible cost savings or revenue generation should be utilized when requesting new security products, tools, or policies. Security professionals often overlook the value of keeping Web portals open for employees. A database that is used by a sales staff to enter contracts or purchases or check inventory will help generate more revenue if it has no downtime. A database that is not accessible or has been hacked is useless for generating revenue.

Strong security can be used to gain a competitive advantage in the marketplace. Having secured systems that are accessible 24 hours a day, seven days a week means that an organization can reach and communicate with its clients and prospective clients more efficiently. An organization that becomes recognized as a good custodian of client records and information can incorporate its security record as part of its branding. This is no different than a car company being recognized for its safety record. In discussions of cars and safety, for example, Volvo is always the first manufacturer mentioned [11].

What must be avoided is the "sky is falling" mentality. There are indeed numerous threats to a network, but we need to be realistic in allocating resources to protect against these threats. As of this writing, the National Vulnerability Database sponsored by the National Institute of Standards and Technology (NIST) lists 33,428 common vulnerabilities and exposures and publishes 18 new vulnerabilities per day [12]. In addition, the media is filled with stories of stolen laptops, credit-card numbers, and identities. The volume of threats to a network can be mind numbing. It is important to approach management with "probable threats" as opposed to "describable threats." Probable threats are those that are most likely to have an impact on your business and the ones most likely to get the attention of management.

Perhaps the best approach is to recognize that management, including the board of directors, is required to exhibit a duty of care in protecting their assets that is comparable to other organizations in their industry. When a security breach or incident occurs, being able to demonstrate the high level of security within the organization can significantly reduce exposure to lawsuits, fines, and bad press.

The goal of any discussion with management is to convince them that in the highly technical and interconnected world we live in, having a secure network and infrastructure is a "nonnegotiable requirement of doing business."[13] An excellent resource for both IT professionals and executives that can provide insight into these issues is CERT's technical report, *Governing for Enterprise Security* [14].

2. Ten Steps to Building a Secure Organization

Having identified some of the challenges to building a secure organization, let's now look at 10 ways to successfully build a secure organization. The following steps will put a business in a robust security posture.

A. Evaluate the Risks and Threats

In attempting to build a secure organization, where should you start? One commonly held belief is that you should initially identify your assets and allocate security resources based on the value of each asset. Though this approach might prove effective, it can lead to some significant vulnerabilities. An infrastructure asset might not hold a high value, for example, but it should be protected with the same effort as a high-value asset. If not, it could be an entry point into your network and provide access to valuable data.

Another approach is to begin by evaluating the threats posed to your organization and your data.

Threats Based on the Infrastructure Model

The first place to start is to identify risks based on an organization's infrastructure model. What infrastructure is in place that is necessary to support the operational needs of the business? A small business that operates out of one office has reduced risks as opposed to an organization that operates out of numerous facilities, includes a mobile workforce utilizing a variety of handheld devices, and offers products or services through a Web-based interface. An organization that has a large number of telecommuters must take steps to protect its proprietary information that could potentially reside on personally owned computers outside company control. An organization that has widely dispersed and disparate systems will have more risk potential than a centrally located one that utilizes uniform systems.

Threats Based on the Business Itself

Are there any specific threats for your particular business? Have high-level executives been accused of inappropriate activities whereby stockholders or employees would have incentive to attack the business? Are there any individuals who have a vendetta against the company for real or imagined slights or accidents? Does the community have a history of antagonism against the organization? A risk management or security team should be asking these questions on a regular basis to evaluate the risks in real time. This part of the security process is often overlooked due to the focus on daily workload.

Threats Based on Industry

Businesses belonging to particular industries are targeted more frequently and with more dedication than those in other industries. Financial institutions and online retailers are targeted because "that's where the money is." Pharmaceutical manufacturers could be targeted to steal intellectual property, but they also could be targeted by special interest groups, such as those that do not believe in testing drugs on live animals.

Identifying some of these threats requires active involvement in industry-specific trade groups in which businesses share information regarding recent attacks or threats they have identified.

Global Threats

Businesses are often so narrowly focused on their local sphere of influence that they forget that by having a network connected to the Internet, they are now connected to the rest of the world. If a piece of malware identified on the other side of the globe targets the identical software used in your organization, you can be sure that you will eventually be impacted by this malware. Additionally, if extremist groups in other countries are targeting your specific industry, you will also be targeted.

Once threats and risks are identified, you can take one of four steps:

- *Ignore the risk.* This is never an acceptable response. This is simply burying your head in the sand and hoping the problem will go away—the business equivalent of not wearing a helmet when riding a motorcycle.
- *Accept the risk.* When the cost to remove the risk is greater than the risk itself, an organization will often decide to simply accept the risk. This is a viable option as long as the organization has spent the time required to evaluate the risk.
- *Transfer the risk.* Organizations with limited staff or other resources could decide to transfer the risk. One method of transferring the risk is to purchase specialized insurance targeted at a specific risk.
- *Mitigate the risk.* Most organizations mitigate risk by applying the appropriate resources to minimize the risks posed to their network.

For organizations that would like to identify and quantify the risks to their network and information assets, CERT provides a free suite of tools to assist with the project. Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) provides risk-based assessment for security assessments and planning [15]. There are three versions of OCTAVE: the original OCTAVE, designed for large organizations (more than 300 employees); OCTAVE-S (100 people or fewer); and OCTAVE-Allegro, which is a streamlined version of the tools and is focused specifically on information assets.

Another risk assessment tool that might prove helpful is the Risk Management Framework developed by Educause/Internet 2 [16]. Targeted at institutions of higher learning, the approach could be applied to other industries.

Tracking specific threats to specific operating systems, products, and applications can be time consuming. Visiting the National Vulnerability Database and manually searching for specific issues would not necessarily be an effective use of time. Fortunately, the Center for Education and Research in Information Assurance and Security (CERIAS) at Purdue University has a tool called Cassandra that can be configured to notify you of specific threats to your particular products and applications [17].

B. Beware of Common Misconceptions

In addressing the security needs of an organization, it is common for professionals to succumb to some very common misconceptions. Perhaps the most common misconception is that the business is obscure, unsophisticated, or boring—simply not a target for malicious activity. Businesses must understand that any network that is connected to the Internet is a potential target, regardless of the type of business.

Attackers will attempt to gain access to a network and its systems for several reasons. The first is to look around to see what they can find. Regardless of the type of business, personnel information will more than likely be stored on one of the systems. This includes Social Security numbers and other personal information. This type of information is a target—always.

Another possibility is that the attacker will modify the information he or she finds or simply reconfigure the systems to behave abnormally. This type of attacker is not interested in financial gain; he is simply the technology version of teenagers who soap windows, egg cars, and cover property with toilet paper. He attacks because he finds it entertaining to do so. Additionally, these attackers could use the systems to store stolen "property" such as child pornography or credit-card numbers. If a system is not secure, attackers can store these types of materials on your system and gain access to them at their leisure.

The final possibility is that an attacker will use the hacked systems to mount attacks on other unprotected networks and systems. Computers can be used to mount denial-of-service (DoS) attacks, relay spam, or spread malicious software. To put it simply, no computer or network is immune from attack.

Another common misconception is that an organization is immune from problems caused by employees, essentially saying, "We trust all our employees, so we don't have to focus our energies on protecting our assets from them." Though this is common for small businesses in which the owners know everyone, it also occurs in larger organizations where companies believe that they only hire "professionals." It is important to remember that no matter how well job candidates present themselves, a business can never know everything about an employee's past. For this reason it is important for businesses to conduct preemployment background checks of all employees. Furthermore, it is important to conduct these background checks properly and completely.

Many employers trust this task to an online solution that promises to conduct a complete background check on an individual for a minimal fee. Many of these sites play on individuals' lack of understanding of how some of these online databases are generated. These sites might not have access to the records of all jurisdictions, since many jurisdictions either do not make their records available online or do not provide them to these databases. In addition, many of the records are entered by minimum wage data-entry clerks whose accuracy is not always 100 percent.

Background checks should be conducted by organizations that have the resources at their disposal to get court records directly from the courthouses where the records are generated and stored. Some firms have a team of "runners" who visit the courthouses daily to pull records; others have a network of contacts who can visit the courts for them. Look for organizations that are active members of the National Association of Professional Background Screeners [18]. Members of this organization are committed to providing accurate and professional results. And perhaps more important, they can provide counseling regarding the proper approach to take as well as interpreting the results of a background check.

If your organization does not conduct background checks, there are several firms that might be of assistance: Accurate Background, Inc., of Lake Forest, California [19]; Credential Check, Inc., of Troy, Michigan [20]; and Validity Screening Solutions in Overland Park, Kansas [21]. The Web sites of these companies all provide informational resources to guide you in the process. (*Note:* For businesses outside the United States or for U.S. businesses with locations overseas, the process might be more difficult because privacy laws could prevent conducting a complete background check. The firms we've mentioned should be able to provide guidance regarding international privacy laws.)

Another misconception is that a preemployment background check is all that is needed. Some erroneously believe that once a person is employed, he or she is "safe" and can no longer pose a threat. However, people's lives and fortunes can change during the course of employment. Financial pressures can cause otherwise law-abiding citizens to take risks they never would have thought possible. Drug and alcohol dependency can alter people's behavior as well. For these and other reasons it is a good idea to do an additional background check when an employee is promoted to a position of higher responsibility and trust. If this new position involves handling financial responsibilities, the background check should also include a credit check.

Though these steps might sound intrusive, which is sometimes a reason cited not to conduct these types of checks, they can also be very beneficial to the employee as well as the employer. If a problem is identified during the check, the employer can often offer assistance to help the employee get through a tough time. Financial counseling and substance abuse counseling can often turn a potentially problematic employee into a very loyal and dedicated one.

Yet another common misconception involves information technology professionals. Many businesses pay their IT staff fairly high salaries because they understand that having a properly functioning technical infrastructure is important for the continued success of the company. Since the staff is adept at setting up and maintaining systems and networks, there is a general assumption that they know everything there is to know about computers. It is important to recognize that although an individual might be very knowledgeable and technologically sophisticated, no one knows *everything* about computers. Because management does not understand technology, they are not in a very good position to judge a person's depth of knowledge and experience in the field. Decisions are often based on the certifications a person has achieved during his or her career. Though certifications can be used to determine a person's level of competency, too much weight is given to them. Many certifications require nothing more than some time and dedication to study and pass a certification test. Some training companies also offer boot camps that guarantee a person will pass the certification test. It is possible for people to become certified without having any real-world experience with the operating systems, applications, or hardware addressed by the certification. When judging a person's competency, look at his or her experience level and background first, and if the person has achieved certifications in addition to having significant real-world experience, the certification is probably a reflection of the employee's true capabilities.

The IT staff does a great deal to perpetuate the image that they know everything about computers. One of the reasons people get involved with the IT field in the first place is because they have an opportunity to try new things and overcome new challenges. This is why when an IT professional is asked if she knows how to do something, she will always respond "Yes." But in reality the real answer should be, "No, but I'll figure it out." Though they frequently can figure things out, when it comes to security we must keep in mind that it is a specialized area, and implementing a strong security posture requires significant training and experience.

C. Provide Security Training for IT Staff—Now and Forever

Just as implementing a robust, secure environment is a dynamic process, creating a highly skilled staff of security professionals is also a dynamic process. It is important to keep in mind that even though an organization's technical infrastructure might not change that frequently, new vulnerabilities are being discovered and new attacks are being launched on a regular basis. In addition, very few organizations have a stagnant infrastructure; employees are constantly requesting new software, and more technologies are added in an effort to improve efficiencies. Each new addition likely adds additional security vulnerabilities.

It is important for the IT staff to be prepared to identify and respond to new threats and vulnerabilities. It is recommended that those interested in gaining a deep security understanding start with a vendor-neutral program. A vendor-neutral program is one that focuses on concepts rather than specific products. The SANS (SysAdmin, Audit, Network, Security) Institute offers two introductory programs: Intro to Information Security (Security 301) [22], a five-day class designed for people just starting out in the security field, and the SANS Security Essentials Bootcamp (Security 401) [23], a six-day class designed for people with some security experience. Each class is also available as a self-study program, and each can be used to prepare for a specific certification.

Another option is start with a program that follows the CompTia Security + certification requirements, such as the Global Knowledge Essentials of Information Security [24]. Some colleges offer similar programs.

Once a person has a good fundamental background in security, he should then undergo vendor-specific training to apply the concepts learned to specific applications and security devices.

A great resource for keeping up with current trends in security is to become actively involved in a security-related trade organization. The key concept here is *actively involved*. Many professionals join organizations so that they can add an item to the “professional affiliations” section of their résumé. Becoming actively involved means attending meetings on a regular basis and serving on a committee or in a position on the executive board. Though this seems like a daunting time commitment, the benefit is that the professional develops a network of resources that can be available to provide insight, serve as a sounding board, or provide assistance when a problem arises. Participating in these associations is a very cost-effective way to get up to speed with current security trends and issues. Here are some organizations [25] that can prove helpful:

- ASIS International, the largest security-related organization in the world, focuses primarily on physical security but has more recently started addressing computer security as well.
- ISACA, formerly the Information Systems Audit and Control Association.
- High Technology Crime Investigation Association (HTCIA).
- Information Systems Security Association (ISSA).
- InfraGard, a joint public and private organization sponsored by the Federal Bureau of Investigation (FBI).

In addition to monthly meetings, many local chapters of these organizations sponsor regional conferences that are usually very reasonably priced and attract nationally recognized experts.

Arguably one of the best ways to determine whether an employee has a strong grasp of information security concepts is if she can achieve the Certified Information Systems Security Professional (CISSP) certification. Candidates for this certification are tested on their understanding of the following 10 knowledge domains:

- Access control
- Application security
- Business continuity and disaster recovery planning
- Cryptography
- Information security and risk management
- Legal, regulations, compliance, and investigations
- Operations security
- Physical (environmental) security

- Security architecture and design
- Telecommunications and network security

What makes this certification so valuable is that the candidate must have a minimum of five years of professional experience in the information security field or four years of experience and a college degree. To maintain certification, a certified individual is required to attend 120 hours of continuing professional education during the three-year certification cycle. This ensures that those holding the CISSP credential are staying up to date with current trends in security. The CISSP certification is maintained by (ISC)^2.[26]

D. Think "Outside the Box"

For most businesses, the threat to their intellectual assets and technical infrastructure comes from the "bad guys" sitting outside their organizations, trying to break in. These organizations establish strong perimeter defenses, essentially "boxing in" their assets. However, internal employees have access to proprietary information to do their jobs, and they often disseminate this information to areas where it is no longer under the control of the employer. This dissemination of data is generally not performed with any malicious intent, simply for employees to have access to data so that they can perform their job responsibilities more efficiently. This also becomes a problem when an employee leaves (or when a person still-employed loses something like a laptop with proprietary information stored on it) and the organization and takes no steps to collect or control their proprietary information in the possession of their now ex-employee.

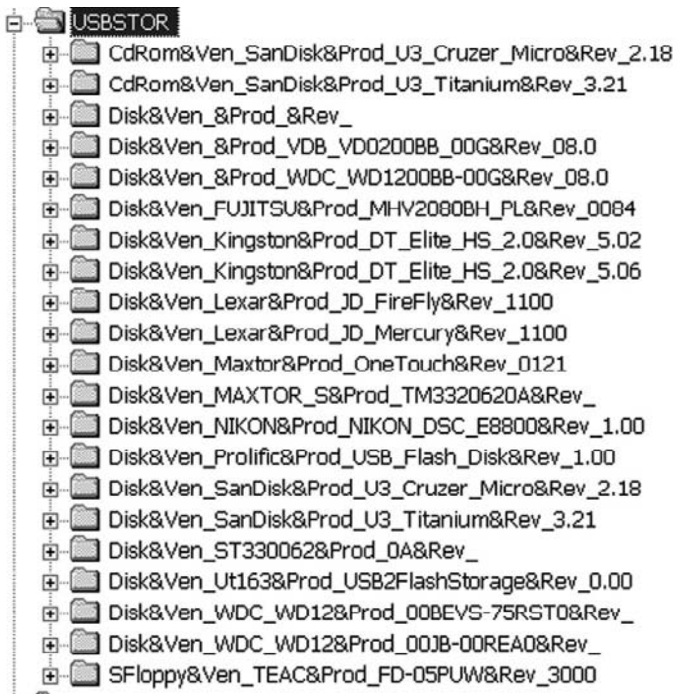
One of the most overlooked threats to intellectual property is the innocuous and now ubiquitous USB Flash drive. These devices, the size of a tube of lipstick, are the modern-day floppy disk in terms of portable data storage. They are a very convenient way to transfer data between computers. But the difference between these devices and a floppy disk is that USB Flash drives can store a very large amount of data. A 16 GB USB Flash drive has the same storage capacity as more than 10,000 floppy disks! As of this writing, a 16 GB USB Flash drive can be purchased for as little as $30. Businesses should keep in mind that as time goes by, the capacity of these devices will increase and the price will decrease, making them very attractive to employees.

These devices are not the only threat to data. Because other devices can be connected to the computer through the USB port, digital cameras, MP3 players, and external hard drives can now be used to remove data from a computer and the network to which it is connected. Most people would recognize that external hard drives pose a threat, but they would not recognize other devices as a threat. Cameras and music players are designed to store images and music, but to a computer they are simply additional mass storage devices. It is difficult for people to understand that an iPod can carry word processing documents, databases, and spreadsheets as well as music.

Fortunately, Microsoft Windows tracks the devices that are connected to a system in a Registry key, HKEY_Local_Machine\System\ControlSet00x\Enum\USBStor. It might prove interesting to look in this key on your own computer to see what types of devices have been connected. Figure 1.1 shows a wide array of devices that have been connected to a system that includes USB Flash drives, a digital camera, and several external hard drives.

Windows Vista has an additional key that tracks connected devices: HKEY_Local_Machine\Software\Microsoft\Windows Portable Devices\Devices [27]. (*Note:* Analyzing the Registry is a great way to investigate the activities of computer users. For many, however, the Registry is tough to navigate and interpret. If you are interested in understanding more about the Registry, you might want to download and play with Harlan Carvey's RegRipper [28].)

Another threat to information that carries data outside the walls of the organization is the plethora of handheld devices currently in use. Many of these devices have the ability to send and receive email as well as create, store, and transmit word processing, spreadsheet, and PDF files. Though most employers will not purchase these devices for their employees, they are more than happy to allow their employees to sync their personally owned devices with their corporate computers. Client contact information, business plans, and other materials can easily be copied from a system. Some businesses feel that they have this threat under control because they provide their employees with corporate-owned devices and they can collect these devices when employees leave their employment. The only problem with this attitude is



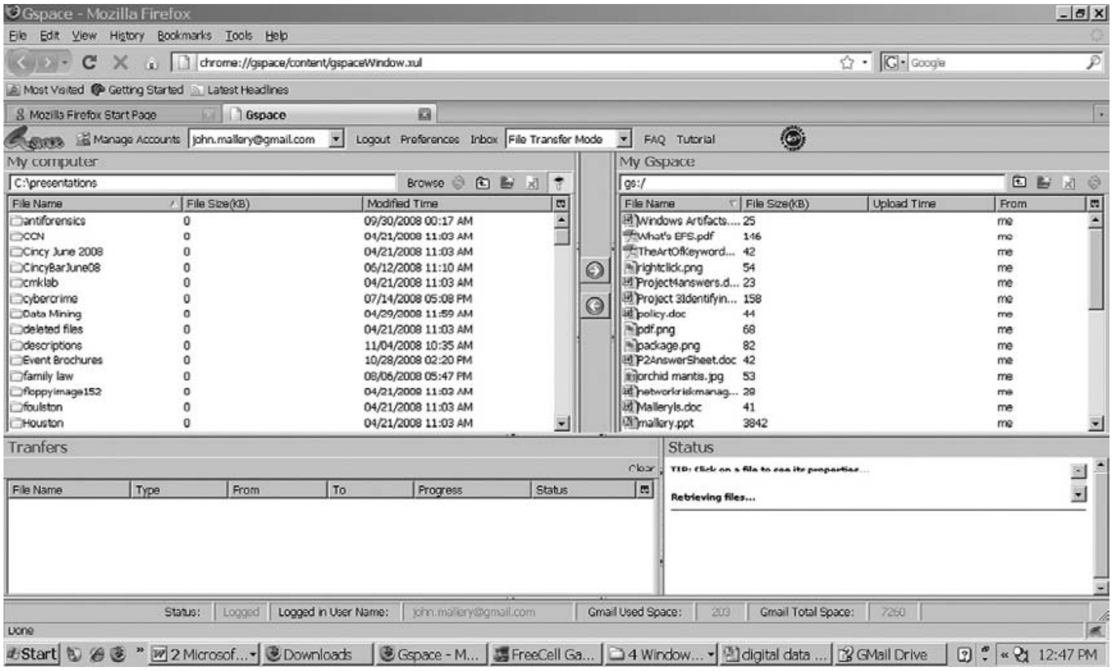
```
USBSTOR
  CdRom&Ven_SanDisk&Prod_U3_Cruzer_Micro&Rev_2.18
  CdRom&Ven_SanDisk&Prod_U3_Titanium&Rev_3.21
  Disk&Ven_&Prod_&Rev_
  Disk&Ven_&Prod_VDB_VD0200BB_00G&Rev_08.0
  Disk&Ven_&Prod_WDC_WD1200BB-00G&Rev_08.0
  Disk&Ven_FUJITSU&Prod_MHV2080BH_PL&Rev_0084
  Disk&Ven_Kingston&Prod_DT_Elite_HS_2.0&Rev_5.02
  Disk&Ven_Kingston&Prod_DT_Elite_HS_2.0&Rev_5.06
  Disk&Ven_Lexar&Prod_JD_FireFly&Rev_1100
  Disk&Ven_Lexar&Prod_JD_Mercury&Rev_1100
  Disk&Ven_Maxtor&Prod_OneTouch&Rev_0121
  Disk&Ven_MAXTOR_S&Prod_TM3320620A&Rev_
  Disk&Ven_NIKON&Prod_NIKON_DSC_E8800&Rev_1.00
  Disk&Ven_Prolific&Prod_USB_Flash_Disk&Rev_1.00
  Disk&Ven_SanDisk&Prod_U3_Cruzer_Micro&Rev_2.18
  Disk&Ven_SanDisk&Prod_U3_Titanium&Rev_3.21
  Disk&Ven_ST330062&Prod_0A&Rev_
  Disk&Ven_Ut163&Prod_USB2FlashStorage&Rev_0.00
  Disk&Ven_WDC_WD12&Prod_00BEVS-75RST0&Rev_
  Disk&Ven_WDC_WD12&Prod_00JB-00REA0&Rev_
  SFloppy&Ven_TEAC&Prod_FD-05PUW&Rev_3000
```

Figure 1.1: Identifying connected USB devices in the USBStor Registry key.

that employees can easily copy data from the devices to their home computers before the devices are returned.

Because of the threat of portable data storage devices and handheld devices, it is important for an organization to establish policies outlining the acceptable use of these devices as well as implementing an enterprise-grade solution to control how, when, or if data can be copied to them. Filling all USB ports with epoxy is an inexpensive solution, but it is not really effective. Fortunately there are several products that can protect against this type of data leak. DeviceWall from Centennial Software [29] and Mobile Security Enterprise Edition from Bluefire Security Technologies [30] are two popular ones.

Another way that data leaves control of an organization is through the use of online data storage sites. These sites provide the ability to transfer data from a computer to an Internet-accessible location. Many of these sites provide 5 GB or more of free storage. Though it is certainly possible to blacklist these sites, there are so many, and more are being developed on a regular basis, that it is difficult if not impossible to block access to all of them. One such popular storage location is the storage space provided with a Gmail account. Gmail provides a large amount of storage space with its free accounts (7260 MB as of this writing, and growing). To access this storage space, users must use the Firefox browser with the Gspace plugin installed [31]. Once logged in, users can transfer files simply by highlighting the file and clicking an arrow. Figure 1.2 shows the Gspace interface.



chrome://gspace/content/gspaceWindow.xul

john.mallery@gmail.com — File Transfer Mode

My computer

C:\presentations

File Name	File Size(KB)	Modified Time
antiforensics	0	09/30/2008 00:17 AM
CCN	0	04/21/2008 11:03 AM
Cincy June 2008	0	04/21/2008 11:03 AM
CincyBarJune08	0	06/12/2008 11:10 AM
cmklab	0	04/21/2008 11:03 AM
cybercrime	0	07/14/2008 05:08 PM
Data Mining	0	04/29/2008 11:59 AM
deleted files	0	04/21/2008 11:03 AM
descriptions	0	11/04/2008 10:35 AM
Event Brochures	0	10/28/2008 02:20 PM
family law	0	08/06/2008 05:47 PM
floppyimage152	0	04/21/2008 11:03 AM
foulston	0	04/21/2008 11:03 AM
Houston	0	04/21/2008 11:03 AM

My Gspace

gs:/

File Name	File Size(KB)	Upload Time	From
Windows Artifacts....	25		me
What's EFS.pdf	146		me
TheArtOfKeyword...	42		me
rightclick.png	54		me
Project4answers.d...	23		me
Project 3Identifyin...	158		me
policy.doc	44		me
pdf.png	68		me
package.png	82		me
P2AnswerSheet.doc	42		me
orchid mantis.jpg	53		me
networkriskmanag...	28		me
Malleryls.doc	41		me
mallery.ppt	3842		me

Tranfers

File Name	Type	From	To	Progress	Status

Status

TIP: Click on a file to see its properties...

Retrieving files...

Status: Logged | Logged in User Name: john.mallery@gmail.com | Gmail Used Space: 203 | Gmail Total Space: 7260

Figure 1.2: Accessing Gspace using the Firefox browser.

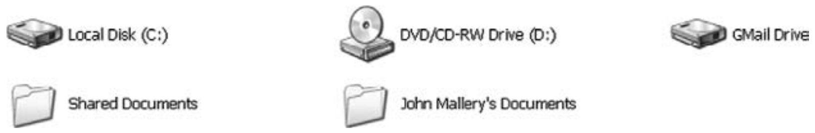


Figure 1.3: Gmail Drive in Windows Explorer.

Another tool that will allow users to access the storage space in their Gmail account is the Gmail Drive shell extension [32]. This shell extension places a drive icon in Windows Explorer, allowing users to copy files to the online storage location as though it were a normal mapped drive. Figure 1.3 shows the Gmail Drive icon in Windows Explorer.

Apple has a similar capability for those users with a MobileMe account. This drive is called iDisk and appears in the Finder. People who utilize iDisk can access the files from anywhere using a Web browser, but they can also upload files using the browser. Once uploaded, the files are available right on the user's desktop, and they can be accessed like any other file. Figures 1.4 and 1.5 show iDisk features.

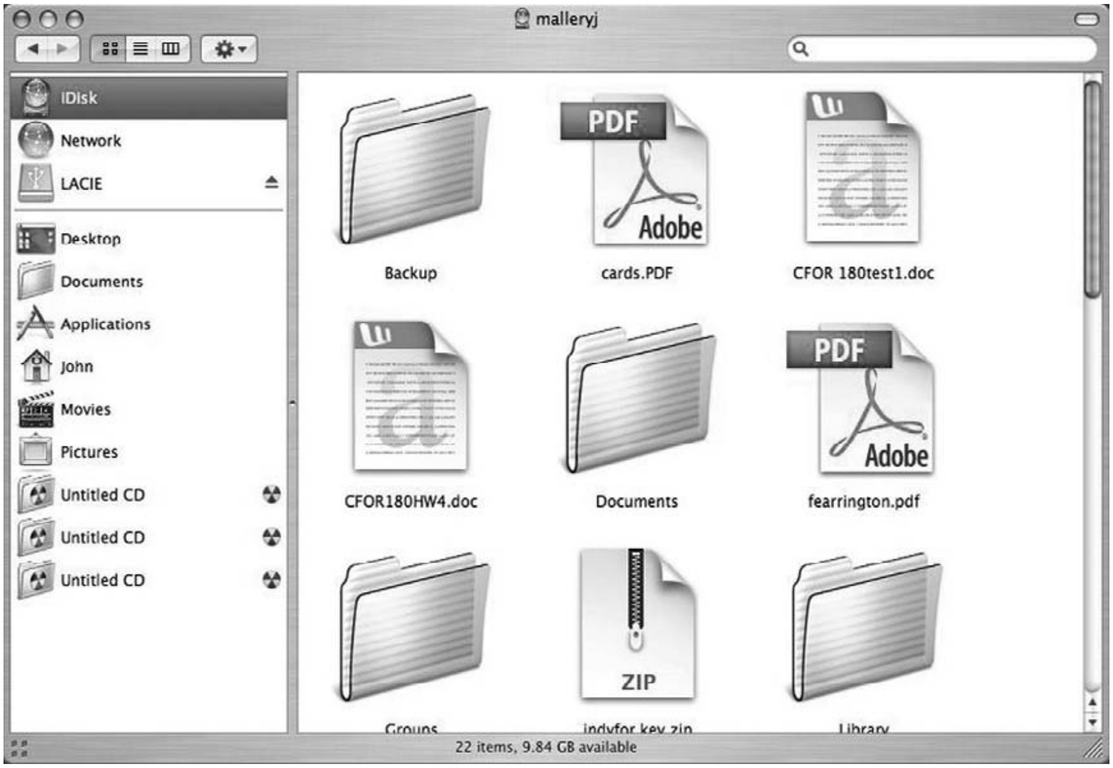


Figure 1.4: Accessing files in iDisk.

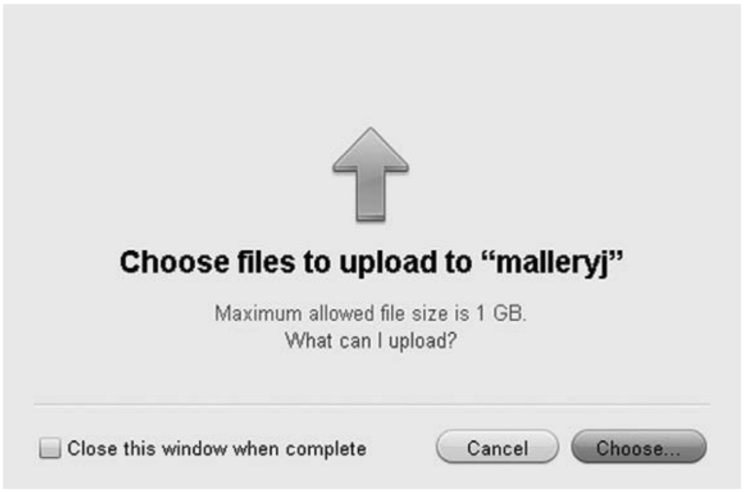

Choose files to upload to "malleryj"
Maximum allowed file size is 1 GB.
What can I upload?
Close this window when complete
Cancel
Choose...

Figure 1.5: iDisk Upload window in Firefox.

In addition, numerous sites provide online storage. A partial list is included here:

- ElephantDrive: www.elephantdrive.com
- Mozy: www.mozy.com
- Box: www.box.net
- Carbonite: www.carbonite.com
- Windows Live SkyDrive: www.skydrive.live.com
- FilesAnywhere: www.filesanywhere.com
- Savefile: www.savefile.com
- Spare Backup: www.sparebackup.com
- Digitalbucket.net: www.digitalbucket.net
- Memeo: www.memeo.com
- Biscu.com: www.biscu.com

Note: Though individuals might find these sites convenient and easy to use for file storage and backup purposes, businesses should think twice about storing data on them. The longevity of these sites is not guaranteed. For example, Xdrive, a popular online storage service created in 1999 and purchased by AOL in 2005 (allegedly for US$30 million), shut down on January 12, 2009.

E. Train Employees: Develop a Culture of Security

One of the greatest security assets is a business's own employees, but only if they have been properly trained to comply with security policies and to identify potential security problems.

Many employees don't understand the significance of various security policies and implementations. As mentioned previously, they consider these policies nothing more than an inconvenience. Gaining the support and allegiance of employees takes time, but it is time well spent. Begin by carefully explaining the reasons behind any security implementation. One of the reasons could be ensuring employee productivity, but focus primarily on the security issues. File sharing using LimeWire and eMule might keep employees away from work, but they can also open up holes in a firewall. Downloading and installing unapproved software can install malicious software that can infect user systems, causing their computers to function slowly or not at all.

Perhaps the most direct way to gain employee support is to let employees know that the money needed to respond to attacks and fix problems initiated by users is money that is then not available for raises and promotions. Letting employees know that they now have some "skin in the game" is one way to get them involved in security efforts. If a budget is set aside for responding to security problems and employees help stay well within the budget, the difference between the money spent and the actual budget could be divided among employees as a bonus. Not only would employees be more likely to speak up if they notice network or system slowdowns, they would probably be more likely to confront strangers wandering through the facility.

Another mechanism that can be used to gain security allies is to provide advice regarding the proper security mechanisms for securing home computers. Though some might not see this as directly benefiting the company, keep in mind that many employees have corporate data on their home computers. This advice can come from periodic, live presentations (offer refreshments and attendance will be higher) or from a periodic newsletter that is either mailed or emailed to employees' personal addresses.

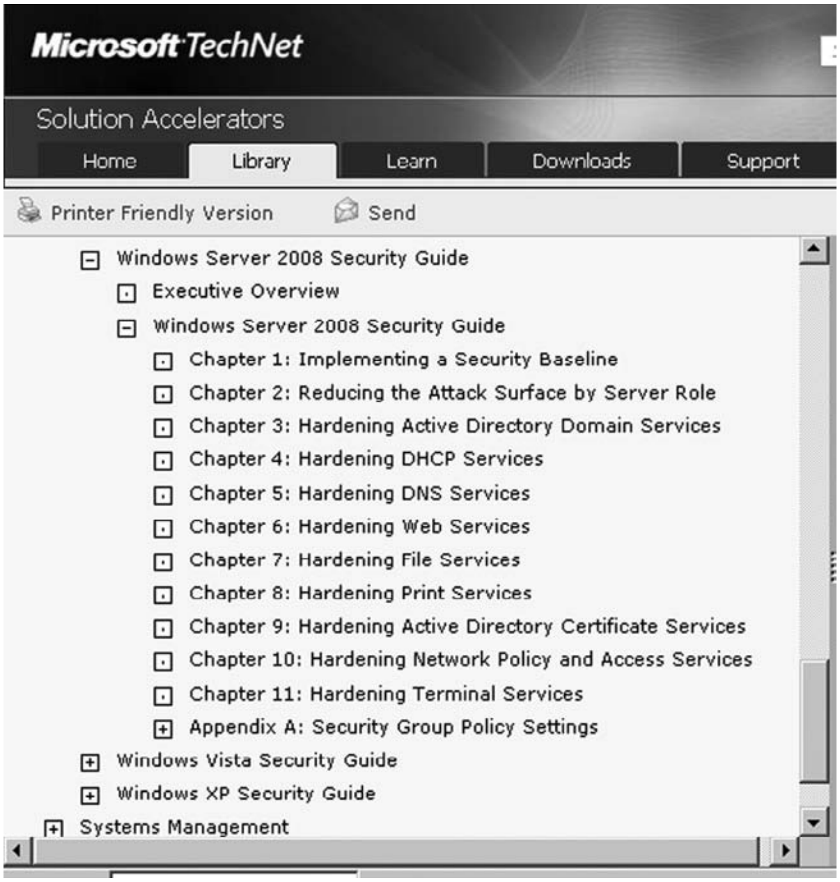
The goal of these activities is to encourage employees to approach management or the security team voluntarily. When this begins to happen on a regular basis, you will have expanded the capabilities of your security team and created a much more secure organization.

The security expert Roberta Bragg used to tell a story of one of her clients who took this concept to a high level. The client provided the company mail clerk with a WiFi hotspot detector and promised him a free steak dinner for every unauthorized wireless access point he could find on the premises. The mail clerk was very happy to have the opportunity to earn three free steak dinners.

F. Identify and Utilize Built-In Security Features of the Operating System and Applications

Many organizations and systems administrators state that they cannot create a secure organization because they have limited resources and simply do not have the funds to purchase robust security tools. This is a ridiculous approach to security because all operating systems and many applications include security mechanisms that require no organizational resources other than time to identify and configure these tools. For Microsoft Windows operating systems, a terrific resource is the online Microsoft TechNet Library [33]. Under the Solutions Accelerators link you can find security guides for all recent Microsoft Windows operating systems. Figure 1.6 shows the table of contents for Windows 2008 Server.

TechNet is a great resource and can provide insight into managing numerous security issues, from Microsoft Office 2007 to security risk management. These documents can assist in implementing the built-in security features of Microsoft Windows products. Assistance is needed in identifying many of these capabilities because they are often hidden from view and turned off by default.

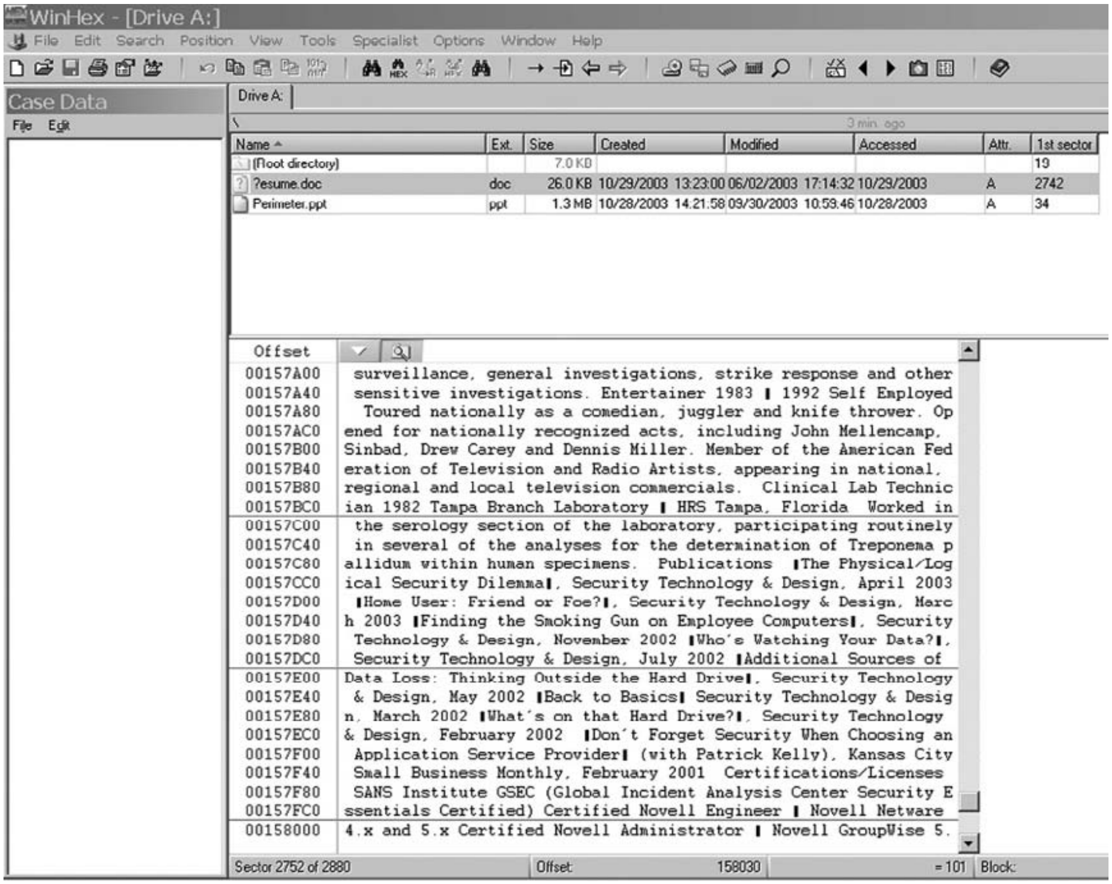


Windows Server 2008 Security Guide
Executive Overview
Windows Server 2008 Security Guide
Chapter 1: Implementing a Security Baseline
Chapter 2: Reducing the Attack Surface by Server Role
Chapter 3: Hardening Active Directory Domain Services
Chapter 4: Hardening DHCP Services
Chapter 5: Hardening DNS Services
Chapter 6: Hardening Web Services
Chapter 7: Hardening File Services
Chapter 8: Hardening Print Services
Chapter 9: Hardening Active Directory Certificate Services
Chapter 10: Hardening Network Policy and Access Services
Chapter 11: Hardening Terminal Services
Appendix A: Security Group Policy Settings
Windows Vista Security Guide
Windows XP Security Guide
Systems Management

Figure 1.6: Windows Server 2008 Security Guide Table of Contents.

One of the biggest concerns in an organization today is data leaks, which are ways that confidential information can leave an organization despite robust perimeter security. As mentioned previously, USB Flash drives are one cause of data leaks; another is the recovery of data found in the unallocated clusters of a computer's hard drive. Unallocated clusters, or *free space*, as it is commonly called, is the area of a hard drive where the operating system and applications dump their artifacts or residual data. Though this data is not viewable through a user interface, the data can easily be identified (and sometimes recovered) using a hex editor such as WinHex [34]. Figure 1.7 shows the contents of a deleted file stored on a floppy disk being displayed by WinHex.

Should a computer be stolen or donated, it is very possible that someone could access the data located in unallocated clusters. For this reason, many people struggle to find an appropriate "disk-scrubbing" utility. Many such commercial utilities exist, but there is one built into Microsoft Windows operating systems. The command-line program cipher.exe is designed to display or alter the encryption of directories (files) stored on NTFS partitions.


WinHex - [Drive A:]
File Edit Search Position View Tools Specialist Options Window Help
Case Data
File Edit
Drive A:
\
3 min. ago

Name	Ext.	Size	Created	Modified	Accessed	Attr.	1st sector
(Root directory)		7.0 KB					19
?esume.doc	doc	26.0 KB	10/29/2003 13:23:00	06/02/2003 17:14:32	10/29/2003	A	2742
Perimeter.ppt	ppt	1.3 MB	10/28/2003 14:21:58	09/30/2003 10:59:46	10/28/2003	A	34

```
Offset
00157A00   surveillance, general investigations, strike response and other
00157A40   sensitive investigations. Entertainer 1983 ▮ 1992 Self Employed
00157A80    Toured nationally as a comedian, juggler and knife thrower. Op
00157AC0  ened for nationally recognized acts, including John Mellencamp,
00157B00  Sinbad, Drew Carey and Dennis Miller. Member of the American Fed
00157B40  eration of Television and Radio Artists, appearing in national,
00157B80  regional and local television commercials.  Clinical Lab Technic
00157BC0  ian 1982 Tampa Branch Laboratory ▮ HRS Tampa, Florida  Worked in
00157C00   the serology section of the laboratory, participating routinely
00157C40   in several of the analyses for the determination of Treponema p
00157C80  allidum within human specimens.  Publications  ▮The Physical/Log
00157CC0  ical Security Dilemma▮, Security Technology & Design, April 2003
00157D00   ▮Home User: Friend or Foe?▮, Security Technology & Design, Marc
00157D40  h 2003 ▮Finding the Smoking Gun on Employee Computers▮, Security
00157D80   Technology & Design, November 2002 ▮Who's Watching Your Data?▮,
00157DC0   Security Technology & Design, July 2002 ▮Additional Sources of
00157E00  Data Loss: Thinking Outside the Hard Drive▮, Security Technology
00157E40   & Design, May 2002 ▮Back to Basics▮ Security Technology & Desig
00157E80  n, March 2002 ▮What's on that Hard Drive?▮, Security Technology
00157EC0  & Design, February 2002  ▮Don't Forget Security When Choosing an
00157F00   Application Service Provider▮ (with Patrick Kelly), Kansas City
00157F40   Small Business Monthly, February 2001  Certifications/Licenses
00157F80   SANS Institute GSEC (Global Incident Analysis Center Security E
00157FC0  ssentials Certified) Certified Novell Engineer ▮ Novell Netware
00158000  4.x and 5.x Certified Novell Administrator ▮ Novell GroupWise 5.
Sector 2752 of 2880        Offset:        158030        = 101  Block:
```

Figure 1.7: WinHex displaying the contents of a deleted Word document.

Few people even know about this command; even fewer are familiar with the /*w* switch. Here is a description of the switch from the program's Help file:

> *Removes data from available unused disk space on the entire volume. If this option is chosen, all other options are ignored. The directory specified can be anywhere in a local volume. If it is a mount point or points to a directory in another volume, the data on that volume will be removed.*

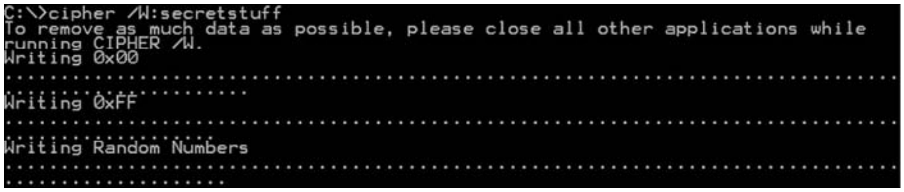
To use Cipher, click **Start Run** and type **cmd**. When the cmd.exe window opens, type **cipher /w:*folder***, where *folder* is any folder in the volume that you want to clean, and then press **Enter**. Figure 1.8 shows Cipher wiping a folder.

For more on secure file deletion issues, see the author's white paper in the SANS reading room, "Secure file deletion: Fact or fiction?"[35]

Another source of data leaks is the personal and editing information that can be associated with Microsoft Office files. In Microsoft Word 2003 you can configure the application to remove personal information on save and to warn you when you are about to print, share, or send a document containing tracked changes or comments.

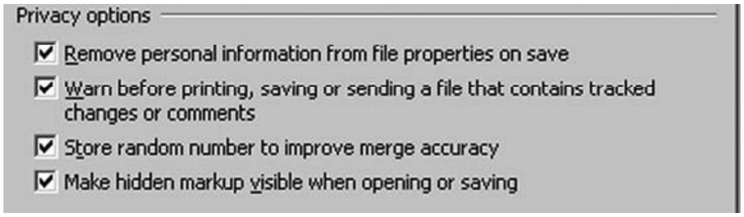
To access this feature, within Word click **Tools Options** and then click the **Security** tab. Toward the bottom of the security window you will notice the two options described previously. Simply select the options you want to use. Figure 1.9 shows these options.

Microsoft Office 2007 made this tool more robust and more accessible. A separate tool called Document Inspector can be accessed by clicking the **Microsoft Office** button, pointing to **Prepare Document**, and then clicking **Inspect Document**. Then select the items you want to remove.



```
C:\>cipher /W:secretstuff
To remove as much data as possible, please close all other applications while
running CIPHER /W.
Writing 0x00
...............................................................................
.....................
Writing 0xFF
...............................................................................
.................
Writing Random Numbers
...............................................................................
.................
```

Figure 1.8: Cipher wiping a folder called Secretstuff.



Privacy options

- [✓] Remove personal information from file properties on save
- [✓] Warn before printing, saving or sending a file that contains tracked changes or comments
- [✓] Store random number to improve merge accuracy
- [✓] Make hidden markup visible when opening or saving

Figure 1.9: Security options for Microsoft Word 2003.

Implementing a strong security posture often begins by making the login process more robust. This includes increasing the complexity of the login password. All passwords can be cracked, given enough time and resources, but the more difficult you make cracking a password, the greater the possibility the asset the password protects will stay protected.

All operating systems have some mechanism to increase the complexity of passwords. In Microsoft Windows XP Professional, this can be accomplished by clicking **Start Control Panel Administrative Tools Local Security Policy**. Under **Security Settings**, expand **Account Policies** and then highlight **Password Policy**. In the right-hand panel you can enable password complexity. Once this is enabled, passwords must contain at least three of the four following password groups [36]:

- English uppercase characters (A through Z)
- English lowercase characters (a through z)
- Numerals (0 through 9)
- Nonalphabetic characters (such as !, $, #, %)

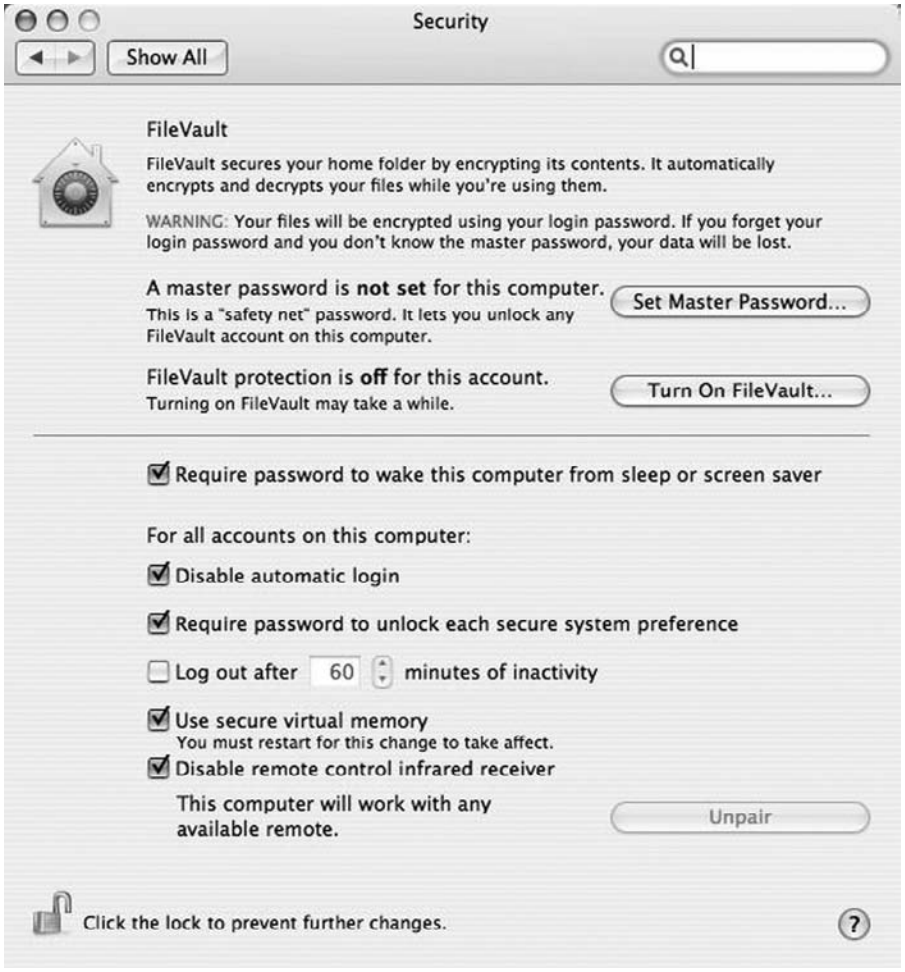
It is important to recognize that all operating systems have embedded tools to assist with security. They often require a little research to find, but the time spent in identifying them is less than the money spent on purchasing additional security products or recovering from a security breach.

Though not yet used by many corporations, Mac OS X has some very robust security features, including File Vault, which creates an encrypted home folder and the ability to encrypt virtual memory. Figure 1.10 shows the security options for Mac OS X.

G. Monitor Systems

Even with the most robust security tools in place, it is important to monitor your systems. All security products are manmade and can fail or be compromised. As with any other aspect of technology, one should never rely on simply one product or tool. Enabling logging on your systems is one way to put your organization in a position to identify problem areas. The problem is, what should be logged? There are some security standards that can help with this determination. One of these standards is the Payment Card Industry Data Security Standard (PCI DSS) [37]. Requirement 10 of the PCI DSS states that organizations must "Track and monitor access to network resources and cardholder data." If you simply substitute *confidential information* for the phrase *cardholder data,* this requirement is an excellent approach to a log management program. Requirement 10 is reproduced here:

> *Logging mechanisms and the ability to track user activities are critical. The presence of logs in all environments allows thorough tracking and analysis if something does go wrong. Determining the cause of a compromise is very difficult without system activity logs:*



FileVault

FileVault secures your home folder by encrypting its contents. It automatically encrypts and decrypts your files while you're using them.

WARNING: Your files will be encrypted using your login password. If you forget your login password and you don't know the master password, your data will be lost.

A master password is **not set** for this computer. This is a "safety net" password. It lets you unlock any FileVault account on this computer.

FileVault protection is **off** for this account. Turning on FileVault may take a while.

- [✓] Require password to wake this computer from sleep or screen saver

For all accounts on this computer:

- [✓] Disable automatic login
- [✓] Require password to unlock each secure system preference
- [ ] Log out after 60 minutes of inactivity
- [✓] Use secure virtual memory
  You must restart for this change to take affect.
- [✓] Disable remote control infrared receiver
  This computer will work with any available remote.

Click the lock to prevent further changes.

Figure 1.10: Security options for Mac OS X.

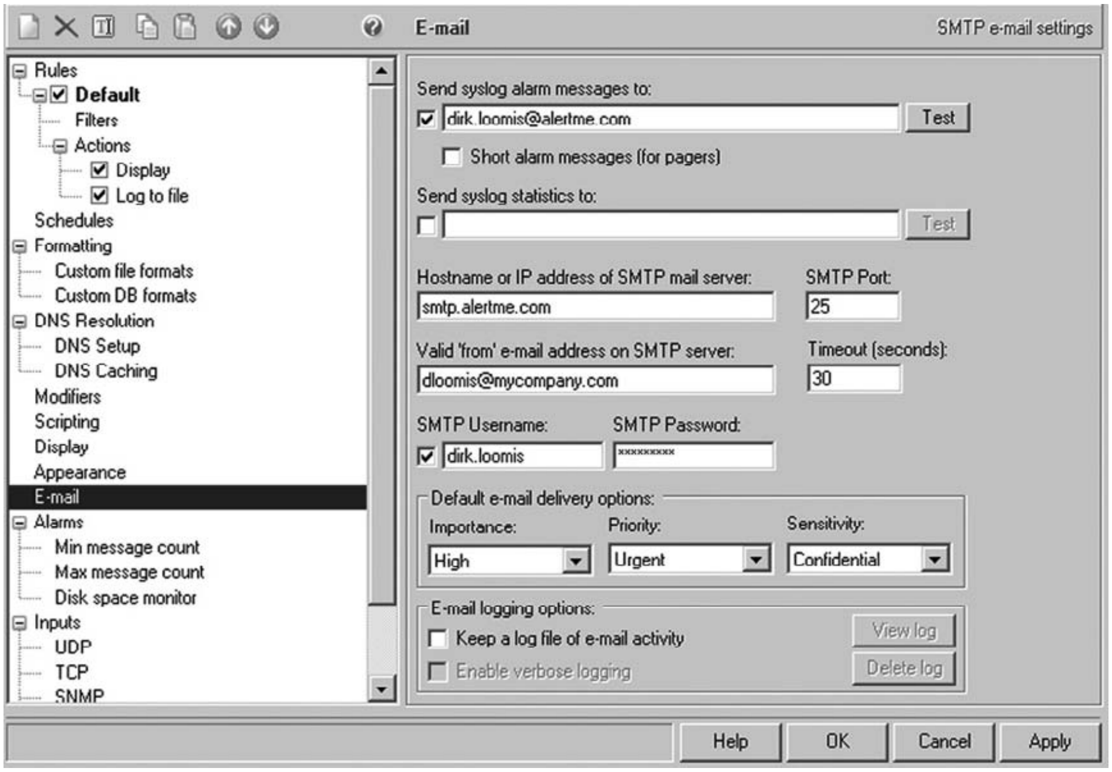
1. Establish a process for linking all access to system components (especially access done with administrative privileges such as root) to each individual user.
2. Implement automated audit trails for all system components to reconstruct the following events:
   - All individual user accesses to cardholder data
   - All actions taken by any individual with root or administrative privileges
   - Access to all audit trails
   - Invalid logical access attempts
   - Use of identification and authentication mechanisms
   - Initialization of the audit logs
   - Creation and deletion of system-level objects

3. Record at least the following audit trail entries for all system components for each event:
   - User identification
   - Type of event
   - Date and time
   - Success or failure indication
   - Origination of event
   - Identity or name of affected data, system component, or resource
4. Synchronize all critical system clocks and times.
5. Secure audit trails so they cannot be altered:
   - Limit viewing of audit trails to those with a job-related need.
   - Protect audit trail files from unauthorized modifications.
   - Promptly back up audit trail files to a centralized log server or media that is difficult to alter.
   - Copy logs for wireless networks onto a log server on the internal LAN.
   - Use file integrity monitoring and change detection software on logs to ensure that existing log data cannot be changed without generating alerts (although new data being added should not cause an alert).
6. Review logs for all system components at least daily. Log reviews must include those servers that perform security functions like intrusion detection system (IDS) and authentication, authorization, and accounting protocol (AAA) servers (e.g. RADIUS). *Note:* Log harvesting, parsing, and alerting tools may be used to achieve compliance.
7. Retain audit trail history for at least one year, with a minimum of three months online availability.

Requirement 6 looks a little overwhelming, since few organizations have the time to manually review log files. Fortunately, there are tools that will collect and parse log files from a variety of sources. All these tools have the ability to notify individuals of a particular event. One simple tool is the Kiwi Syslog Daemon [38] for Microsoft Windows. Figure 1.11 shows the configuration screen for setting up email alerts in Kiwi.

Additional log parsing tools include Microsoft's Log Parser [39] and, for Unix, Swatch [40]. Commercial tools include Cisco Security Monitoring, Analysis, and Response System (MARS) [41] and GFI EventsManager [42].

An even more detailed approach to monitoring your systems is to install a packet-capturing tool on your network so you can analyze and capture traffic in real time. One tool that can be very helpful is Wireshark, which is "an award-winning network protocol analyzer developed by an international team of networking experts." [43] Wireshark is based on the original packet capture tool, Ethereal. Analyzing network traffic is not a trivial task and requires some training, but it is the perhaps the most accurate way to determine what is happening on your network. Figure 1.12 shows Wireshark monitoring the traffic on a wireless interface.



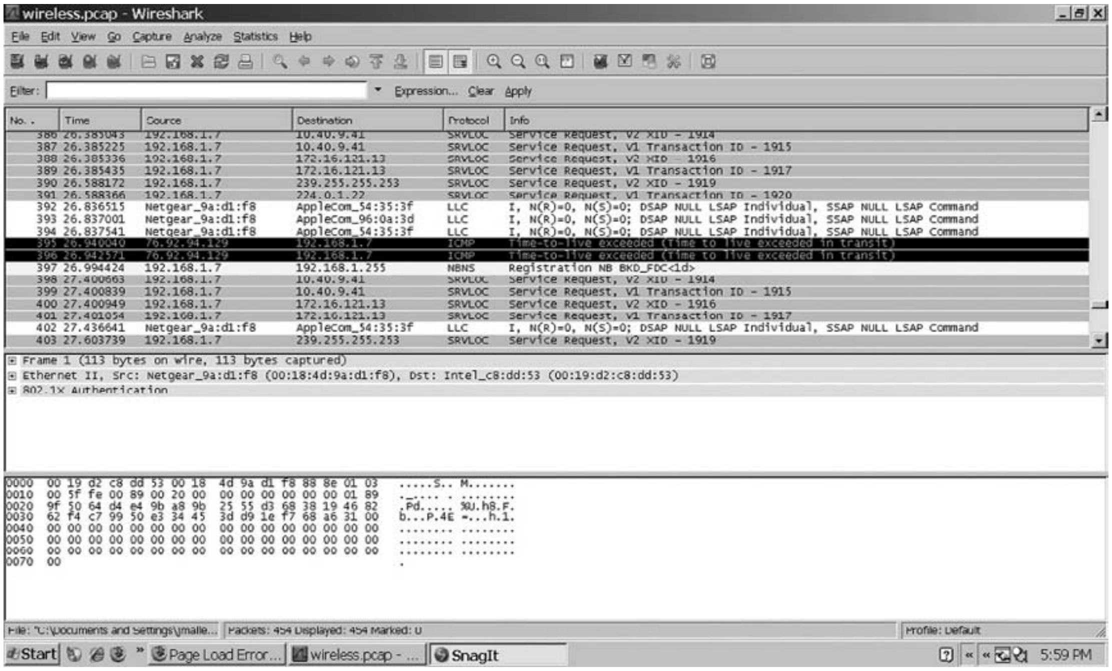
Send syslog alarm messages to: [✓] dirk.loomis@alertme.com
[ ] Short alarm messages (for pagers)
Send syslog statistics to: [ ]
Hostname or IP address of SMTP mail server: smtp.alertme.com
SMTP Port: 25
Valid 'from' e-mail address on SMTP server: dloomis@mycompany.com
Timeout (seconds): 30
SMTP Username: [✓] dirk.loomis
SMTP Password: xxxxxxxxx
Default e-mail delivery options:
Importance: High
Priority: Urgent
Sensitivity: Confidential
E-mail logging options:
[ ] Keep a log file of e-mail activity
[ ] Enable verbose logging

Figure 1.11: Kiwi Syslog Daemon Email Alert Configuration screen.

H. Hire a Third Party to Audit Security

Regardless of how talented your staff is, there is always the possibility that they overlooked something or inadvertently misconfigured a device or setting. For this reason it is very important to bring in an extra set of "eyes, ears, and hands" to review your organization's security posture.

Though some IT professionals will become paranoid having a third party review their work, intelligent staff members will recognize that a security review by outsiders can be a great learning opportunity. The advantage of having a third party review your systems is that the outsiders have experience reviewing a wide range of systems, applications, and devices in a variety of industries. They will know what works well and what might work but cause problems in the future. They are also more likely to be up to speed on new vulnerabilities and the latest product updates. Why? Because this is all they do. They are not encumbered by administrative duties, internal politics, and help desk requests. They will be more objective than in-house staff, and they will be in a position to make recommendations after their analysis.



```
No. .  Time       Source             Destination          Protocol  Info
  386 26.385043  192.168.1.7        10.40.9.41           SRVLOC    Service Request, V2 XID - 1914
  387 26.385225  192.168.1.7        10.40.9.41           SRVLOC    Service Request, V1 Transaction ID - 1915
  388 26.385336  192.168.1.7        172.16.121.13        SRVLOC    Service Request, V2 XID - 1916
  389 26.385435  192.168.1.7        172.16.121.13        SRVLOC    Service Request, V1 Transaction ID - 1917
  390 26.588172  192.168.1.7        239.255.255.253      SRVLOC    Service Request, V2 XID - 1919
  391 26.588366  192.168.1.7        224.0.1.22           SRVLOC    Service Request, V1 Transaction ID - 1920
  392 26.836515  Netgear_9a:d1:f8   AppleCom_54:35:3f    LLC       I, N(R)=0, N(S)=0; DSAP NULL LSAP Individual, SSAP NULL LSAP Command
  393 26.837001  Netgear_9a:d1:f8   AppleCom_96:0a:3d    LLC       I, N(R)=0, N(S)=0; DSAP NULL LSAP Individual, SSAP NULL LSAP Command
  394 26.837541  Netgear_9a:d1:f8   AppleCom_54:35:3f    LLC       I, N(R)=0, N(S)=0; DSAP NULL LSAP Individual, SSAP NULL LSAP Command
  395 26.940040  76.92.94.129       192.168.1.7          ICMP      Time-to-live exceeded (Time to live exceeded in transit)
  396 26.942571  76.92.94.129       192.168.1.7          ICMP      Time-to-live exceeded (Time to live exceeded in transit)
  397 26.994424  192.168.1.7        192.168.1.255        NBNS      Registration NB BKD_FDC<1d>
  398 27.400663  192.168.1.7        10.40.9.41           SRVLOC    Service Request, V2 XID - 1914
  399 27.400839  192.168.1.7        10.40.9.41           SRVLOC    Service Request, V1 Transaction ID - 1915
  400 27.400949  192.168.1.7        172.16.121.13        SRVLOC    Service Request, V2 XID - 1916
  401 27.401054  192.168.1.7        172.16.121.13        SRVLOC    Service Request, V1 Transaction ID - 1917
  402 27.436641  Netgear_9a:d1:f8   AppleCom_54:35:3f    LLC       I, N(R)=0, N(S)=0; DSAP NULL LSAP Individual, SSAP NULL LSAP Command
  403 27.603739  192.168.1.7        239.255.255.253      SRVLOC    Service Request, V2 XID - 1919

⊞ Frame 1 (113 bytes on wire, 113 bytes captured)
⊞ Ethernet II, Src: Netgear_9a:d1:f8 (00:18:4d:9a:d1:f8), Dst: Intel_c8:dd:53 (00:19:d2:c8:dd:53)
⊞ 802.1X Authentication

0000  00 19 d2 c8 dd 53 00 18  4d 9a d1 f8 88 8e 01 03   .....S.. M.......
0010  00 5f fe 00 89 00 20 00  00 00 00 00 00 00 01 89   ._.... . ........
0020  9f 50 64 d4 e4 9b a8 9b  25 55 d3 68 38 19 46 82   .Pd..... %U.h8.F.
0030  62 f4 c7 99 50 e3 34 45  3d d9 1e f7 68 a6 31 00   b...P.4E =...h.1.
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00                                                 .

File: "C:\Documents and Settings\jmalle...  Packets: 454 Displayed: 454 Marked: 0   Profile: Default
```

wireless.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: Expression... Clear Apply
Start
Page Load Error...
wireless.pcap - ...
SnagIt
5:59 PM

Figure 1.12: The protocol analyzer Wireshark monitoring a wireless interface.

The third-party analysis should involve a two-pronged approach: They should identify how the network appears to attackers and how secure the system is, should attackers make it past the perimeter defenses. You don't want to have "Tootsie Pop security"—a hard crunchy shell with a soft center. The external review, often called a *penetration test,* can be accomplished in several ways; the first is a *no knowledge* approach, whereby the consultants are provided with absolutely no information regarding the network and systems prior to their analysis. Though this is a very realistic approach, it can be time consuming and very expensive. Using this approach, consultants must use publicly available information to start enumerating systems for testing. This is a realistic approach, but a *partial knowledge* analysis is more efficient and less expensive. If provided with a network topology diagram and a list of registered IP addresses, the third-party reviewers can complete the review faster and the results can be addressed in a much more timely fashion. Once the penetration test is complete, a review of the internal network can be initiated. The audit of the internal network will identify open shares, unpatched systems, open ports, weak passwords, rogue systems, and many other issues.

I. Don't Forget the Basics

Many organizations spend a great deal of time and money addressing perimeter defenses and overlook some fundamental security mechanisms, as described here.

Change Default Account Passwords

Nearly all network devices come preconfigured with a password/username combination. This combination is included with the setup materials and is documented in numerous locations. Very often these devices are the gateways to the Internet or other internal networks. If these default passwords are not changed upon configuration, it becomes a trivial matter for an attacker to get into these systems. Hackers can find password lists on the Internet [44], and vendors include default passwords in their online manuals. For example, Figure 1.13 shows the default username and password for a Netgear router.

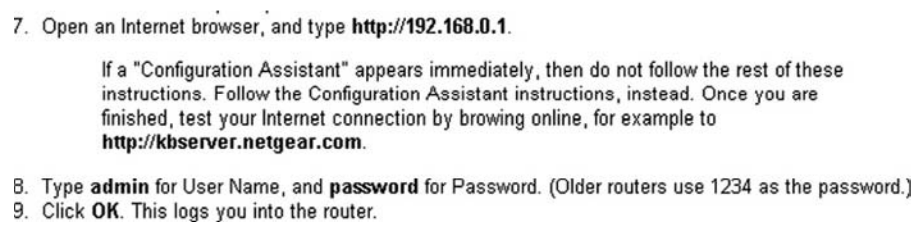
Use Robust Passwords

With the increased processing power of our computers and password-cracking software such as the Passware products [45] and AccessData's Password Recovery Toolkit [46], cracking passwords is fairly simple and straightforward. For this reason it is extremely important to create robust passwords. Complex passwords are hard for users to remember, though, so it is a challenge to create passwords that can be remembered without writing them down. One solution is to use the first letter of each word in a phrase, such as "**I** **l**ike **t**o **e**at **i**mported **c**heese **f**rom **H**olland." This becomes *IlteicfH*, which is an eight-character password using upper- and lowercase letters. This can be made even more complex by substituting an exclamation point for the letter *I* and substituting the number 3 for the letter *e*, so that the password becomes *!lt3icfH*. This is a fairly robust password that can be remembered easily.

Close Unnecessary Ports

Ports on a computer are logical access points for communication over a network. Knowing what ports are open on your computers will allow you to understand the types of access points that exist. The well-known port numbers are 0 through 1023. Some easily recognized ports and what they are used for are listed here:

- Port 21: FTP
- Port 23: Telnet
- Port 25: SMTP



7. Open an Internet browser, and type **http://192.168.0.1**.

   If a "Configuration Assistant" appears immediately, then do not follow the rest of these instructions. Follow the Configuration Assistant instructions, instead. Once you are finished, test your Internet connection by browing online, for example to **http://kbserver.netgear.com**.

8. Type **admin** for User Name, and **password** for Password. (Older routers use 1234 as the password.)
9. Click **OK**. This logs you into the router.

Figure 1.13: Default username and password for Netgear router.

- Port 53: DNS
- Port 80: HTTP
- Port 110: POP
- Port 119: NNTP

Since open ports that are not needed can be an entrance into your systems, and open ports that are open unexpectedly could be a sign of malicious software, identifying open ports is an important security process. There are several tools that will allow you to identify open ports. The built-in command-line tool *netstat* will allow you to identify open ports and process IDs by using the following switches:

-*a* Displays all connections and listening ports
-*n* Displays addresses and port numbers in numerical form
-*o* Displays the owning process ID associated with each connection

(*Note:* In Unix, netstat is also available but utilizes the following switches: -*atvp*.)

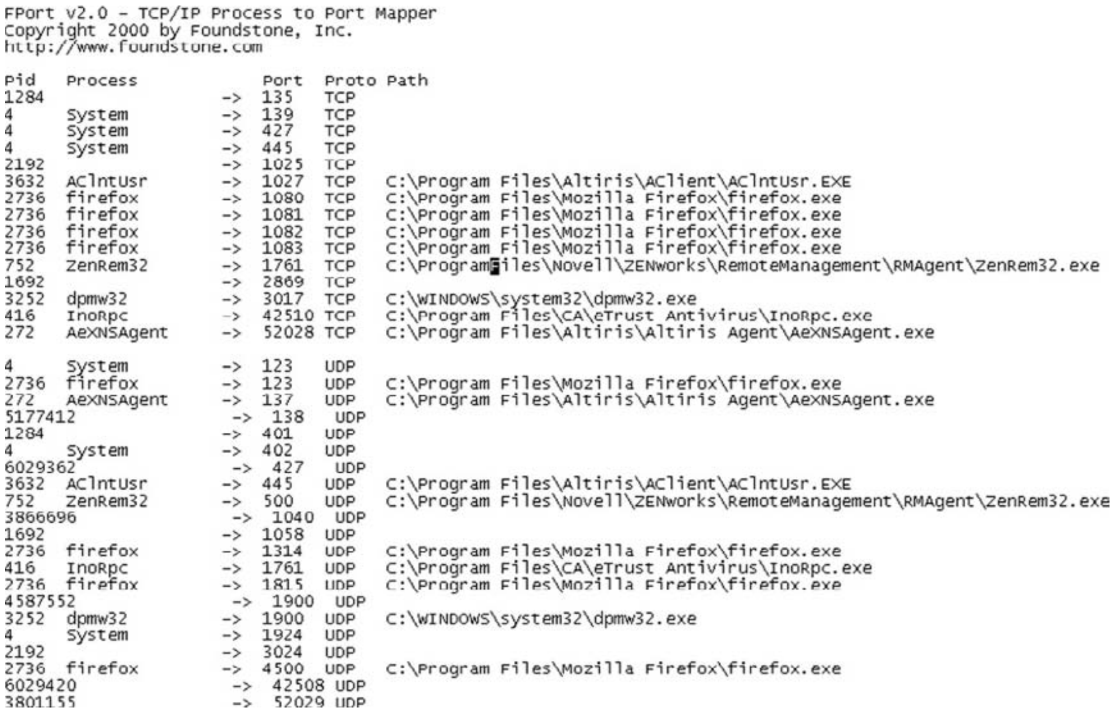
Other tools that can prove helpful are ActivePorts [47], a graphical user interface (GUI) tool that allows you to export the results in delimited format, and Fport [48], a popular command-line tool. Sample results are shown in Figure 1.14.

J. Patch, Patch, Patch

Nearly all operating systems have a mechanism for automatically checking for updates. This notification system should be turned on. Though there is some debate as to whether updates should be installed automatically, systems administrators should at least be notified of updates. They might not want to have them installed automatically, since patches and updates have been known to cause more problems than they solve. However, administrators should not wait too long before installing updates, because this can unnecessarily expose systems to attack. A simple tool that can help keep track of system updates is the Microsoft Baseline Security Analyzer [49], which also will examine other fundamental security configurations.

Use Administrator Accounts for Administrative Tasks

A common security vulnerability is created when systems administrators conduct administrative or personal tasks while logged into their computers with administrator rights. Tasks such as checking email, surfing the Internet, and testing questionable software can expose the computer to malicious software. This means that the malicious software can run with administrator privileges, which can create serious problems. Administrators should log into their systems using a standard user account to prevent malicious software from gaining control of their computers.



```
FPort v2.0 - TCP/IP Process to Port Mapper
Copyright 2000 by Foundstone, Inc.
http://www.foundstone.com

Pid   Process            Port  Proto Path
1284                 ->  135   TCP
4     System         ->  139   TCP
4     System         ->  427   TCP
4     System         ->  445   TCP
2192                 ->  1025  TCP
3632  AClntUsr       ->  1027  TCP   C:\Program Files\Altiris\AClient\AClntUsr.EXE
2736  firefox        ->  1080  TCP   C:\Program Files\Mozilla Firefox\firefox.exe
2736  firefox        ->  1081  TCP   C:\Program Files\Mozilla Firefox\firefox.exe
2736  firefox        ->  1082  TCP   C:\Program Files\Mozilla Firefox\firefox.exe
2736  firefox        ->  1083  TCP   C:\Program Files\Mozilla Firefox\firefox.exe
752   ZenRem32       ->  1761  TCP   C:\Program Files\Novell\ZENworks\RemoteManagement\RMAgent\ZenRem32.exe
1692                 ->  2869  TCP
3252  dpmw32         ->  3017  TCP   C:\WINDOWS\system32\dpmw32.exe
416   InoRpc         ->  42510 TCP   C:\Program Files\CA\eTrust Antivirus\InoRpc.exe
272   AeXNSAgent     ->  52028 TCP   C:\Program Files\Altiris\Altiris Agent\AeXNSAgent.exe

4     System         ->  123   UDP
2736  firefox        ->  123   UDP   C:\Program Files\Mozilla Firefox\firefox.exe
272   AeXNSAgent     ->  137   UDP   C:\Program Files\Altiris\Altiris Agent\AeXNSAgent.exe
5177412               ->  138   UDP
1284                 ->  401   UDP
4     System         ->  402   UDP
6029362               ->  427   UDP
3632  AClntUsr       ->  445   UDP   C:\Program Files\Altiris\AClient\AClntUsr.EXE
752   ZenRem32       ->  500   UDP   C:\Program Files\Novell\ZENworks\RemoteManagement\RMAgent\ZenRem32.exe
3866696               ->  1040  UDP
1692                 ->  1058  UDP
2736  firefox        ->  1314  UDP   C:\Program Files\Mozilla Firefox\firefox.exe
416   InoRpc         ->  1761  UDP   C:\Program Files\CA\eTrust Antivirus\InoRpc.exe
2736  firefox        ->  1815  UDP   C:\Program Files\Mozilla Firefox\firefox.exe
4587552               ->  1900  UDP
3252  dpmw32         ->  1900  UDP   C:\WINDOWS\system32\dpmw32.exe
4     System         ->  1924  UDP
2192                 ->  3024  UDP
2736  firefox        ->  4500  UDP   C:\Program Files\Mozilla Firefox\firefox.exe
6029420               ->  42508 UDP
3801155               ->  52029 UDP
```

Figure 1.14: Sample output from Fport.

Restrict Physical Access

With a focus on technology, it is often easy to overlook nontechnical security mechanisms. If an intruder can gain physical access to a server or other infrastructure asset, the intruder will own the organization. Critical systems should be kept in secure areas. A secure area is one that provides the ability to control access to only those who need access to the systems as part of their job responsibilities. A room that is kept locked using a key that is only provided to the systems administrator, with the only duplicate stored in a safe in the office manager's office, is a good start. The room should not have any windows that can open. In addition, the room should have no labels or signs identifying it as a server room or network operations center. The equipment should not be stored in a closet where other employees, custodians, or contractors can gain access. The validity of your security mechanisms should be reviewed during a third-party vulnerability assessment.

Don't Forget Paper!

With the advent of advanced technology, people have forgotten how information was stolen in the past—on paper. Managing paper documents is fairly straightforward. Locking file cabinets should be used—and locked consistently. Extra copies of proprietary documents, document drafts, and expired internal communications are some of the materials that should be shredded. A policy should be created to tell employees what they should and should not do with printed documents. The following example of the theft of trade secrets underscores the importance of protecting paper documents:

> *A company surveillance camera caught Coca-Cola employee Joya Williams at her desk looking through files and "stuffing documents into bags," Nahmias and FBI officials said. Then in June, an undercover FBI agent met at the Atlanta airport with another of the defendants, handing him $30,000 in a yellow Girl Scout Cookie box in exchange for an Armani bag containing confidential Coca-Cola documents and a sample of a product the company was developing, officials said [50].*

The steps to achieving security mentioned in this chapter are only the beginning. They should provide some insight into where to start building a secure organization.

References

[1] www.ncsl.org/programs/lis/cip/priv/breachlaws.htm (October 2, 2008).
[2] Pop quiz: What was the first personal computer? www.blinkenlights.com/pc.shtml (October 26, 2008).
[3] http://www.sixapart.com (March 24, 2009).
[4] www.dropsend.com (October 26, 2008).
[5] www.filesanywhere.com (October 26, 2008).
[6] www.swivel.com (October 26, 2008).
[7] www.getgspace.com (October 27, 2008).
[8] Report: Cybercrime groups starting to operate like the Mafia, published July 16 2008 http://arstechnica.com/news.ars/post/20080716-report-cybercrime-groups-starting-to-operate-like-the-mafia.html (October 27, 2008).
[9] www.gcio.nsw.gov.au/library/guidelines/resolveuid/87c81d4c6afbc1ae163024bd38aac9bd (October 29, 2008).
[10] www.csds.uidaho.edu/deb/costbenefit.pdf (October 29, 2008).
[11] Allen J, Pollak W. Why leaders should care about security. Podcast October 17, 2006. www.cert.org/podcast/show/20061017allena.html (November 2, 2008).
[12] http://nvd.nist.gov/home.cfm (October 29, 2008).
[13] Allen J, Pollak W. Why leaders should care about security. Podcast October 17, 2006. www.cert.org/podcast/show/20061017allena.html (November 2, 2008).
[14] www.cert.org/archive/pdf/05tn023.pdf.
[15] OCTAVE. www.cert.org/octave/ (November 2, 2008).
[16] Risk Management Framework. https://wiki.internet2.edu/confluence/display/secguide/Risk+Management+Framework.
[17] Cassandra. https://cassandra.cerias.purdue.edu/main/index.html.
[18] National Association of Professional Background Screeners. www.napbs.com.
[19] www.accuratebackground.com.
[20] www.credentialcheck.com.
[21] www.validityscreening.com.
[22] SANS Intro to Computer Security. www.sans.org.
[23] SANS Security Essentials Bootcamp. www.sans.org.

[24] www.globalknowledge.com/training/course.asp?pageid=9&courseid=10242&catid=191&country=United+States.

[25] ASIS International. www.asisonline.org; ISACA, www.isaca.org; HTCIA, www.htcia.org; ISSA, www.issa.org; InfraGard, www.infragard.net.

[26] (ISC)^2. www.isc2.org.

[27] http://windowsir.blogspot.com/2008/06/portable-devices-on-vista.html (November 8, 2008).

[28] RegRipper. www.regripper.net.

[29] DeviceWall. www.devicewall.com.

[30] Bluefire Security Technologies, 1010 Hull St., Ste. 210, Baltimore, Md. 21230.

[31] Gspace. www.getgspace.com.

[32] Gmail Drive. www.viksoe.dk/code/gmail.htm.

[33] Microsoft TechNet Library. http://technet.microsoft.com/en-us/library/default.aspx.

[34] WinHex. www.x-ways.net/winhex/index-m.html.

[35] Secure file deletion: Fact or fiction? www.sans.org/reading_room/whitepapers/incident/631.php (November 8, 2008).

[36] Users receive a password complexity requirements message that does not specify character group requirements for a password. http://support.microsoft.com/kb/821425 (November 8, 2008).

[37] PCI DSS. www.pcisecuritystandards.org/.

[38] Kiwi Syslog Daemon. www.kiwisyslog.com.

[39] Log Parser 2.2. www.microsoft.com/downloads/details.aspx?FamilyID=890cd06b-abf8-4c25-91b2-f8d975cf8c07&displaylang=en.

[40] Swatch. http://sourceforge.net/projects/swatch/.

[41] Cisco MARS. www.cisco.com/en/US/products/ps6241/.

[42] GFI EventsManager. www.gfi.com/eventsmanager/.

[43] Wireshark. www.wireshark.org.

[44] www.phenoelit-us.org/dpl/dpl.html.

[45] Passware. www.lostpassword.com.

[46] Password Recovery Toolkit. www.accessdata.com/decryptionTool.html.

[47] ActivePorts. www.softpile.com.

[48] Fport. www.foundstone.com/us/resources/proddesc/fport.htm.

[49] Microsoft Baseline Security Analyzer. http://technet.microsoft.com/en-us/security/cc184923.aspx.

[50] 3 accused in theft of Coke secrets. Washington Post July 26, 2006 www.washingtonpost.com/wp-dyn/content/article/2006/07/05/AR2006070501717.html (November 8, 2008).

This page intentionally left blank

CHAPTER 2

A Cryptography Primer

Scott R. Ellis
RGL Forensics

Man is a warrior creature, a species that ritually engages in a type of warfare where the combat can range from the subtlety of inflicting economic damage, or achieving economic superiority and advantage, to moving someone's chair a few inches from sitting distance or putting rocks in their shoes, to the heinousness of the outright killing of our opponents. As such, it is in our nature to want to prevent others who would do us harm from intercepting private communications (which could be about them!). Perhaps nothing so perfectly illustrates this fact as the art of cryptography. It is, in its purpose, an art form entirely devoted to the methods whereby we can prevent information from falling into the hands of those who would use it against us—our enemies.

Since the beginning of sentient language, cryptography has been a part of communication. It is as old as language itself. In fact, one could make the argument that the desire and ability to encrypt communication, to alter a missive in such a way so that only the intended recipient may understand it, is an innate ability hardwired into the human genome. Aside from the necessity to communicate, it could very well be what led to the development of language itself. Over time, languages and dialects evolve, as we can see with Spanish, French, Portuguese, and Italian—all "Latin" languages. People who speak French have a great deal of trouble understanding people who speak Spanish, and vice versa. The profundity of Latin cognates in these languages is undisputed, but generally speaking, the two languages are so far removed that they are not dialects, they are separate languages. But why is this? Certain abilities, such as walking, are hardwired into our nervous systems. Other abilities, such as language, are not.

So why isn't language hardwired into our nervous system, as it is with bees, who are born knowing how to tell another bee how far away a flower is, as well as the quantity of pollen and whether there is danger present? Why don't we humans all speak the exact same language? Perhaps we do, to a degree, but we choose not to do so. The reason is undoubtedly because humans, unlike bees, understand that knowledge is power, and knowledge is

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00002-6

communicated via spoken and written words. Plus we weren't born with giant stingers with which to simply sting people we don't like. With the development of evolving languages innate in our genetic wiring, the inception of cryptography was inevitable.

In essence, computer-based cryptography is the art of creating a form of communication that embraces the following precepts:

- Can be readily understood by the intended recipients
- Cannot be understood by unintended recipients
- Can be adapted and changed easily with relatively small modifications, such as a changed passphrase or word

Any artificially created lexicon, such as the Pig Latin of children, pictograph codes, gang-speak, or corporate lingo—and even the names of music albums, such as *Four Flicks*—are all manners of cryptography where real text, sometimes not so ciphered, is hidden in what appears to be plain text. They are attempts at hidden communications.

1. What Is Cryptography? What Is Encryption?

Ask any ancient Egyptian and he'll undoubtedly define *cryptography* as the practice of burying their dead so that they cannot be found again. They were very good at it; thousands of years later, new crypts are still being discovered. The Greek root *krypt* literally means "a hidden place," and as such it is an appropriate base for any term involving cryptology. According to the Online Etymology Dictionary, *crypto-* as a prefix, meaning "concealed, secret," has been used since 1760, and from the Greek *graphikos*, "of or for writing, belonging to drawing, picturesque." [1] Together, *crypto* + *graphy* would then mean "hiding place for ideas, sounds, pictures, or words." *Graph*, technically from its Greek root, is "the art of writing." *Encryption*, in contrast, merely means the act of carrying out some aspect of cryptography. *Cryptology*, with its *-ology* ending, is the study of cryptography. Encryption is subsumed by cryptography.

How Is Cryptography Done?

For most information technology occupations, knowledge of cryptography is a very small part of a broader skill set, and is generally limited to relevant application. The argument could be made that this is why the Internet is so extraordinarily plagued with security breaches. The majority of IT administrators, software programmers, and hardware developers are barely cognizant of the power of true cryptography. Overburdened with battling the plague that they inherited, they can't afford to devote the time or resources needed to implement a truly secure strategy. And the reason, as we shall come to see, is because as good at cryptographers can be—well, just as it is said that everyone has an evil twin somewhere in the world, for every cryptographer there is a de cryptographer working just as diligently to decipher a new encryption algorithm.

Traditionally, cryptography has consisted of any means possible whereby communications may be encrypted and transmitted. This could be as simple as using a language with which the opposition is not familiar. Who hasn't been somewhere where everyone around you was speaking a language you didn't understand? There are thousands of languages in the world, nobody can know them all. As was shown in World War II, when Allied Forces used Navajo as a means of communicating freely, some languages are so obscure that an entire nation may not contain one person who speaks it! All true cryptography is composed of three parts: a cipher, an original message, and the resultant encryption. The *cipher* is the method of encryption used. Original messages are referred to as *plain text* or as *clear text.* A message that is transmitted without encryption is said to be sent "in the clear." The resultant message is called *ciphertext* or *cryptogram.* This section begins with a simple review of cryptography procedures and carries them through; each section building on the next to illustrate the principles of cryptography.

2. Famous Cryptographic Devices

The past few hundred years of technical development and advances have brought greater and greater means to decrypt, encode, and transmit information. With the advent of the most modern warfare techniques and the increase in communication and ease of reception, the need for encryption has never been greater.

World War II publicized and popularized cryptography in modern culture. The Allied Forces' ability to capture, decrypt, and intercept Axis communications is said to have hastened WWII's end by several years. Here we take a quick look at some famous cryptographic devices from that era.

The Lorenz Cipher

The Lorenz cipher machine was an industrial-strength ciphering machine used in teleprinter circuits by the Germans during WWII. Not to be confused with its smaller cousin, the Enigma machine, the Lorenz cipher could possibly be best compared to a virtual private network tunnel for a telegraph line—only it wasn't sending Morse code, it was using a code not unlike a sort of American Standard Code for Information Interchange (ASCII) format. A granddaddy of sorts, called Baudot code, was used to send alphanumeric communications across telegraph lines. Each character was represented by a series of 5 bits.

It is often confused with the famous Enigma, but unlike the Enigma (which was a portable field unit), the Lorenz cipher could receive typed messages, encrypt them, send them to

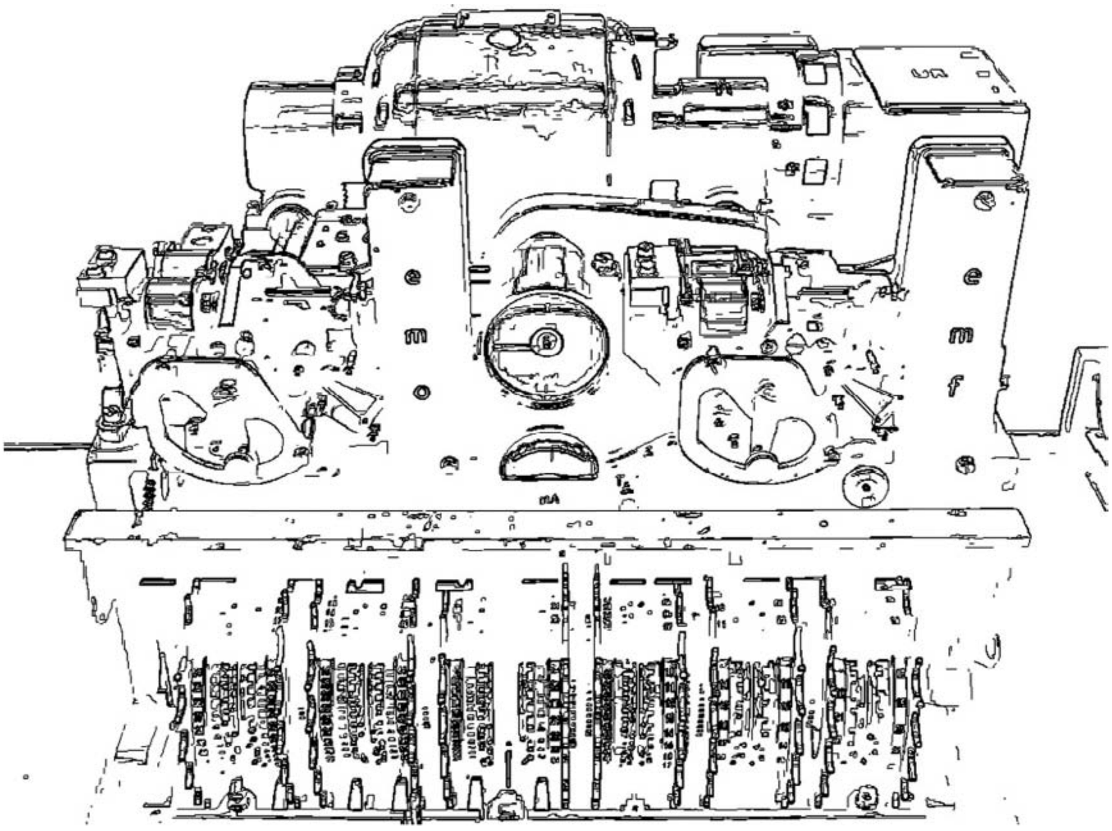


Figure 2.1: The Lorenz machine was set inline with a teletype to produce encrypted telegraphic signals.

another distant Lorenz cipher, which would then decrypt the signal. It used a pseudorandom cipher XOR'd with plaintext. The machine would be inserted inline as an attachment to a Lorenz teleprinter. Figure 2.1 is a rendered drawing from a photograph of a Lorenz cipher machine.

Enigma

The Enigma machine was a field unit used in WWII by German field agents to encrypt and decrypt messages and communications. Similar to the Feistel function of the 1970s, the Enigma machine was one of the first mechanized methods of encrypting text using an iterative cipher. It employed a series of rotors that, with some electricity, a light bulb, and a reflector, allowed the operator to either encrypt or decrypt a message. The original position of the rotors, set with each encryption and based on a prearranged pattern that in turn was based on the calendar, allowed the machine to be used, even if it was compromised.

When the Enigma was in use, with each subsequent key press, the rotors would change in alignment from their set positions in such a way that a different letter was produced each time. The operator, with a message in hand, would enter each character into the machine by pressing a typewriter-like key. The rotors would align, and a letter would then illuminate, telling the operator what the letter *really* was. Likewise, when enciphering, the operator would press the key and the illuminated letter would be the cipher text. The continually changing internal flow of electricity that caused the rotors to change was not random, but it did create a polyalphabetic cipher that could be different each time it was used.

3. Ciphers

Cryptography is built on one overarching premise: the need for a cipher that can reliably, and portably, be used to encrypt text so that, through any means of cryptanalysis—differential, deductive, algebraic, or the like—the ciphertext cannot be undone with any available technology. Throughout the centuries, there have been many attempts to create simple ciphers that can achieve this goal. With the exception of the One Time Pad, which is not particularly portable, success has been limited.

Let's look at a few of these methods now.

The Substitution Cipher

In this method, each letter of the message is replaced with a single character. See Table 2.1 for an example of a substitution cipher. Because some letters appear more often and certain words appear more often than others, some ciphers are extremely easy to decrypt, and some can be deciphered at a glance by more practiced cryptologists.

By simply understanding probability and with some applied statistics, certain metadata about a language can be derived and used to decrypt any simple, one-for-one substitution cipher. Decryption methods often rely on understanding the context of the *ciphertext*. What was encrypted—business communication? Spreadsheets? Technical data? Coordinates?
For example, using a hex editor and an access database to conduct some statistics, we can use the information in Table 2.2 to gain highly specialized knowledge about the data in

Table 2.1: A simple substitution cipher. Letters are numbered by their order in the alphabet to provide a numeric reference key. To encrypt a message, the letters are replaced, or substituted, by the numbers. This is a particularly easy cipher to reverse.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
O	C	Q	W	B	X	Y	E	I	L	Z	A	D	R	J	S	P	F	G	K	H	N	T	U	M	V
15	3	17	23	2	24	25	5	9	12	26	1	4	18	10	19	16	6	7	11	8	14	20	21	13	22

Table 2.2: Statistical data of interest in encryption. An analysis of a selection of a manuscript (in this case, the preedited version of Chapter 6 of *Managing Information Security*) can provide insight into the reasons that good ciphers need to be developed.

Character Analysis	Count
Number of distinct alphanumeric combinations	1958
Distinct characters	68
Number of four-letter words	984
Number of five-letter words	1375

Chapter 6, "Computer Forensics" by Scott R. Ellis, found in the book *Managing Information Security*, ed. by John Vacca. A long chapter at nearly 25,000 words, it provides a sufficiently large statistical pool to draw some meaningful analyses.

Table 2.3 gives additional data about the occurrence of specific words in "Computer Forensics". Note that because it is a technical text, words such as *computer, files, email,* and *drive* emerge as leaders. Analysis of these leaders can reveal individual and paired alpha frequencies. Being armed with knowledge about the type of communication can be very beneficial in decrypting it.

Further information about types of data being encrypted includes word counts by length of the word. Table 2.4 contains such a list for "Computer Forensics". This information can be used to begin to piece together useful and meaningful short sentences, which can provide cues to longer and more complex structures. It is exactly this sort of activity that good cryptography attempts to defeat.

Were it encrypted using a simple substitution cipher, a good start to deciphering "Computer Forensics" could be made using the information we've gathered. As a learning exercise, game, or logic puzzle, substitution ciphers are quite useful. Some substitution ciphers that are more elaborate can be just as difficult to crack. Ultimately, though, the weakness behind a substitution cipher is the fact that the ciphertext remains a one-to-one, directly corresponding substitution; ultimately anyone with a pen and paper and a large enough sample of the ciphertext can defeat it. Using a computer, deciphering a simple substitution cipher becomes child's play.

The Shift Cipher

Also known as the Caesar cipher, the shift cipher is one that anyone can readily understand and remember for decoding. It is a form of the substitution cipher. By shifting the alphabet a few positions in either direction, a simple sentence can become unreadable to casual inspection. Example 2.1 is an example of such a shift.

Table 2.3: Five-letter word recurrences in "Computer Forensics": A glimpse of the leading five-letter words found in the preedited manuscript. Once unique letter groupings have been identified, substitution, often by trial and error, can result in a meaningful reconstruction that allows the entire cipher to be revealed.

Words Field	Number of Recurrences
Files	125
Drive	75
There	67
email	46
These	43
Other	42
about	41
where	36
would	33
every	31
Court	30
Their	30
First	28
Using	28
which	24
Could	22
Table	22
After	21
image	21
Don't	19
Tools	19
Being	18
Entry	18

Interestingly, for cryptogram word games, the spaces are always included. Often puzzles use numbers instead of letters for the substitution. Removing the spaces in this particular example can make the ciphertext somewhat more secure. The possibility for multiple solutions becomes an issue; any number of words might fit the pattern.

Today many software tools are available to quickly and easily decode most cryptograms (at least, those that are not written in a dead language). You can have some fun with these tools; for example, the name Scott Ellis, when decrypted, turns into Still Books. The name of a friend of the author's decrypts to "His Sinless." It is apparent, then, that smaller-sample simple substitution ciphers can have more than one solution.

Much has been written and much has been said about frequency analysis; it is considered the "end-all and be-all" with respect to cipher decryption. This is not to be confused

Table 2.4: Leaders by word length in the preedited manuscript for "Computer Forensics". The context of the clear text can make the cipher less secure. There are, after all, only a finite number of words. Fewer of them are long.

Words Field	Number of Dupes	Word Length
XOriginalArrivalTime:	2	21
interpretations	2	15
XOriginatingIP:	2	15
Electronically	4	14
investigations	5	14
Interpretation	6	14
reconstructing	3	14
irreproducible	2	14
professionally	2	14
inexperienced	2	13
Demonstrative	2	13
XAnalysisOut:	8	13
Steganography	7	13
Understanding	8	13
Certification	2	13
circumstances	8	13
unrecoverable	4	13
Investigation	15	13
Automatically	2	13
Admissibility	2	13
XProcessedBy:	2	13
Administrator	4	13
Determination	3	13
Investigative	3	13
Practitioners	2	13
preponderance	2	13
Intentionally	2	13
Consideration	2	13
Interestingly	2	13

with cipher breaking, which is a modern attack against the actual cryptographic algorithms themselves. However, to think of simply plugging in some numbers generated from a Google search is a bit naïve. The frequency chart in Table 2.5 is commonplace on the Web.

It is beyond the scope of this chapter to delve into the accuracy of the table, but suffice it to say that our own analysis of the chapter's 118,000 characters, a technical text, yielded a much different result; see Table 2.6. Perhaps it is the significantly larger sample and the fact that it

Table 2.5: "In a random sampling of 1000 letters," this pattern emerges.

Letter	Frequency
E	130
T	93
N	78
R	77
I	74
O	74
A	73
S	63
D	44
H	35
L	35
C	30
F	28
P	27
U	27
M	25
Y	19
G	16
W	16
V	13
B	9
X	5
K	3
Q	3
J	2
Z	1
Total	**1000**

is a technical text that makes the results different after the top two. Additionally, where computers are concerned, an actual frequency analysis would take into consideration all ASCII characters, as shown in Table 2.6.

Frequency analysis is not difficult; once all the letters of a text are pulled into a database program, it is fairly straightforward to do a count of all the duplicate values. The snippet of code in Example 2.2 demonstrates one way whereby text can be transformed into a single column and imported into a database.

The cryptograms that use formatting (every word becomes the same length) are considerably more difficult for basic online decryption programs to crack. They must take into consideration spacing and word lengths when considering whether or not a string matches a word. It

Table 2.6: Using MS Access to perform some frequency analysis of "Computer Forensics". Characters with fewer repetitions than z were excluded from the return. Character frequency analysis of different types of communications yield slightly different results.

"Computer Forensics" Letters	Frequency
E	14,467
T	10,945
A	9239
I	8385
O	7962
S	7681
N	7342
R	6872
H	4882
L	4646
D	4104
C	4066
U	2941
M	2929
F	2759
P	2402
Y	2155
G	1902
W	1881
B	1622
V	1391
	1334
,	1110
K	698
0	490
X	490
Q	166
7	160
*	149
5	147
)	147
(	146
J	145
3	142
6	140
Æ	134
Ò	134
Ô	129
Ö	129
4	119
Z	116
Total	**116,798**

stands to reason, then, that the formulation of the cipher, where a substitution that is based partially on frequency similarities and with a whole lot of obfuscation so that when messages are decrypted they have ambiguous or multiple meanings, would be desirable for simple ciphers. However, this would only be true for very short and very obscure messages that could be code words to decrypt other messages or could simply be sent to misdirect the opponent. The amount of ciphertext needed to successfully break a cipher is called *unicity distance*. Ciphers with small unicity distances are weaker than those with large ones.

Example 2.1 A sample cryptogram. Try this out:

Gv Vw, Dtwvg?

Hint: Caesar said it, and it is Latin [2].

Example 2.2

```
Sub Letters2column ()
Dim bytText () As Byte
Dim bytNew() As Byte
Dim lngCount As Long
With ActiveDocument.Content
bytText = .Text
ReDim bytNew((((UBound(bytText()) + 1) * 2) - 5))
For lngCount = 0 To (UBound(bytText()) - 2) Step 2
bytNew((lngCount * 2)) = bytText(lngCount)
bytNew(((lngCount * 2) + 2)) = 13
Next lngCount
Text = bytNew()
End With
End Sub
```

Ultimately, substitution ciphers are vulnerable to either word-pattern analysis, letter-frequency analysis, or some combination of both. Where numerical information is encrypted, tools such as Benford's Law can be used to elicit patterns of numbers that *should* be occurring. Forensic techniques incorporate such tools to uncover accounting fraud. So, though this particular cipher is a child's game, it is useful in that it is an underlying principle of cryptography and should be well understood before continuing. The primary purpose of discussing it here is as an introduction to ciphers.

Further topics of interest and places to find information involving substitution ciphers are the chi-square statistic, Edgar Allan Poe, Sherlock Holmes, Benford's Law, Google, and Wikipedia.

The Polyalphabetic Cipher

The previous section clearly demonstrated that though the substitution cipher is fun and easy, it is also vulnerable and weak. It is especially susceptible to frequency analysis. Given a large enough sample, a cipher can easily be broken by mapping the frequency of the letters in the ciphertext to the frequency of letters in the language or dialect of the ciphertext (if it is known). To make ciphers more difficult to crack, Blaise de Vigenère from the 16th-century court of Henry III of France proposed a polyalphabetic substitution. In this cipher, instead of a one-to-one relationship, there is a one-to-many. A single letter can have multiple substitutes. The Vigenère solution was the first known cipher to use a keyword.

It works like this: First, a *tableau* is developed, as in Table 2.7. This tableau is a series of shift ciphers. In fact, since there can be only 26 additive shift ciphers, it is all of them.

In Table 2.7, a table in combination with a keyword is used to create the cipher. For example, if we choose the keyword *rockerrooks*, overlay it over the plaintext, and cross-index it to Table 2.7, we can produce the ciphertext. In this example, the top row is used to look up the plaintext, and the leftmost column is used to reference the keyword.

For example, we lay the word *rockerrooks* over the sentence, "Ask not what your country can do for you." Line 1 is the keyword, line 2 is the plain text, and line 3 is the ciphertext.

```
Keyword:     ROC  KER  ROOK  SROC  KERROOK  SRO  CK  ERR  OOK
Plaintext:   ASK  NOT  WHAT  YOUR  COUNTRY  CAN  DO  FOR  YOU
Ciphertext:  RGM  XSK  NVOD  QFIT  MSLEHFI  URB  FY  JFI  MCE
```

The similarity of this tableau to a mathematical table like the one in Table 2.8 becomes apparent. Just think letters instead of numbers and it becomes clear how this works. The top row is used to "look up" a letter from the plaintext, the leftmost column is used to locate the overlaying keyword letter, and where the column and the row intersect is the ciphertext.

This similarity is, in fact, the weakness of the cipher. Through some creative "factoring," the length of the keyword can be determined. Since the tableau is, in practice, a series of shift ciphers, the length of the keyword determines how many ciphers are used. The keyword *rockerrook*, with only six distinct letters, uses only six ciphers. Regardless, for nearly 300 years many people believed the cipher to be unbreakable.

Table 2.7: Vigenère's tableau arranges all of the shift ciphers in a single table. It then implements a keyword to create a more complex cipher than the simple substitution or shift ciphers. The number of spurious keys, that is, bogus decryptions that result from attempting to decrypt a polyalphabetic encryption, is greater than those created during the decryption of a single shift cipher.

Letter	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Table 2.8: The multiplication table is the inspiration for the Vigenère tableau.

Multiplier	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

The Kasiski/Kerckhoff Method

Now let's look at Kerckhoff's principle—"only secrecy of the key provides security" (not to be confused with Kirchhoff's law, a totally different man and rule). In the 19th century, Auguste Kerckhoff said that essentially, a system should still be secure, even when everyone knows everything about the system (except the password). Basically, his feeling was that if more than one person knows something, it's no longer a secret. Throughout modern cryptography, the inner workings of cryptographic techniques have been well known and published. Creating a portable, secure, unbreakable code is easy if nobody knows how it works. The problem lies in the fact that we people just can't keep a secret!

Example 2.3 A repetitious, weak keyword combines with plaintext to produce an easily deciphered ciphertext

```
Keyword        to to .toto.to. to. toto o to
Plaintext      It is. whatit .is, isn't.. it?
Ciphertext     BH....BG..PVTH....BH
                 BG...BGG...H...BH
```

In 1863 Kasiski, a Prussian major, proposed a method to crack the Vigenère cipher. His method, in short, required that the cryptographer deduce the length of the keyword used and then dissect the cryptogram into a corresponding number of ciphers. Each cipher would then be solved independently. The method required that a suitable number of bigrams be located. A *bigram* is a portion of the ciphertext, two characters long, that repeats itself in a discernible pattern. In Example 2.3, a repetition has been deliberately made simple with a short keyword (*toto*) and engineered by crafting a harmonic between the keyword and the plaintext.

This might seem an oversimplification, but it effectively demonstrates the weakness of the polyalphabetic cipher. Similarly, the polyalphanumeric ciphers, such as the Gronsfeld cipher, are even weaker since they use 26 letters and 10 digits. This one also happens to decrypt to "On of when on of," but a larger sample with such a weak keyword would easily be cracked by even the least intelligent of Web-based cryptogram solvers. The harmonic is created by the overlaying keyword with the underlying text; when the bigrams "line up" and repeat themselves, the highest frequency will be the length of the password. The distance between the two occurrences will be the length of the password. In Example 2.3, we see BH and BG repeating, and then we see BG repeating at a very tight interval of 2, which tells us the password might be two characters long and based on two shift ciphers that, when decrypted side by side, will make a real word. Not all bigrams will be indicators of this, so some care must be taken. As can be seen, BH repeats with an interval of 8, but the password is not eight digits long (but it is a factor of 8!). By locating the distance of all the repeating bigrams and factoring them, we can deduce the length of the keyword.

4. Modern Cryptography

Some of cryptography's greatest stars emerged in WWII. For the first time during modern warfare, vast resources were devoted to enciphering and deciphering communications. Both sides made groundbreaking advances in cryptography. Understanding the need for massive calculations (for the time—more is probably happening in the RAM of this author's PC over a period of five minutes than happened in all of WWII), both sides developed new machinery—predecessors to the modern solid-state computers—that could be coordinated to perform the calculations and procedures needed to crack enemy ciphers.

The Vernam Cipher (Stream Cipher)

Gilbert Sandford Vernam (1890–1960) was said to have invented the stream cipher in 1917. Vernam worked for Bell Labs, and his patent described a cipher in which a prepared key, on a paper tape, combined with plaintext to produce a transmitted ciphertext message. The same tape would then be used to decrypt the ciphertext. In effect, the Vernam and "one-time pad" ciphers are very similar. The primary difference is that the "one-time pad" cipher implements an XOR for the first time and dictates that a truly random stream cipher be used for the encryption. The stream cipher had no such requirement and used a different method of relay logic to combine a pseudo-random stream of bits with the plaintext bits. More about the XOR process is discussed in the section on XOR ciphering. In practice today, the Vernam cipher is any stream cipher in which pseudo-random or random text is combined with plaintext to produce cipher text that is the same length as the cipher. RC4 is a modern example of a Vernam cipher.

Example 2.4 Using the random cipher, a modulus shift instead of an XOR, and plaintext to produce ciphertext

```
Plaintext 1

t h i s w i l l b e s o e a s y t o b r e a k i t w i l l b e f u n n y
20 8 9 19 23 9 12 12 2 5 19 15 5 1 19 25 20 15 2 18 5 1 11 9 20 23 9 12 12 2 5 6
21 14 14 25

Cipher One

q e r t y u i o p a s d f g h j k l z x c v b n m q a z w s x e r f v t
17 5 18 20 25 21 9 15 16 1 19 4 6 7 8 10 11 12 26 24 3 22 2 14 13 17 1 26 23 19
24 5 18 6 22 20
```

```
CipherText 1

11 13 1 13 22 4 21 1 18 6 12 19 11 8 1 9 5 1 2 16 8 23 13 23 7 14 10 12 9 21 3
11 13 20 10 19
k m a m v d u a r f l s k h a i e a b p h w m w g n j l w u c k m t j s

Plaintext 2

T h i s w i l l n o t b e e a s y t o b r e a k o r b e t o o f u n n y
20 8 9 19 23 9 12 12 14 15 20 2 5 5 1 19 25 20 15 2 18 5 1 11 15 18 2 5 20 15 15
6 21 14 14 25

Ciphertext 2, also using Cipher One.

11 13 1 13 22 4 21 1 4 16 13 6 11 12 9 3 10 6 15 0 21 1 3 25 2 9 3 5 17 8 13 11
13 20 10 19
k m a m v d u a e p m f k l i f j f o z u a c y b i c e q h m k m t j s
```

The One-Time Pad

The “one-time pad” cipher, attributed to Joseph Mauborgne [3], is perhaps one of the most secure forms of cryptography. It is very difficult to break if used properly, and if the key stream is perfectly random, the ciphertext gives away absolutely no details about the plaintext, which renders it unbreakable. And, just as the name suggests, it uses a single random key that is the same length as the entire message, and it uses the key only once. The word *pad* is derived from the fact that the key would be distributed in pads of paper, with each sheet torn off and destroyed as it was used.

There are several weaknesses to this cipher. We begin to see that the more secure the encryption, the more it will rely on other means of key transmission. The more a key has to be moved around, the more likely it is that someone who shouldn’t have it will have it. The following weaknesses are apparent in this “bulletproof” style of cryptography:

- Key length has to equal plaintext length.
- It is susceptible to key interception; the key must be transmitted to the recipient, and the key is as long as the message!
- It’s cumbersome, since it doubles the traffic on the line.
- The cipher must be perfectly random.
- One-time use is absolutely essential. As soon as two separate messages are available, the messages can be decrypted. Example 2.4 demonstrates this.

Since most people don’t use binary, the author takes the liberty in Example 2.4 of using decimal numbers modulus 26 to represent the XOR that would take place in a bitstream encryption (see the section on the XOR cipher) that uses the method of the one-time pad.

Table 2.9: A simple key is created so that random characters and regular characters may be combined with a modulus function. Without the original cipher, this key is meaningless intelligence. It is used here in a similar capacity as an XOR, which is also a function that everyone knows how to do.

Key																									
A	b	c	D	e	f	g	h	i	j	k	l	m	n	o	p	Q	r	s	t	u	v	w	x	y	z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Example 2.5

The two ciphertexts, side by side, show a high level of harmonics. This indicates that two different ciphertexts actually have the same cipher. Where letters are different, since XOR is a known process and our encryption technique is also publicly known, it's a simple matter to say that r = 18, e = 5 (see Table 2.9) and thusly construct an algorithm that can tease apart the cipher and ciphertext to produce plaintext.

A numeric value is assigned to each letter, per Table 2.9. By assigning a numeric value to each letter, adding the plaintext value to the ciphertext value, modulus 26, yields a pseudo-XOR, or a wraparound Caesar shift that has a different shift for each letter in the entire message.

As this example demonstrates, by using the same cipher twice, a dimension is introduced that allows for the introduction of frequency analysis. By placing the two streams side by side, we can identify letters that are the same. In a large enough sample, where the cipher text is sufficiently randomized, frequency analysis of the aligned values will begin to crack the cipher wide open because we know that they are streaming in a logical order—the order in which they were written. One of the chief advantages of 21st-century cryptography is that the "eggs" are scrambled and descrambled during decryption based on the key, which you don't, in fact, want people to know. If the same cipher is used repeatedly, multiple inferences can be made and eventually the entire key can be deconstructed. Because plaintext 1 and plaintext 2 are so similar, this sample yields the following harmonics (in bold and boxed) as shown in Example 2.5.

Cracking Ciphers

One method of teasing out the frequency patterns is through the application of some sort of mathematical formula to test a hypothesis against reality. The chi-square test is perhaps one of the most commonly used; it allows someone to use what is called *inferential statistics* to draw certain inferences about the data by testing it against known statistical distributions.

Using the chi-square test against an encrypted text would allow certain inference to be made, but only where the contents, or the type of contents (random or of an expected distribution), of the text were known. For example, someone may use a program that encrypts files. By creating the null hypothesis that the text is completely random, and by reversing the encryption steps, a block cipher may emerge as the null hypothesis is disproved through the chi-square test. This would be done by reversing the encryption method and XORing against the bytes with a block created from the known text. At the point where the non-encrypted text matches the positioning of the encrypted text, chi-square would reveal that the output is not random and the block cipher would be revealed.

$$\textit{Chi-squared} = \ldots (\textit{observed} - \textit{expected})2/(\textit{expected})$$

Observed would be the actual zero/one ratio produced by XORing the data streams together, and expected would be the randomness of zeroes and ones (50/50) expected in a body of pseudorandom text.

Independent of having a portion of the text, a large body of encrypted text could be reverse encrypted using a block size of all zeroes; in this manner it may be possible to tease out a block cipher by searching for non random block sized strings. Modern encryption techniques generate many, many block cipher permutations that are layered against previous iterations (n-1) of permutated blocks. The feasibility of running such decryption techniques would require both a heavy-duty programmer, statistician, an incredible amount of processing power, and in-depth knowledge of the encryption algorithm used. An unpublished algorithm would render such testing worthless.

Some Statistical Tests for Cryptographic Applications by Adrian Fleissig

In many applications, it is often important to determine if a sequence is random. For example, a random sequence provides little or no information in cryptographic analysis. When estimating economic and financial models, it is important for the residuals from the estimated model to be random. Various statistical tests can be used to evaluate if a sequence is actually a random sequence or not. For a truly random sequence, it is assumed that each element is generated independently of any prior and/or future elements. A statistical test is used to compute the probability that the observed sequence is random compared to a truly random sequence. The procedures have test statistics that are used to evaluate the null hypothesis which typically assumes that the observed sequence is random. The alternative hypothesis is that the sequence is non random. Thus failing to accept the null hypothesis, at some critical level selected by the researcher, suggests that the sequence may be non random.

There are many statistical tests to evaluate for randomness in a sequence such as Frequency Tests, Runs Tests, Discrete Fourier Transforms, Serial Tests and many others. The tests statistics often have chi-square or standard normal distributions which are used *to evaluate the hypothesis. While no test is overall superior to the other tests, a Frequency or Runs Test is a good starting point to examine for non-randomness in a sequence. As an example, a Frequency or Runs Test typically evaluates if the number of zeros and ones in a sequence are about the same, as would be the case if the sequence was truly random.*

It is important to examine the results carefully. For example, the researcher may incorrectly fail to accept the null hypothesis that the sequence is random and thereby makes a Type I Error. Incorrectly accepting the null of randomness when the sequence is actually non random results in committing a Type II Error. The reliability of the results depends on having a sufficiently large number of elements in a sequence. In addition, it is important to perform alternative tests to evaluate if a sequence is random [4].

Notably, the methods and procedures used in breaking encryption algorithms are used throughout society in many applications where a null hypothesis needs to be tested. Forensic consultants use pattern matching and similar decryption techniques to combat fraud on a daily basis. Adrian Fleissig, a seasoned economist, makes use of many statistical tests to examine corporate data (see side bar, "Some Statistical Tests for Cryptographic Applications").

The XOR Cipher and Logical Operands

In practice, the XOR cipher is not so much a cipher as it is a mechanism whereby ciphertext is produced. *Random binary stream cipher* would be a better term. The terms *XOR, logical disjunction*, and *inclusive disjunction* may be used interchangeably. Most people are familiar with the logical functions of speech, which are words such as *and, or, nor,* and *not*. A girl can tell her brother, "Mother is either upstairs or at the neighbor's," which means she could be in either state, but you have no way of knowing which one it is. The mother could be in either place, and you can't infer from the statement the greater likelihood of either. The outcome is undecided.

Alternately, if a salesman offers a customer either a blue car or a red car, the customer knows that he can have red or he can have blue. Both statements are true. Blue cars and red cars exist simultaneously in the world. A person can own both a blue car and a red car. But Mother will never be in more than one place at a time. Purportedly, there is widespread belief that no author has produced an example of an English *or* sentence that appears to be false because both of its inputs are true [5]. Quantum physics takes considerable exception to this statement (which explains quantum physicists) at the quantum mechanical level. In the Schrodinger cat experiment, the sentence "The cat is alive or dead" or the statement "The photon is a particle and a wave until you look at it, then it is a particle or a wave, depending on how you observed it" both create a quandary for logical operations, and there are no Venn diagrams or words that are dependant on time or quantum properties of the physical universe. Regardless of this exception, when speaking of things in the world in a more rigorously descriptive fashion (in the macroscopically nonphenomenological sense), greater accuracy is needed.

Example 2.6 Line 1 and line 2 are combined with an XOR operand to produce line 3.

Line 1, plaintext : 1 0 0 1 1 1 0 1 0 1 1 0 1 1 1 1

Line 2, random cipher "": 1 0 0 0 1 1 0 1 0 1 0 0 1 0 0 1

Line 3, XOR ciphertext: 0 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0

To create a greater sense of accuracy in discussions of logic, the operands as listed in Figure 2.2 were created. When attempting to understand this chart, the best thing to do is to assign a word to the A and B values and think of each Venn diagram as a universe of documents, perhaps in a document database or just on a computer being searched. If A stands for the word *tree* and B for *frog*, then each letter simply takes on a very significant and distinct meaning.

In computing, it is traditional that a value of 0 is false and a value of 1 is true. An XOR operation, then, is the determination of whether two possibilities can be combined to produce a value of true or false, based on whether both operations are true, both are false, or one of the values is true.

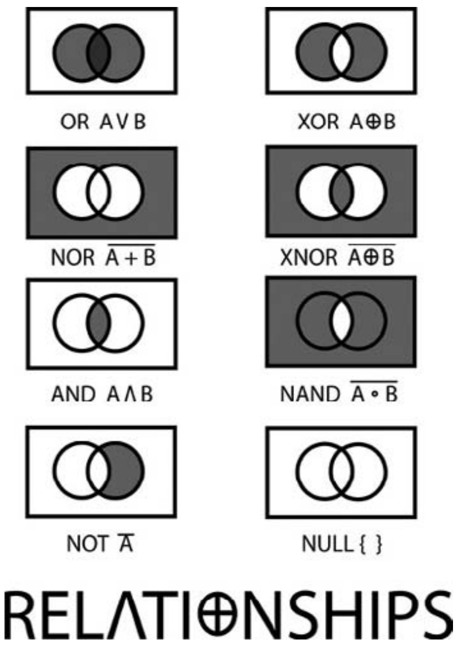


Figure 2.2: In each Venn diagram, the possible outcome of two inputs is decided.

```
1 XOR 1 = 0
0 XOR 0 = 0
1 XOR 0 = 1
0 XOR 1 = 1
```

In an XOR operation, if the two inputs are different, the resultant is TRUE, or 1. If the two inputs are the same, the resultant value is FALSE, or 0.

In Example 2.6, the first string represents the plaintext and the second line represents the cipher. The third line represents the ciphertext. If, and only exactly if, just one of the items has a value of TRUE, the results of the XOR operation will be true.

Without the cipher, and if the cipher is truly random, decoding the string becomes impossible. However, as in the one-time pad, if the same cipher is used, then (1) the cryptography becomes vulnerable to a known text attack, and (2) it becomes vulnerable to statistical analysis. Example 2.7 demonstrates this by showing exactly where the statistical aberration can be culled in the stream. If we know they both used the same cipher, can anyone solve for Plaintext A and Plaintext B?

Block Ciphers

Block ciphers work very similarly to the polyalphabetic cipher with the exception that a block cipher pairs together two algorithms for the creation of ciphertext and its decryption. It is also somewhat similar in that, where the polyalphabetic cipher used a repeating key, the block cipher uses a permutating, yet repeating, cipher block. Each algorithm uses two inputs: a key and a "block" of bits, each of a set size. Each output block is the same size as the input block, the block being transformed by the key. The key, which is algorithm based, is able to select the permutation of its bijective mapping from 2^n, where *n* is equal to the number of bits in the *input* block. Often, when 128-bit encryption is discussed, it is referring to the size of the *input* block. Typical encryption methods involve use of XOR chaining or some similar operation; see Figure 2.3.

Block ciphers have been very widely used since 1976 in many encryption standards. As such, cracking these ciphers became, for a long time, the top priority of cipher crackers everywhere. Block ciphers provide the backbone algorithmic technology behind most modern-era ciphers.

Example 2.7

To reconstruct the cipher if the plaintext is known, PlaintextA can be XOR'd to ciphertextB to produce cipherA! Clearly, in a situation where plaintext may be captured, using the same cipher key twice could completely expose the message. By using statistical analysis, the unique possibilities for PlaintextA and PlaintextB will emerge; *unique possibilities* means

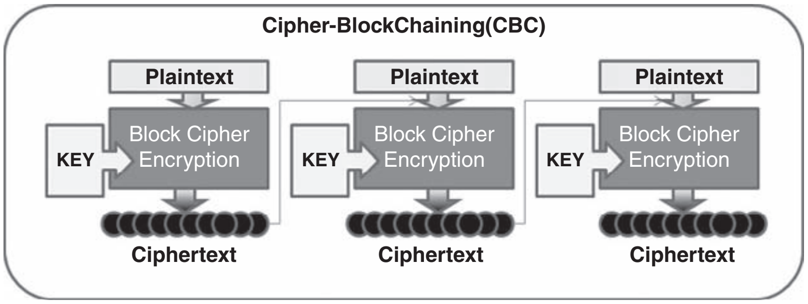


Figure 2.3: XOR chaining, or cipher-block chaining (CBC), is a method whereby the next block of plaintext to be encrypted is XOR'd with the previous block of ciphertext before being encrypted.

that for ciphertext = x, where the cipher is truly random, this should be at about 50% of the sample. Additions of ciphertext $n + 1$ will increase the possibilities for unique combinations because, after all, these binary streams must be converted to text and the set of binary stream possibilities that will combine into ASCII characters is relatively small. Using basic programming skills, you can develop algorithms that will quickly and easily sort through this data to produce a deciphered result. An intelligent person with some time on her hands could sort it out on paper or in an Excel spreadsheet. When the choice is "The red house down the street from the green house is where we will meet" or a bunch of garbage, it begins to become apparent how to decode the cipher.

```
CipherA and PlaintextA are XOR'd to produce ciphertextA:
PlaintextA: 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
cipherA: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
ciphertextA: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
PlaintextB and cipherA are XOR'd to produce ciphertextB:
ciphertextB: 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
cipherA: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
PlaintextB: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
| , - - - - - Column 1 - - - - - - - - - - . |, - - - - - - - - - Column 2 - - - - - - |
```

Note: Compare ciphertextA to ciphertextB!

5. The Computer Age

To many people, January 1, 1970, is considered the dawn of the computer age. That's when Palo Alto Research Center (PARC) in California introduced modern computing; the graphical user interface (no more command line and punch cards), networking on an Ethernet, and object-oriented programming have all been attributed to PARC. The 1970s also featured the Unix clock, Alan Shepherd on the moon, the U.S. Bicentennial, the civil rights movement, women's liberation, Robert Heinlein's sci-fi classic, *Stranger in a Strange Land*, and, most important to this chapter, modern cryptography. The late 1960s and early 1970s changed the face of the modern world at breakneck speed. Modern warfare reached tentative heights with radio-guided missiles, and warfare needed a new hero. And then there was the Data Encryption Standard, or DES; in a sense DES was the turning point for cryptography in that, for the first time, it fully leveraged the power of modern computing in its algorithms. The sky appeared to be the limit, but, unfortunately for those who wanted to keep their information secure, decryption techniques were not far behind.

Data Encryption Standard

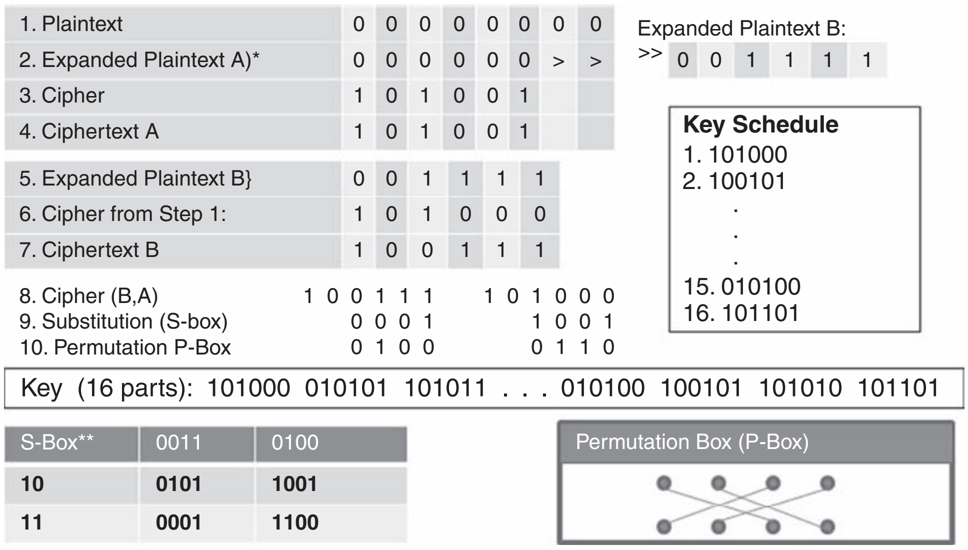
In the mid-1970s the U.S. government issued a public specification, through its National Bureau of Standards (NBS), called the Data Encryption Standard or, most commonly, DES. This could perhaps be considered the dawn of modern cryptography because it was very likely the first block cipher, or at least its first widespread implementation. But the 1970s were a relatively untrusting time. "Big Brother" loomed right around the corner (as per George Orwell's *1984*), and the majority of people didn't understand or necessarily trust DES. Issued under the NBS, now called the National Institute of Standards and Technology (NIST), hand in hand with the National Security Agency (NSA), DES led to tremendous interest in the reliability of the standard among academia's ivory towers. A shortened key length and the implementation of substitution boxes, or "S-boxes," in the algorithm led many to think that the NSA had deliberately weakened the algorithms and left a security "back door" of sorts.

The use of S-boxes in the standard was not generally understood until the design was published in 1994 by Don Coppersmith. The S-boxes, it turned out, had been deliberately designed to prevent a sort of cryptanalysis attack called *differential cryptanalysis*, as was discovered by IBM researchers in the early 1970s; the NSA had asked IBM to keep quiet about it. In 1990 the method was "rediscovered" independently and, when used against DES, the usefulness of the S-boxes became readily apparent.

Theory of Operation

DES used a 64-bit block cipher combined with a mode of operation based on cipher-block chaining (CBC) called the *Feistel function*. This consisted of an initial expansion permutation followed by 16 rounds of XOR key mixing via subkeys and a key schedule, substitution (S-boxes), and permutation [6]. In this strategy, a block is increased from 32 bits to 48 bits (expansion permutation). Then the 48-bit block is divided in half. The first half is XORs, with parts of the key according to a key schedule. These are called subkeys. Figure 2.4 shows this concept in a simplified format.

Feistel Structure of DES (Simplified)



1. Plaintext	0	0	0	0	0	0	0	0
2. Expanded Plaintext A)*	0	0	0	0	0	0	>	>
3. Cipher	1	0	1	0	0	1		
4. Ciphertext A	1	0	1	0	0	1		

Expanded Plaintext B:

>>	0	0	1	1	1	1

5. Expanded Plaintext B}	0	0	1	1	1	1
6. Cipher from Step 1:	1	0	1	0	0	0
7. Ciphertext B	1	0	0	1	1	1

8. Cipher (B,A)	1 0 0 1 1 1	1 0 1 0 0 0
9. Substitution (S-box)	0 0 0 1	1 0 0 1
10. Permutation P-Box	0 1 0 0	0 1 1 0

Key Schedule
1. 101000
2. 100101
.
.
.
15. 010100
16. 101101

Key (16 parts): 101000 010101 101011 . . . 010100 100101 101010 101101

S-Box**	0011	0100
10	**0101**	**1001**
11	**0001**	**1100**

*A bijective function is applied to expand from 32 bits (represented here by 8 bits) to 48 bits. A function is bijective if inverse relation $f^{-1}(x)$ is also a function. Reduction is achieved through the S-box operation.

** This S-box is reduced to include only the four bits that are in our cipher. Typical S-boxes are as large as needed and may be based partially on the key.

Figure 2.4: The Feistel function with a smaller key size.

The resulting cipher is then XOR'd with the half of the cipher that was not used in step 1. The two halves switch sides. Substitution boxes reduce the 48 bits down to 32 bits via a nonlinear function and then a permutation, according to a permutation table, takes place. Then the entire process is repeated again, 16 times, except in the last step the two halves are not flipped. Finally, this diffusive strategy produced via substitution, permutation, and key schedules creates an effective ciphertext. Because a fixed-length cipher, a block cipher, is used, the permutations and the S-box introduce enough confusion that the cipher cannot be deduced through brute-force methods without extensive computing power.

With the increase in size of hard drives and computer memory, the need for disk space and bandwidth still demand that a block cipher algorithm be portable. DES, Triple DES, and the Advanced Encryption Standard (AES) all provide or have provided solutions that are secure and practical.

Implementation

Despite the controversy at the time, DES was implemented. It became the encryption standard of choice until the late 1990s, when it was broken when Deep Crack and distributed.net broke a DES key in 22 hours and 15 minutes. Later that year a new form of DES called Triple DES, which encrypted the plaintext in three iterations, was published. It remained in effect until 2002, when it was superseded by AES.

Rivest, Shamir, and Adleman (RSA)

The release of DES also included the creation and release of Ron Rivest, Adi Shamir, and Leonard Adleman's encryption algorithm (RSA). Rivest, Shamir, and Adleman, based at the Massachusetts Institute of Technology (MIT), publicly described the algorithm in 1977. RSA is the first encryption standard to introduce (to public knowledge) the new concept of digital signing. In 1997 it was revealed through declassification of papers that Clifford Cocks, a British mathematician working for the U.K. Government Communications Headquarters (GCHQ), had, in 1973, written a paper describing this process. Assigned a status of top secret, the work had previously never seen the light of day. Because it was submitted in 1973, the method had been considered unattainable, since computing power at the time could not handle its methods.

Advanced Encryption Standard (AES or Rijndael)

AES represents one of the latest chapters in the history of cryptography. It is currently one of the most popular of encryption standards and, for people involved in any security work, its occurrence on the desktop is frequent. It also enjoys the free marketing and acceptance that it received when it was awarded the title of official cryptography standard in 2001 [7]. This designation went into effect in May of the following year.

Similarly to DES, AES encrypts plaintext in a series of rounds, involves the use of a key and block sizes, and leverages substitution and permutation boxes. It differs from DES in the following respects:

- It supports 128-bit block sizes.
- The key schedule is based on the S-box.
- It expands the key, not the plaintext.
- It is not based on a Feistel cipher.
- It is extremely complex.

The AES algorithms are to symmetric ciphers what a bowl of spaghetti is to the shortest distance between two points. Through a series of networked XOR operations, key substitutions, temporary variable transformations, increments, iterations, expansions, value swapping, S-boxing, and the like, a very strong encryption is created that, with modern computing, is impossible to break. It is conceivable that, with so complex a series of operations, a computer file and block could be combined in such a way as to produce all zeroes. Theoretically, the AES cipher could be broken by solving massive quadratic equations that take into consideration every possible vector and solve 8000 quadratic equations with 1600 binary unknowns. This sort of an attack is called an *algebraic attack* and, where traditional methods such as differential or differential cryptanalysis fail, it is suggested that the strength in AES lies in the current inability to solve supermultivariate quadratic equations with any sort of efficiency.

Reports that AES is not as strong as it should be are likely, at this time, to be overstated and inaccurate, because anyone can present a paper that is dense and difficult to understand and claims to achieve the incredible [8]. It is unlikely that, any time in the near or maybe not-so-near future (this author hedges his bets), AES will be broken using multivariate quadratic polynomials in thousands of dimensions. Mathematica is very likely one of the most powerful tools that can solve quadratic equations, and it is still many years away from being able to perform this feat.

References

[1] www.etymonline.com.
[2] *Et tu, Brute?*
[3] Wikipedia, Gilbert Vernam entry.
[4] Adrian Fleissig is the Senior Economist of Counsel for RGL Forensics, 2006–present. He is also a Full Professor, California State University Fullerton (CSUF) since 2003 with a joint Ph.D. in Economics and Statistics. North Carolina State University; 1993.
[5] Barrett, Stenner. The myth of the exclusive "Or". Mind 1971;80(317):116–21.
[6] Sorkin A. LUCIFER: A cryptographic algorithm. Cryptologia 1984;8(1):22–35.
[7] U.S. FIPS PUB 197 (FIPS 197), November 26, 2001.
[8] Schneier B. Crypto-Gram Newsletter, September 15, 2002.

CHAPTER 3

Preventing System Intrusions

Michael West
Independent Technical Writer

The moment you establish an active Web presence, you put a target on your company's back. And like the hapless insect that lands in the spider's web, your company's size determines the size of the disturbance you create on the Web—and how quickly you're noticed by the bad guys. How attractive you are as prey is usually directly proportionate to what you have to offer a predator. If yours is an ecommerce site whose business thrives on credit card or other financial information or a company with valuable secrets to steal, your "juiciness" quotient goes up; you have more of value there to steal. And if your business is new and your Web presence is recent, the assumption could be made that perhaps you're not yet a seasoned veteran in the nuances of cyber warfare and, thus, are more vulnerable to an intrusion.

Unfortunately for you, many of those who seek to penetrate your network defenses are educated, motivated, and quite brilliant at developing faster and more efficient methods of quietly sneaking around your perimeter, checking for the smallest of openings. Most IT professionals know that an enterprise's firewall is ceaselessly being probed for weaknesses and vulnerabilities by crackers from every corner of the globe. Anyone who follows news about software understands that seemingly every few months, word comes out about a new, exploitable opening in an operating system or application. It's widely understood that no one—not the most savvy network administrator or the programmer who wrote the software—can possibly find and close all the holes in today's increasingly complex software.

Bugs exist in applications, operating systems, server processes (daemons), and clients. System configurations can also be exploited, such as not changing the default administrator's password or accepting default system settings, or unintentionally leaving a hole open by configuring the machine to run in a nonsecure mode. Even Transmission Control Protocol/Internet Protocol (TCP/IP), the foundation on which all Internet traffic operates, can be exploited, since the protocol was designed before the threat of hacking was really widespread. Therefore it contains design flaws that can allow, for example, a cracker to easily alter IP data.

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00003-8

Once the word gets out that a new and exploitable opening exists in an application (and word *will* get out), crackers around the world start scanning sites on the Internet searching for any and all sites that have that particular opening.

Making your job even harder is the fact that many openings into your network can be caused by your employees. Casual surfing of porn sites can expose the network to all kinds of nasty bugs and malicious code, merely by an employee visiting the site. The problem is that, to users, it might not seem like such a big deal. They either don't realize or don't care that they're leaving the network wide open to intrusion.

1. So, What Is an Intrusion?

A network intrusion is an unauthorized penetration of a computer in your enterprise or an address in your assigned domain. An intrusion can be passive (in which penetration is gained stealthily and without detection) or active (in which changes to network resources are affected). Intrusions can come from outside your network structure or inside (an employee, customer, or business partner). Some intrusions are simply meant to let you know the intruder was there, defacing your Web site with various kinds of messages or crude images. Others are more malicious, seeking to extract critical information on either a one-time basis or as an ongoing parasitic relationship that will continue to siphon off data until it's discovered. Some intruders will seek to implant carefully crafted code designed to crack passwords, record keystrokes, or mimic your site while directing unaware users to their site. Others will embed themselves into the network and quietly siphon off data on a continuing basis or to modify public-facing Web pages with various kinds of messages.

An attacker can get into your system physically (by having physical access to a restricted machine and its hard drive and/or BIOS), externally (by attacking your Web servers or finding a way to bypass your firewall), or internally (your own users, customers, or partners).

2. Sobering Numbers

So how often do these intrusions occur? The estimates are staggering: Depending on which reporting agency you listen to, anywhere from 79 million to over 160 million compromises of electronic data occurred worldwide between 2007 and 2008. U.S. government statistics show an estimated 37,000 known and reported incidents against federal systems alone in 2007, and the number is expected to rise as the tools employed by crackers become increasingly sophisticated.

In one case, credit- and debit-card information for over 45 million users was stolen from a large merchant in 2005, and data for an additional 130,000 were lifted in 2006. Merchants reported that the loss would cost them an estimated $5 million.

Spam continues to be one of the biggest problems faced by businesses today and has been steadily increasing every year. An Internet threat report published by Secure Computing Corporation in October 2008 states, “The acquisition of innocent machines via email and Web-based infections continued in Q3 with over 5000 new zombies created every hour.” [1] And in the election year of 2008, election-related spam messages were estimated to exceed 100 million messages per day.

According to research done by Secure Computing, malware use is also on a steady rise, “with nearly 60% of all malware-infected URLs” coming from the United States and China. And Web-related attacks will become more widespread, with political and financially motivated attacks topping the list. With the availability of Web attack toolkits increasing, Secure Computing’s research estimates that “about half of all Web-borne attacks will likely be hosted on compromised legitimate Web sites.”

Alarmingly, there is also a rise in teenage involvement in cracking. Chris Boyd, director of malware research at FaceTime Security, was quoted in an October 29, 2008, posting on the BBC’s Web site that he’s “seeing kids of 11 and 12 sharing credit-card details and asking for hacks.” [2] Some of the teens have resorted to posting videos of their work on YouTube, not realizing they’ve thus made it incredibly easy to track them down. But the fact that they exist and are sharing information via well-known venues is worrisome enough, the fumbling teen crackers of today are tomorrow’s network security nightmares in the making.

Whatever the goal of the intrusion—fun, greed, bragging rights, or theft of data—the end result is the same: a weakness in your network security has been detected and exploited. And unless you discover that weakness—the intrusion entry point—it will continue to be an open door into your environment.

So, just who’s out there looking to break into your network?

3. Know Your Enemy: Hackers versus Crackers

An entire community of people—experts in programming and computer networking and those who thrive on solving complex problems—have been around since the earliest days of computing. The term *hacker* originated from the members of this culture, and they are quick to point out that it was hackers who built and make the Internet run, and hackers who created the Unix operating system. Hackers see themselves as members of a community who build things and make them work. And the term *cracker* is, to those in their culture, a badge of honor.

Ask a traditional hacker about people who sneak into computer systems to steal data or cause havoc, and he’ll most likely correct you by telling you those people aren’t true hackers. (In the cracker community, the term for these types is *cracker*, and the two labels aren't synonymous.) So, to not offend traditional hackers, I'll use the term *crackers* and focus on them and their efforts.

From the lone-wolf cracker seeking peer recognition to the disgruntled former employee out for revenge or the deep pockets and seemingly unlimited resources of a hostile government bent on taking down wealthy capitalists, crackers are out there in force, looking to find the chink in your system's defensive armor.

A cracker's specialty—or in some cases, his mission in life—is seeking out and exploiting vulnerabilities of an individual computer or network for their own purposes. Crackers' intentions are normally malicious and/or criminal in nature. They have, at their disposal, a vast library of information designed to help them hone their tactics, skills, and knowledge, and they can tap into the almost unlimited experience of other crackers through a community of like-minded individuals sharing information across underground networks.

They usually begin this life learning the most basic of skills: software programming. The ability to write code that can make a computer do what they want is seductive in and of itself. As they learn more and more about programming, they also expand their knowledge of operating systems and, as a natural course of progression, operating systems' weaknesses. They also quickly learn that, to expand the scope and type of their illicit handiwork, they need to learn HTML—the code that allows them to create phony Web pages that lure unsuspecting users into revealing important financial or personal data.

There are vast underground organizations to which these new crackers can turn for information. They hold meetings, write papers, and develop tools that they pass along to each other. Each new acquaintance they meet fortifies their skill set and gives them the training to branch out to more and more sophisticated techniques. Once they gain a certain level of proficiency, they begin their trade in earnest.

They start off simply by researching potential target firms on the Internet (an invaluable source for all kinds of corporate network related information). Once a target has been identified, they might quietly tiptoe around, probing for old forgotten back doors and operating system vulnerabilities. They can start off simply and innocuously by running basic DNS queries that can provide IP addresses (or ranges of IP addresses) as starting points for launching an attack. They might sit back and listen to inbound and/or outbound traffic, record IP addresses, and test for weaknesses by pinging various devices or users.

They can surreptitiously implant password cracking or recording applications, keystroke recorders, or other malware designed to keep their unauthorized connection alive—and profitable.

The cracker wants to act like a cyber-ninja, sneaking up to and penetrating your network without leaving any trace of the incursion. Some more seasoned crackers can put multiple layers of machines, many hijacked, between them and your network to hide their activity. Like standing in a room full of mirrors, the attack appears to be coming from so many locations you can't pick out the real from the ghost. And before you realize what they've done, they've up and disappeared like smoke in the wind.

4. Motives

Though the goal is the same—to penetrate your network defenses—crackers' motives are often different. In some cases, a network intrusion could be done from the inside by a disgruntled employee looking to hurt the organization or steal company secrets for profit.

There are large groups of crackers working diligently to steal credit-card information that they then turn around and make available for sale. They want a quick grab and dash—take what they want and leave. Their cousins are the network parasites—those who quietly breach your network, then sit there siphoning off data.

A new and very disturbing trend is the discovery that certain governments have been funding digital attacks on network resources of both federal and corporate systems. Various agencies from the U.S. Department of Defense to the governments of New Zealand, France, and Germany have reported attacks originating from unidentified Chinese hacking groups. It should be noted that the Chinese government denies any involvement, and there is no evidence that it is or was involved. Furthermore, in October 2008, the South Korean Prime Minister is reported to have issued a warning to his cabinet that "about 130,000 items of government information had been hacked [by North Korean computer crackers] over the past four years." [3]

5. Tools of the Trade

Crackers today are armed with an increasingly sophisticated and well-stocked tool kit for doing what they do. Like the professional thief with his custom-made lock picks, crackers today can obtain a frightening array of tools to covertly test your network for weak spots. Their tools range from simple password-stealing malware and keystroke recorders (loggers) to methods of implanting sophisticated parasitic software strings that copy data streams coming in from customers who want to perform an ecommerce transaction with your company. Some of the more widely used tools include these:

- *Wireless sniffers*. Not only can these devices locate wireless signals within a certain range, they can siphon off the data being transmitted over the signals. With the rise in popularity and use of remote wireless devices, this practice is increasingly responsible for the loss of critical data and represents a significant headache for IT departments.

- *Packet sniffers.* Once implanted in a network data stream, these passively analyze data packets moving into and out of a network interface, and utilities capture data packets passing through a network interface.
- *Port scanners.* A good analogy for these utilities is a thief casing a neighborhood, looking for an open or unlocked door. These utilities send out successive, sequential connection requests to a target system's ports to see which one responds or is open to the request. Some port scanners allow the cracker to slow the rate of port scanning—sending connection requests over a longer period of time—so the intrusion attempt is less likely to be noticed. These devices' usual targets are old, forgotten "back doors," or ports inadvertently left unguarded after network modifications.
- *Port knocking.* Sometimes network administrators create a secret back-door method of getting through firewall-protected ports—a secret knock that enables them to quickly access the network. Port-knocking tools find these unprotected entries and implant a Trojan horse that listens to network traffic for evidence of that secret knock.
- *Keystroke loggers.* These are spyware utilities planted on vulnerable systems that record a user's keystrokes. Obviously, when someone can sit back and record every keystroke a user makes, it doesn't take long to obtain things like usernames, passwords, and ID numbers.
- *Remote administration tools.* Programs embedded on an unsuspecting user's system that allow the cracker to take control of that system.
- *Network scanners.* Explore networks to see the number and kind of host systems on a network, the services available, the host's operating system, and the type of packet filtering or firewalls being used.
- *Password crackers.* These sniff networks for data streams associated with passwords and then employ a brute-force method of peeling away any encryption layers protecting those passwords.

6. Bots

A new and particularly virulent threat that has emerged over the past few years is one in which a virus is surreptitiously implanted in large numbers of unprotected computers (usually those found in homes), hijacking them (without the owners' knowledge) and turning them into slaves to do the cracker's bidding. These compromised computers, known as *bots*, are linked in vast and usually untraceable networks called *botnets*. Botnets are designed to operate in such a way that instructions come from a central PC and are rapidly shared among other botted computers in the network. Newer botnets are now using a "peer-to-peer" method that, because they lack a central identifiable point of control, makes it difficult if not impossible for law enforcement agencies to pinpoint. And because they often cross international boundaries into countries without the means (or will) to investigate and shut them down, they can grow with alarming speed. They can be so lucrative that they've now become the cracker's tool of choice.

Botnets exist, in large part, because of the number of users who fail to observe basic principles of computer security—installed and/or up-to-date antivirus software, regular scans for suspicious code, and so on—and thereby become unwitting accomplices. Once taken over and "botted," their machines are turned into channels through which large volumes of unwanted spam or malicious code can be quickly distributed. Current estimates are that, of the 800 million computers on the Internet, up to 40% are bots controlled by cyber thieves who are using them to spread new viruses, send out unwanted spam email, overwhelm Web sites in denial-of-service (DoS) attacks, or siphon off sensitive user data from banking or shopping Web sites that look and act like legitimate sites with which customers have previously done business.

It's such a pervasive problem that, according to a report published by security firm Damballa [4], botnet attacks rose from an estimated 300,000 per day in August 2006 to over 7 million per day one year later, and over 90% of what was sent out was spam email. Even worse for ecommerce sites is a growing trend in which a site's operators are threatened with DoS attacks unless they pay protection money to the cyber extortionist. Those who refuse to negotiate with these terrorists quickly see their sites succumb to relentless rounds of cyber "carpet bombing."

Bot controllers, also called *herders*, can also make money by leasing their networks to others who need a large and untraceable means of sending out massive amounts of advertisements but don't have the financial or technical resources to create their own networks. Making matters worse is the fact that botnet technology is available on the Internet for less than $100, which makes it relatively easy to get started in what can be a very lucrative business.

7. Symptoms of Intrusions

As stated earlier, your company's mere presence on the Web places a target on your back. It's only a matter of time before you experience your first attack. It could be something as innocent looking as several failed login attempts or as obvious as an attacker having defaced your Web site or crippled your network. It's important that you go into this knowing you're vulnerable.

Crackers are going to first look for known weaknesses in the operating system (OS) or any applications you are using. Next, they would start probing, looking for holes, open ports, or forgotten back doors—faults in your security posture that can quickly or easily be exploited.

Arguably one of the most common symptoms of an intrusion—either attempted or successful—is repeated signs that someone is trying to take advantage of your organization's own security systems, and the tools you use to keep watch for suspicious network activity may actually be used against you quite effectively. Tools such as network security and file integrity scanners, which can be invaluable at helping you conduct ongoing assessments of your network's vulnerability, are also available and can be used by crackers looking for a way in.

Large numbers of unsuccessful login attempts are also a good indicator that your system has been targeted. The best penetration-testing tools can be configured with attempt thresholds that, when exceeded, will trigger an alert. They can passively distinguish between legitimate and suspicious activity of a repetitive nature, monitor the time intervals between activities (alerting when the number exceeds the threshold you set), and build a database of signatures seen multiple times over a given period.

The "human element" (your users) is a constant factor in your network operations. Users will frequently enter a mistyped response but usually correct the error on the next try. However, a sequence of mistyped commands or incorrect login responses (with attempts to recover or reuse them) can be a signs of brute-force intrusion attempts.

Packet inconsistencies—direction (inbound or outbound), originating address or location, and session characteristics (ingoing sessions vs. outgoing sessions)—can also be good indicators of an attack. If a packet has an unusual source or has been addressed to an abnormal port—say, an inconsistent service request—it could be a sign of random system scanning. Packets coming from the outside that have local network addresses that request services on the inside can be a sign that IP spoofing is being attempted.

Sometimes odd or unexpected system behavior is itself a sign. Though this is sometimes difficult to track, you should be aware of activity such as changes to system clocks, servers going down or server processes inexplicably stopping (with system restart attempts), system resource issues (such as unusually high CPU activity or overflows in file systems), audit logs behaving in strange ways (decreasing in size without administrator intervention), or unexpected user access to resources. If you note unusual activity at regular times on given days, heavy system use (possible DoS attack) or CPU use (brute-force password-cracking attempts) should always be investigated.

8. What Can You Do?

It goes without saying that the most secure network—the one that has the least chance of being compromised—is one that has no direct connection to the outside world. But that's hardly a practical solution, since the whole reason you have a Web presence is to do business. And in the game of Internet commerce, your biggest concern isn't the sheep coming in but the wolves dressed like sheep coming in with them. So, how do you strike an acceptable balance between keeping your network intrusion free and keeping it accessible at the same time?

As your company's network administrator, you walk a fine line between network security and user needs. You have to have a good defensive posture that still allows for access. Users and customers can be both the lifeblood of your business and its greatest potential source of infection. Furthermore, if your business thrives on allowing users access, you have no choice but to let them in. It seems like a monumentally difficult task at best.

Like a castle, imposing but stationary, every defensive measure you put up will eventually be compromised by the legions of very motivated thieves looking to get in. It's a game of move/countermove: You adjust, they adapt. So you have to start with defenses that can quickly and effectively adapt and change as the outside threats adapt.

First and foremost, you need to make sure that your perimeter defenses are as strong as they can be, and that means keeping up with the rapidly evolving threats around you. The days of relying solely on a firewall that simply does firewall functions are gone; today's crackers have figured out how to bypass the firewall by exploiting weaknesses in applications themselves. Simply being reactive to hits and intrusions isn't a very good option, either; that's like standing there waiting for someone to hit you before deciding what to do rather than seeing the oncoming punch and moving out of its way or blocking it. You need to be flexible in your approach to the newest technologies, constantly auditing your defenses to ensure that your network's defensive armor can meet the latest threat. You have to have a very dynamic and effective policy of constantly monitoring for suspicious activities that, when discovered, can be quickly dealt with so that someone doesn't slip something past without your noticing it. Once that happens, it's too late.

Next, and this is also a crucial ingredient for network administrators: You have to educate your users. No matter how good a job you've done at tightening up your network security processes and systems, you still have to deal with the weakest link in your armor—your users. It doesn't do any good to have bulletproof processes in place if they're so difficult to manage that users work around them to avoid the difficulty, or if they're so loosely configured that a casually surfing user who visits an infected site will pass that infection along to your network. The degree of difficulty in securing your network increases dramatically as the number of users goes up.

User education becomes particularly important where mobile computing is concerned. Losing a device, using it in a place (or manner) in which prying eyes can see passwords or data, awareness of hacking tools specifically designed to sniff wireless signals for data, and logging on to unsecured networks are all potential problem areas with which users need to be familiar.

Know Today's Network Needs

The traditional approach to network security engineering has been to try to erect preventative measures—firewalls—to protect the infrastructure from intrusion. The firewall acts like a filter, catching anything that seems suspicious and keeping everything behind it as sterile as possible. However, though firewalls are good, they typically don't do much in the way of identifying compromised applications that use network resources. And with the speed of evolution seen in the area of penetration tools, an approach designed simply to prevent attacks will be less and less effective.

Today's computing environment is no longer confined to the office, as it used to be. Though there are still fixed systems inside the firewall, ever more sophisticated remote and mobile devices are making their way into the workforce. This influx of mobile computing has expanded the traditional boundaries of the network to farther and farther reaches and requires a different way of thinking about network security requirements.

Your network's endpoint or perimeter is mutating—expanding beyond its historical boundaries. Until recently, that endpoint was the user, either a desktop system or laptop, and it was relatively easy to secure those devices. To use a metaphor: The difference between endpoints of early network design and those of today is like the difference between the battles of World War II and the current war on terror. In the battles of WWII there were very clearly defined "front lines"—one side controlled by the Allied powers, the other by the Axis. Today, however, the war on terror has no such front lines and is fought in multiple areas with different techniques and strategies that are customized for each combat theater.

With today's explosion of remote users and mobile computing, your network's endpoint is no longer as clearly defined as it once was, and it is evolving at a very rapid pace. For this reason, your network's physical perimeter can no longer be seen as your best "last line of defense," even though having a robust perimeter security system is still a critical part of your overall security policy.

Any policy you develop should be organized in such a way as to take advantage of the strength of your unified threat management (UTM) system. Firewalls, antivirus, and intrusion detection systems (IDSs), for example, work by trying to block all currently known threats—the "blacklist" approach. But the threats evolve more quickly than the UTM systems can, so it almost always ends up being an "after the fact" game of catch-up. Perhaps a better, and more easily managed, policy is to specifically state which devices are allowed access and which applications are allowed to run in your network's applications. This "whitelist" approach helps reduce the amount of time and energy needed to keep up with the rapidly evolving pace of threat sophistication, because you're specifying what gets in versus what you have to keep out.

Any UTM system you employ should provide the means of doing two things: specify which applications and devices are allowed and offer a policy-based approach to managing those applications and devices. It should allow you to secure your critical resources against unauthorized data extraction (or data leakage), offer protection from the most persistent threats (viruses, malware, and spyware), and evolve with the ever-changing spectrum of devices and applications designed to penetrate your outer defenses.

So, what's the best strategy for integrating these new remote endpoints? First, you have to realize that these new remote, mobile technologies are becoming increasingly ubiquitous and aren't going away anytime soon. In fact, they most likely represent the future of computing. As these devices gain in sophistication and function, they are unchaining end users from their desks and, for some businesses, are indispensable tools. iPhones, Blackberries, Palm Treos, and other smart phones and devices now have the capability to interface with corporate email systems, access networks, run enterprise-level applications, and do full-featured remote computing. As such, they also now carry an increased risk for network administrators due to loss or theft (especially if the device is unprotected by a robust authentication method) and unauthorized interception of their wireless signals from which data can be siphoned off.

To cope with the inherent risks, you engage an effective security policy for dealing with these devices: under what conditions can they be used, how many of your users need to employ them, what levels and types of access will they have, and how will they be authenticated?

Solutions are available for adding strong authentication to users seeking access via wireless LANs. Tokens, either of the hardware or software variety, are used to identify the user to an authentication server for verification of their credentials. For example, PremierAccess by Aladdin Knowledge Systems can handle incoming access requests from a wireless access point and, if the user is authenticated, pass them into the network.

Key among the steps you take to secure your network while allowing mobile computing is to fully educate the users of such technology. They need to understand, in no uncertain terms, the risks to your network (and ultimately to the company in general) represented by their mobile devices and that their mindfulness of both the device's physical and electronic security is an absolute necessity.

Network Security Best Practices

So, how do you either "clean and tighten up" your existing network or design a new one that can stand up to the inevitable onslaught of attacks? Let's look at some basics. Consider the diagram shown in Figure 3.1.

The illustration in Figure 3.1 shows what could be a typical network layout. Users outside the DMZ approach the network via a secure (HTTPS) Web or VPN connection. They are

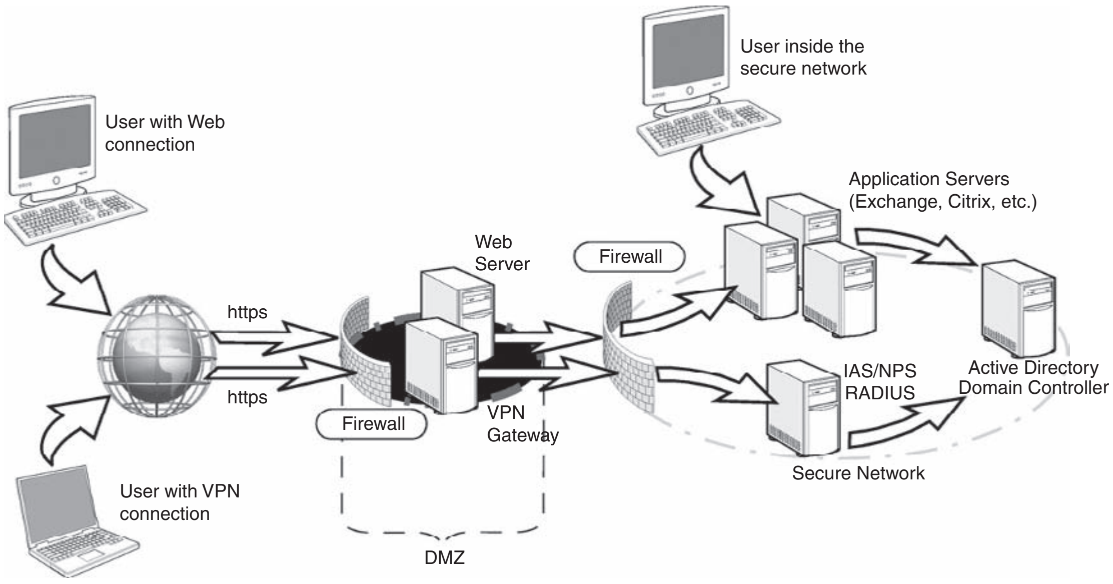


Figure 3.1: Network diagram.

authenticated by the perimeter firewall and handed off to either a Web server or a VPN gateway. If allowed to pass, they can then access resources inside the network.

If you're the administrator of an organization that has only, say, a couple dozen users with whom to contend, your task (and the illustration layout) will be relatively easy to manage. But if you have to manage several hundred (or several thousand) users, the complexity of your task increases by an order of magnitude. That makes a good security policy an absolute necessity.

9. Security Policies

Like the tedious prep work before painting a room, organizations need a good, detailed, and well-written security policy. Not something that should be rushed through "just to get it done," your security policy should be well thought out; in other words, the "devil is in the details." Your security policy is designed to get everyone involved with your network "thinking along the same lines."

The policy is almost always a work in progress. It must evolve with technology, especially those technologies aimed at surreptitiously getting into your system. The threats will continue to evolve, as will the systems designed to hold them at bay.

A good security policy isn't always a single document; rather, it is a conglomeration of policies that address specific areas, such as computer and network use, forms of authentication, email policies, remote/mobile technology use, and Web surfing policies. It should be written in such a way that, while comprehensive, it can be easily understood by those it affects. Along those lines, your policy doesn't have to be overly complex. If you hand new employees something that resembles *War and Peace* in size and tell them they're responsible for knowing its content, you can expect to have continued problems maintaining good network security awareness. Keep it simple.

First, you need to draft some policies that define your network and its basic architecture. A good place to start is by asking the following questions:

- What kinds of resources need to be protected (user financial or medical data, credit-card information, etc.)?
- How many users will be accessing the network on the inside (employees, contractors, etc.)?
- Will there need to be access only at certain times or on a 24/7 basis (and across multiple time zones and/or internationally)?
- What kind of budget do I have?
- Will remote users be accessing the network, and if so, how many?
- Will there be remote sites in geographically distant locations (requiring a failsafe mechanism, such as replication, to keep data synched across the network)?

Next, you should spell out responsibilities for security requirements, communicate your expectations to your users (one of the weakest links in any security policy), and lay out the role(s) for your network administrator. It should list policies for activities such as Web surfing, downloading, local and remote access, and types of authentication. You should address issues such as adding users, assigning privileges, dealing with lost tokens or compromised passwords, and under what circumstances you will remove users from the access database.

You should establish a security team (sometimes referred to as a "tiger team") whose responsibility it will be to create security policies that are practical, workable, and sustainable. They should come up with the best plan for implementing these policies in a way that addresses both network resource protection and user friendliness. They should develop plans for responding to threats as well as schedules for updating equipment and software. And there should be a very clear policy for handling changes to overall network security—the types of connections through your firewall that will and will not be allowed. This is especially important because you don't want an unauthorized user gaining access, reaching into your network, and simply taking files or data.

10. Risk Analysis

You should have some kind of risk analysis done to determine, as near as possible, the risks you face with the kind of operations you conduct (ecommerce, classified/proprietary information handling, partner access, or the like). Depending on the determined risk, you might need to rethink your original network design. Though a simple extranet/intranet setup with mid-level firewall protection might be okay for a small business that doesn't have much to steal, that obviously won't work for a company that deals with user financial data or proprietary/classified information. In that case, what might be needed is a tiered system in which you have a "corporate side" (on which things such as email, intranet access, and regular Internet access are handled) and a separate, secure network not connected to the Internet or corporate side. These networks can only be accessed by a user on a physical machine, and data can only be moved to them by "sneaker-net" physical media (scanned for viruses before opening). These networks can be used for data systems such as test or lab machines (on which, for example, new software builds are done and must be more tightly controlled, to prevent inadvertent corruption of the corporate side), or networks on which the storage or processing of proprietary, business-critical, or classified information are handled. In Department of Defense parlance, these are sometimes referred to as *red nets* or *black nets*.

Vulnerability Testing

Your security policy should include regular vulnerability testing. Some very good vulnerability testing tools, such as WebInspect, Acunetix, GFI LANguard, Nessus, HFNetChk, and Tripwire, allow you to conduct your own security testing. Furthermore, there are third-party companies with the most advanced suite of testing tools available that can be contracted to scan your network for open and/or accessible ports, weaknesses in firewalls, and Web site vulnerability.

Audits

You should also factor in regular, detailed audits of all activities, with emphasis on those that seem to be near or outside established norms. For example, audits that reveal high rates of data exchanges after normal business hours, when that kind of traffic would not normally be expected, is something that should be investigated. Perhaps, after checking, you'll find that it's nothing more than an employee downloading music or video files. But the point is that your audit system saw the increase in traffic and determined it to be a simple Internet use policy violation rather than someone siphoning off more critical data.

There should be clearly established rules for dealing with security, use, and/or policy violations as well as attempted or actual intrusions. Trying to figure out what to do after the intrusion is too late. And if an intrusion does occur, there should be a clear-cut system for determining the extent of damage; isolation of the exploited application, port, or machine; and a rapid response to closing the hole against further incursions.

Recovery

Your plan should also address the issue of recovery after an attack has occurred. You need to address issues such as how the network will be reconfigured to close off the exploited opening. This might take some time, since the entry point might not be immediately discernable. There has to be an estimate of damage—what was taken or compromised, was malicious code implanted somewhere, and, if so, how to most efficiently extract it and clean the affected system. In the case of a virus in a company's email system, the ability to send and receive email could be halted for days while infected systems are rebuilt. And there will have to be discussions about how to reconstruct the network if the attack decimated files and systems.

This will most likely involve more than simply reinstalling machines from archived backups. Because the compromise will most likely affect normal business operations, the need to expedite the recovery will hamper efforts to fully analyze just what happened.

This is the main reason for preemptively writing a disaster recovery plan and making sure that all departments are represented in its drafting. However, like the network security policy itself, the disaster recovery plan will also be a work in progress that should be reviewed regularly to ensure that it meets the current needs. Things such as new threat notifications, software patches and updates, vulnerability assessments, new application rollouts, and employee turnover all have to be addressed.

11. Tools of Your Trade

Though the tools available to people seeking unauthorized entry into your domain are impressive, you also have a wide variety of tools to help keep them out. Before implementing a network security strategy, however, you must be acutely aware of the specific needs of those who will be using your resources.

Simple antispyware and antispam tools aren't enough. In today's rapidly changing software environment, strong security requires penetration shielding, threat signature recognition, autonomous reaction to identified threats, and the ability to upgrade your tools as the need arises.

The following discussion talks about some of the more common tools you should consider adding to your arsenal.

Firewalls

Your first line of defense should be a good firewall, or better yet, a system that effectively incorporates several security features in one. Secure Firewall (formerly Sidewinder) from Secure Computing is one of the strongest and most secure firewall products available, and as of this writing it has never been successfully hacked. It is trusted and used by government and defense agencies. Secure Firewall combines the five most necessary security systems—firewall, antivirus/spyware/spam, virtual private network (VPN), application filtering, and intrusion prevention/detection systems—into a single appliance.

Intrusion Prevention Systems

A good *intrusion prevention system* (IPS) is a vast improvement over a basic firewall in that it can, among other things, be configured with policies that allow it to make autonomous decisions as to how to deal with application-level threats as well as simple IP address or port-level attacks. IPS products respond directly to incoming threats in a variety of ways, from automatically dropping (extracting) suspicious packets (while still allowing legitimate ones to pass) to, in some cases, placing an intruder into a "quarantine" file. IPS, like an application layer firewall, can be considered another form of access control in that it can make pass/fail decisions on application content.

For an IPS to be effective, it must also be very good at discriminating between a real threat signature and one that looks like but isn't one (false positive). Once a signature interpreted to be an intrusion is detected, the system must quickly notify the administrator so that the appropriate evasive action can be taken. The following are types of IPS:

- *Network-based.* Network-based IPSs create a series of choke points in the enterprise that detect suspected intrusion attempt activity. Placed inline at their needed locations, they invisibly monitor network traffic for known attack signatures that they then block.
- *Host-based.* These systems don't reside on the network per se but rather on servers and individual machines. They quietly monitor activities and requests from applications, weeding out actions deemed prohibited in nature. These systems are often very good at identifying post-decryption entry attempts.
- *Content-based.* These IPSs scan network packets, looking for signatures of content that is unknown or unrecognized or that has been explicitly labeled threatening in nature.
- *Rate-based.* These IPSs look for activity that falls outside the range of normal levels, such as activity that seems to be related to password cracking and brute-force penetration attempts, for example.

When searching for a good IPS, look for one that provides, at minimum:

- Robust protection for your applications, host systems, and individual network elements against exploitation of vulnerability-based threats as “single-bullet attacks,” Trojan horses, worms, botnets, and surreptitious creation of “back doors” in your network
- Protection against threats that exploit vulnerabilities in specific applications such as Web services, mail, DNS, SQL, and any Voice over IP (VoIP) services
- Detection and elimination of spyware, phishing, and anonymizers (tools that hide a source computer’s identifying information so that Internet activity can be undertaken surreptitiously)
- Protection against brute-force and DoS attacks, application scanning, and flooding
- A regular method of updating threat lists and signatures

Application Firewalls

Application firewalls (AFs) are sometimes confused with IPSs in that they can perform IPS-like functions. But an AF is specifically designed to limit or deny an application’s level of access to a system’s OS—in other words, closing any openings into a computer’s OS to deny the execution of harmful code within an OS’s structure. AFs work by looking at applications themselves, monitoring the kind of data flow from an application for suspicious or administrator-blocked content from specific Web sites, application-specific viruses, and any attempt to exploit an identified weakness in an application’s architecture. Though AF systems can conduct intrusion prevention duties, they typically employ proxies to handle firewall access control and focus on traditional firewall-type functions. Application firewalls can detect the signatures of recognized threats and block them before they can infect the network.

Windows’ version of an application firewall, called Data Execution Prevention (DEP), prevents the execution of any code that uses system services in such a way that could be deemed harmful to data or Virtual Memory (VM). It does this by considering RAM data as nonexecutable—in essence, refusing to run new code coming from the data-only area of RAM, since any harmful or malicious code seeking to damage existing data would have to run from this area.

The Macintosh Operating System (MacOS) Version 10.5.x also includes a built-in application firewall as a standard feature. The user can configure it to employ two-layer protection in which installing network-aware applications will result in an OS-generated warning that prompts for user authorization of network access. If authorized, MacOS will digitally sign the application in such a way that subsequent application activity will not prompt for further authorization. Updates invalidate the original certificate, and the user will have to revalidate before the application can run again.

The Linux OS has, for example, an application firewall called AppArmor that allows the admin to create and link to every application a security policy that restricts its access capabilities.

Access Control Systems

Access control systems (ACSs) rely on administrator-defined rules that allow or restrict user access to protected network resources. These access rules can, for example, require strong user authentication such as tokens or biometric devices to prove the identity of users requesting access. They can also restrict access to various network services based on time of day or group need.

Some ACS products allow for the creation of an *access control list* (ACL), which is a set of rules that define security policy. These ACLs contain one or more *access control entries* (ACEs), which are the actual rule definitions themselves. These rules can restrict access by specific user, time of day, IP address, function (department, management level, etc.), or specific system from which a logon or access attempt is being made.

A good example of an ACS is SafeWord by Aladdin Knowledge Systems. SafeWord is considered a two-factor authentication system in that it uses what the user knows (such as a personal identification number, or PIN) and what the user has (such as a one-time passcode, or OTP, token) to strongly authenticate users requesting network access. SafeWord allows administrators to design customized access rules and restrictions to network resources, applications, and information.

In this scheme, the tokens are a key component. The token's internal cryptographic key algorithm is made "known" to an authentication server when the token's file is imported into a central database.

When the token is assigned to a user, its serial number is linked to that user in the user's record. On making an access request, the authentication server prompts the user to enter a username and the OTP generated by the token. If a PIN was also assigned to that user, she must either prepend or append that PIN to the token-generated passcode. As long as the authentication server receives what it expects, the user is granted whatever access privileges she was assigned.

Unified Threat Management

The latest trend to emerge in the network intrusion prevention arena is referred to as *unified threat management*, or UTM. UTM systems are multilayered and incorporate several security technologies into a single platform, often in the form of a plug-in appliance. UTM products can provide such diverse capabilities as antivirus, VPN, firewall services, and antispam as well as intrusion prevention.

The biggest advantages of a UTM system are its ease of operation and configuration and the fact that its security features can be quickly updated to meet rapidly evolving threats.

Sidewinder by Secure Computing is a UTM system that was designed to be flexible, easily and quickly adaptable, and easy to manage. It incorporates firewall, VPN, trusted source, IPS, antispam and antivirus, URL filtering, SSL decryption, and auditing/reporting.

Other UTM systems include Symantec's Enterprise Firewall and Gateway Security Enterprise Firewall Appliance, Fortinet, LokTek's AIRlok Firewall Appliance, and SonicWall's NSA 240 UTM Appliance, to name a few.

12. Controlling User Access

Traditionally users—also known as employees—have been the weakest link in a company's defensive armor. Though necessary to the organization, they can be a nightmare waiting to happen to your network. How do you let them work within the network while controlling their access to resources? You have to make sure your system of user authentication knows who your users are.

Authentication, Authorization, and Accounting

Authentication is simply proving that a user's identity claim is valid and authentic. Authentication requires some form of "proof of identity." In network technologies, physical proof (such as a driver's license or other photo ID) cannot be employed, so you have to get something else from a user. That typically means having the user respond to a challenge to provide genuine credentials at the time he requests access.

For our purposes, credentials can be something the user knows, something the user has, or something they are. Once they provide authentication, there also has to be authorization, or permission to enter. Finally, you want to have some record of users' entry into your network—username, time of entry, and resources. That is the accounting side of the process.

What the User Knows

Users know a great many details about their own lives—birthdays, anniversaries, first cars, their spouse's name—and many will try to use these nuggets of information as a simple form of authentication. What they don't realize is just how insecure those pieces of information are.

In network technologies, these pieces of information are often used as fixed passwords and PINs because they're easy to remember. Unless some strict guidelines are established on what form a password or PIN can take (e.g., a minimum number of characters or a mixture of letters and numbers), a password will offer little to no real security.

Unfortunately, to hold down costs, some organizations allow users to set their own passwords and PINs as credentials and then rely on a simple challenge-response mechanism in which these weak credentials are provided to gain access. Adding to the loss of security is the fact that not only are the fixed passwords far too easy to guess, but because the user already has too much to remember, she writes them down somewhere near the computer she uses (often in some "cryptic" scheme to make it more difficult to guess). To increase the effectiveness of any security system, that system needs to require a much stronger form of authentication.

What the User Has

The most secure means of identifying users is by a combination of (1) hardware device in their possession that is "known" to an authentication server in your network, coupled with (2) what they know. A whole host of devices available today—tokens, smart cards, biometric devices—are designed to more positively identify a user. Since it's my opinion that a good token is the most secure of these options, I focus on them here.

Tokens

A *token* is a device that employs an encrypted key for which the encryption algorithm—the method of generating an encrypted password—is known to a network's authentication server. There are both software and hardware tokens. The software tokens can be installed on a user's desktop system, in their cellular phone, or on their smart phone. The hardware tokens come in a variety of form factors, some with a single button that both turns the token on and displays its internally generated passcode; others with a more elaborate numerical keypad for PIN input. If lost or stolen, tokens can easily be removed from the system, quickly rendering them completely ineffective. And the passcodes they generate are of the "one-time-passcode," or OTP, variety, meaning that a generated passcode expires once it's been used and cannot be used again for a subsequent logon attempt.

Tokens are either programmed onsite with token programming software or offsite at the time they are ordered from their vendor. During programming, functions such as a token's cryptographic key, password length, whether a PIN is required, and whether it generates passwords based on internal clock timing or user PIN input are written into the token's memory. When programming is complete, a file containing this information and the token's serial number are imported into the authentication server so that the token's characteristics are known.

A token is assigned to a user by linking its serial number to the user's record, stored in the system database. When a user logs onto the network and needs access to, say, her email, she is presented with some challenge that she must answer using her assigned token.

Tokens operate in one of three ways: time synchronous, event synchronous, or challenge-response (also known as asynchronous).

Time Synchronous

In time synchronous operation, the token's internal clock is synched with the network's clock. Each time the token's button is pressed, it generates a passcode in hash form, based on its internal timekeeping. As long as the token's clock is synched with the network clock, the passcodes are accepted. In some cases (e.g., when the token hasn't been used for some time or its battery dies), the token gets out of synch with the system and needs to be resynched before it can be used again.

Event Synchronous

In event synchronous operations, the server maintains an ordered passcode sequence and determines which passcode is valid based on the current location in that sequence.

Challenge–Response

In challenge–response, a challenge, prompting for username, is issued to the user by the authentication server at the time of access request. Once the user's name is entered, the authentication server checks to see what form of authentication is assigned to that user and issues a challenge back to the user. The user inputs the challenge into the token, then enters the token's generated response to the challenge. As long as the authentication server receives what it expected, authentication is successful and access is granted.

The User Is Authenticated, But Is She Authorized?

Authorization is independent of authentication. A user can be permitted entry into the network but not be authorized to access a resource. You don't want an employee having access to HR information or a corporate partner getting access to confidential or proprietary information.

Authorization requires a set of rules that dictate the resources to which a user will have access. These permissions are established in your security policy.

Accounting

Say that our user has been granted access to the requested resource. But you want (or in some cases are required to have) the ability to call up and view activity logs to see who got into what resource. This information is mandated for organizations that deal with user financial or medical information or DoD classified information or that go through annual inspections to maintain certification for international operations.

Accounting refers to the recording, logging, and archiving of all server activity, especially activity related to access attempts and whether they were successful. This information should be written into audit logs that are stored and available any time you want or need to view them. The audit logs should contain, at minimum, the following information:

- The user's identity
- The date and time of the request
- Whether the request passed authentication and was granted

Any network security system you put into place should store, or archive, these logs for a specified period of time and allow you to determine for how long these archives will be maintained before they start to age out of the system.

Keeping Current

One of the best ways to stay ahead is to not fall behind in the first place. New systems with increasing sophistication are being developed all the time. They can incorporate a more intelligent and autonomous process in the way the system handles a detected threat, a faster and more easily accomplished method for updating threat files, and configuration flexibility that allows for very precise customization of access rules, authentication requirements, user role assignment, and how tightly it can protect specific applications.

Register for newsletters, attend seminars and network security shows, read white papers, and, if needed, contract the services of network security specialists. The point is, you shouldn't go cheap on network security. The price you pay to keep ahead will be far less than the price you pay to recover from a security breach or attack.

13. Conclusion

Preventing network intrusions is no easy task. Like cops on the street—usually outnumbered and underequipped compared to the bad guys—you face an enemy with determination, skill, training, and a frightening array of increasingly sophisticated tools for hacking their way through your best defenses. And no matter how good your defenses are today, it's only a matter of time before a tool is developed that can penetrate them. If you know that ahead of time, you'll be much more inclined to keep a watchful eye for what "they" have and what you can use to defeat them.

Your best weapon is a logical, thoughtful, and nimble approach to network security. You have to be nimble—to evolve and grow with changes in technology, never being content to keep things as they are because "Hey, they're working just fine." Today's "just fine" will be tomorrow's "What the hell happened?"

Stay informed. There is no shortage of information available to you in the form of white papers, seminars, contract security specialists, and online resources, all dealing with various aspects of network security.

Have a good, solid, comprehensive, yet easy-to-understand network security policy in place. The very process of developing one will get all involved parties thinking about how to best secure your network while addressing user needs. When it comes to your users, you simply can't overeducate them where network security awareness is concerned. The more they know, the better equipped they'll be to act as allies against, rather than accomplices of, the hoards of crackers looking to steal, damage, hobble, or completely cripple your network.

Do your research and invest in good, multipurpose network security systems. Select systems that are easy to install and implement, are adaptable and quickly configurable, can be customized to suit your needs of today as well as tomorrow, and are supported by companies that keep pace with current trends in cracker technology.

References

[1] Internet Threats Report and Predictions for 2009. Secure Computing Corporation; October 27, 2008.
[2] http://news.bbc.co.uk; October 29, 2008.
[3] Quoted from http://news.theage.com.au; October 15, 2008.
[4] Quoted from USA Today March 17, 2008.

CHAPTER 4

Guarding Against Network Intrusions

Tom Chen,[*] Patrick J. Walsh[†]
[*]Swansea University, [†]eSoft Inc.

Virtually all computers today are connected to the Internet through dialup, broadband, Ethernet, or wireless technologies. The reason for this Internet ubiquity is simple: Applications depending on the network, such as email, Web, remote login, instant messaging, and VoIP, have become essential to the computing experience. Unfortunately, the Internet exposes computer users to risks from a wide variety of possible attacks. Users have much to lose—their privacy, valuable data, control of their computers, and possibly theft of their identities. The network enables attacks to be carried out remotely, with relative anonymity and low risk of traceability.

The nature of network intrusions has evolved over the years. A few years ago, a major concern was fast worms such as Code Red, Nimda, Slammer, and Sobig. More recently, concerns shifted to spyware, Trojan horses, and botnets. Although these other threats still continue to be major problems, the Web has become the primary vector for stealthy attacks today [1].

1. Traditional Reconnaissance and Attacks

Traditionally, attack methods follow sequential steps analogous to physical attacks, as shown in Figure 4.1: reconnaissance, compromise, and cover-up [2]. Here we are only addressing attacks directed at a specific target host. Some other types of attacks, such as worms, are not directed at specific targets. Instead, they attempt to hit as many targets as quickly as possible without caring who or what the targets are.

In the first step of a directed attack, the attacker performs reconnaissance to learn as much as possible about the chosen target before carrying out an actual attack. A thorough reconnaissance can lead to a more effective attack because the target's weaknesses can be discovered. One might expect the reconnaissance phase to possibly tip off the target about an

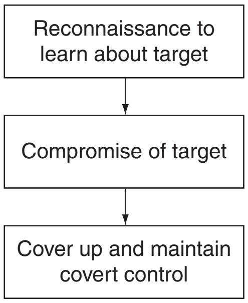

Reconnaissance to learn about target
Compromise of target
Cover up and maintain covert control

Figure 4.1: Steps in directed attacks.

impending attack, but scans and probes are going on constantly in the "background noise" of network traffic, so systems administrators might ignore attack probes as too troublesome to investigate.

Through pings and traceroutes, an attacker can discover IP addresses and map the network around the target. Pings are ICMP echo request and echo reply messages that verify a host's IP address and availability. Traceroute is a network mapping utility that takes advantage of the time to live (TTL) field in IP packets. It sends out packets with TTL = 1, then TTL = 2, and so on. When the packets expire, the routers along the packets' path report that the packets have been discarded, returning ICMP "time exceeded" messages and thereby allowing the traceroute utility to learn the IP addresses of routers at a distance of one hop, two hops, and so on.

Port scans can reveal open ports. Normally, a host might be expected to have certain well-known ports open, such as TCP port 80 (HTTP), TCP port 21 (FTP), TCP port 23 (Telnet), or TCP port 25 (SMTP). A host might also happen to have open ports in the higher range. For example, port 12345 is the default port used by the Netbus remote access Trojan horse, or port 31337 is the default port used by the Back Orifice remote access Trojan horse. Discovery of ports indicating previous malware infections could obviously help an attacker considerably.

In addition to discovering open ports, the popular NMAP scanner (www.insecure.org/nmap) can discover the operating system running on a target. NMAP uses a large set of heuristic rules to identify an operating system based on a target's responses to carefully crafted TCP/IP probes. The basic idea is that different operating systems will make different responses to probes to open TCP/UDP ports and malformed TCP/IP packets. Knowledge of a target's operating system can help an attacker identify vulnerabilities and find effective exploits.

Vulnerability scanning tests a target for the presence of vulnerabilities. Vulnerability scanners such as SATAN, SARA, SAINT, and Nessus typically contain a database of known vulnerabilities that is used to craft probes to a chosen target. The popular Nessus tool (www.nessus.org) has an extensible plug-in architecture to add checks for backdoors, misconfiguration errors, default accounts and passwords, and other types of vulnerabilities.

In the second step of a directed attack, the attacker attempts to compromise the target through one or more methods. Password attacks are common because passwords might be based on common words or names and are guessable by a dictionary attack, although computer systems today have better password policies that forbid easily guessable passwords. If an attacker can obtain the password file from the target, numerous password-cracking tools are available to carry out a brute-force password attack. In addition, computers and networking equipment often ship with default accounts and passwords intended to help systems administrators set up the equipment. These default accounts and passwords are easy to find on the Web (e.g,, www.phenoelit-us.org/dpl/dpl.html). Occasionally users might neglect to change or delete the default accounts, offering intruders an easy way to access the target.

Another common attack method is an exploit attack code written to take advantage of a specific vulnerability [3]. Many types of software, including operating systems and applications, have vulnerabilities. In the second half of 2007, Symantec observed an average of 11.7 vulnerabilities per day [4]. Vulnerabilities are published by several organizations such as CERT and MITRE as well as vendors such as Microsoft through security bulletins. MITRE maintains a database of publicly known vulnerabilities identified by common vulnerabilities and exposures (CVE) numbers. The severity of vulnerabilities is reflected in the industry-standard common vulnerability scoring system (CVSS). In the second half of 2007, Symantec observed that 3% of vulnerabilities were highly severe, 61% were medium-severe, and 36% were low-severe [5]. Furthermore, 73% of vulnerabilities were easily exploitable. For 2007, Microsoft reported that 32% of known vulnerabilities in Microsoft products had publicly available exploit code [6]. Microsoft released 69 security bulletins covering 100 unique vulnerabilities.

Historically, buffer overflows have been the most common type of vulnerability [7]. They have been popular because buffer overflow exploits can often be carried out remotely and lead to complete compromise of a target. The problem arises when a program has allocated a fixed amount of memory space (such as in the stack) for storing data but receives more data than expected. If the vulnerability exists, the extra data will overwrite adjacent parts of memory, which could mess up other variables or pointers. If the extra data is random, the computer might crash or act unpredictably. However, if an attacker crafts the extra data carefully, the buffer overflow could overwrite adjacent memory with a consequence that benefits the attacker. For instance, an attacker might overwrite the return pointer in a stack, causing the program control to jump to malicious code inserted by the attacker.

An effective buffer overflow exploit requires technical knowledge of the computer architecture and operating system, but once the exploit code is written, it can be reused again. Buffer overflows can be prevented by the programmer or compiler performing bounds checking or during runtime. Although C/C + + has received a good deal of blame as a programming language for not having built-in checking that data written to arrays stays within bounds, buffer overflow vulnerabilities appear in a wide variety of other programs, too.

Structured Query Language injection is a type of vulnerability relevant to Web servers with a database backend [8]. SQL is an internationally standardized interactive and programming language for querying data and managing databases. Many commercial database products support SQL, sometimes with proprietary extensions. Web applications often take user input (usually from a Web form) and pass the input into an SQL statement. An SQL injection vulnerability can arise if user input is not properly filtered for string literal escape characters, which can allow an attacker to craft input that is interpreted as embedded SQL statements and thereby manipulate the application running on the database.

Servers have been attacked and compromised by toolkits designed to automate customized attacks. For example, the MPack toolkit emerged in early 2007 and is sold commercially in Russia, along with technical support and regular software updates. It is loaded into a malicious or compromised Web site. When a visitor goes to the site, a malicious code is launched through an iframe (inline frame) within the HTML code. It can launch various exploits, expandable through modules, for vulnerabilities in Web browsers and client software.

Metasploit (www.metasploit.com) is a popular Perl-based tool for developing and using exploits with an easy-to-use Web or command-line interface. Different exploits can be written and loaded into Metasploit and then directed at a chosen target. Exploits can be bundled with a payload (the code to run on a compromised target) selected from a collection of payloads. The tool also contains utilities to experiment with new vulnerabilities and help automate the development of new exploits.

Although exploits are commonplace, not all attacks require an exploit. *Social engineering* refers to types of attacks that take advantage of human nature to compromise a target, typically through deceit. A common social engineering attack is *phishing,* used in identity theft [9]. Phishing starts with a lure, usually a spam message that appears to be from a legitimate bank or ecommerce business. The message attempts to provoke the reader into visiting a fraudulent Web site pretending to be a legitimate business. These fraudulent sites are often set up by automated phishing toolkits that spoof legitimate sites of various brands, including the graphics of those brands. The fraudulent site might even have links to the legitimate Web site, to appear more valid. Victims are thus tricked into submitting valuable personal information such as account numbers, passwords, and Social Security numbers.

Other common examples of social engineering are spam messages that entice the reader into opening an email attachment. Most people know by now that attachments could be dangerous, perhaps containing a virus or spyware, even if they appear to be innocent at first glance. But if the message is sufficiently convincing, such as appearing to originate from an acquaintance, even wary users might be tricked into opening an attachment. Social engineering attacks can be simple but effective because they target people and bypass technological defenses.

The third step of traditional directed attacks involves cover-up of evidence of the compromise and establishment of covert control. After a successful attack, intruders want to maintain remote control and evade detection. Remote control can be maintained if the attacker has managed to install any of a number types of malicious software: a backdoor such as Netcat; a remote access Trojan such as BO2K or SubSeven; or a bot, usually listening for remote instructions on an Internet relay chat (IRC) channel, such as phatbot.

Intruders obviously prefer to evade detection after a successful compromise, because detection will lead the victim to take remedial actions to harden or disinfect the target. Intruders might change the system logs on the target, which will likely contain evidence of their attack. In Windows, the main event logs are secevent.evt, sysevent.evt, and appevent. evt. A systems administrator looking for evidence of intrusions would look in these files with the built-in Windows Event Viewer or a third-party log viewer. An intelligent intruder would not delete the logs but would selectively delete information in the logs to hide signs of malicious actions.

A *rootkit* is a stealthy type of malicious software (*malware*) designed to hide the existence of certain processes or programs from normal methods of detection [10]. Rootkits essentially alter the target's operating system, perhaps by changing drivers or dynamic link libraries (DLLs) and possibly at the kernel level. An example is the kernel-mode FU rootkit that manipulates kernel memory in Windows 2000, XP, and 2003. It consists of a device driver, msdirectx.sys, that might be mistaken for Microsoft's DirectX tool. The rootkit can hide certain events and processes and change the privileges of running processes.

If an intruder has installed malware for covert control, he will want to conceal the communications between himself and the compromised target from discovery by network-based *intrusion detection systems* (IDSs). Intrusion detection systems are designed to listen to network traffic and look for signs of suspicious activities. Several concealment methods are used in practice. *Tunneling* is a commonly used method to place packets of one protocol into the payload of another packet. The "exterior" packet serves a vehicle to carry and deliver the "interior" packet intact. Though the protocol of the exterior packet is easily understood by an IDS, the interior protocol can be any number of possibilities and hence difficult to interpret.

Encryption is another obvious concealment method. Encryption relies on the secrecy of an encryption key shared between the intruder and the compromised target. The encryption key is used to mathematically scramble the communications into a form that is unreadable without the key to decrypt it. Encryption ensures secrecy in practical terms but does not guarantee perfect security. Encryption keys can be guessed, but the time to guess the correct key increases exponentially with the key length. Long keys combined with an algorithm for periodically changing keys can ensure that encrypted communications will be difficult to break within a reasonable time.

Fragmentation of IP packets is another means to conceal the contents of messages from IDSs, which often do not bother to reassemble fragments. IP packets may normally be fragmented into smaller packets anywhere along a route and reassembled at the destination. An IDS can become confused with a flood of fragments, bogus fragments, or deliberately overlapping fragments.

2. Malicious Software

Malicious software, or malware, continues to be an enormous problem for Internet users because of its variety and prevalence and the level of danger it presents [11–13]. It is important to realize that malware can take many forms. A large class of malware is *infectious*, which includes viruses and worms. Viruses and worms are self-replicating, meaning that they spread from host to host by making copies of themselves. Viruses are pieces of code attached to a normal file or program. When the program is run, the virus code is executed and copies itself to (or infects) another file or program. It is often said that viruses need a human action to spread, whereas worms are standalone automated programs. Worms look for vulnerable targets across the network and transfer a copy of themselves if a target is successfully compromised.

Historically, several worms have become well known and stimulated concerns over the possibility of a fast epidemic infecting Internet-connected hosts before defenses could stop it. The 1988 Robert Morris Jr. worm infected thousands of Unix hosts, at the time a significant portion of the Arpanet (the predecessor to the Internet). The 1999 Melissa worm infected Microsoft Word documents and emailed itself to addresses found in a victim's Outlook address book. Melissa demonstrated that email could be a very effective vector for malware distribution, and many subsequent worms have continued to use email, such as the 2000 Love Letter worm. In the 2001–04 interval, several fast worms appeared, notably Code Red, Nimda, Klez, SQL Slammer/Sapphire, Blaster, Sobig, and MyDoom.

An important feature of viruses and worms is their capability to carry a *payload*—malicious code that is executed on a compromised host. The payload can be virtually anything. For instance, SQL Slammer/Sapphire had no payload, whereas Code Red carried an agent to perform a denial-of-service (DoS) attack on certain fixed addresses. The Chernobyl or CIH virus had one of the most destructive payloads, attempting to overwrite critical system files and the system BIOS that is needed for a computer to boot up. Worms are sometimes used to deliver other malware, such as bots, in their payload. They are popular delivery vehicles because of their ability to spread by themselves and carry anything in their payload.

Members of a second large class of malware are characterized by attempts to conceal themselves. This class includes Trojan horses and rootkits. Worms are not particularly stealthy (unless they are designed to be), because they are typically indiscriminate in their attacks. They probe potential targets in the hope of compromising many targets quickly. Indeed, fast-spreading worms are relatively easy to detect because of the network congestion caused by their probes.

Stealth is an important feature for malware because the critical problem for antivirus software is obviously detection of malware. Trojan horses are a type of malware that appears to perform a useful function but hides a malicious function. Thus, the presence of the Trojan horse might not be concealed, but functionality is not fully revealed. For example, a video codec could offer to play certain types of video but also covertly steal the user's data in the background. In the second half of 2007, Microsoft reported a dramatic increase of 300% in the number of Trojan downloaders and droppers, small programs to facilitate downloading more malware later [4].

Rootkits are essentially modifications to the operating system to hide the presence of files or processes from normal means of detection. Rootkits are often installed as drivers or kernel modules. A highly publicized example was the extended copy protection (XCP) software included in some Sony BMG audio CDs in 2005, to prevent music copying. The software was installed automatically on Windows PCs when a CD was played. Made by a company called First 4 Internet, XCP unfortunately contained a hidden rootkit component that patched the operating system to prevent it from displaying any processes, Registry entries, or files with names beginning with *sys*. Although the intention of XCP was not malicious, there was concern that the rootkit could be used by malware writers to conceal malware.

A third important class of malware is designed for remote control. This class includes remote access Trojans (RATs) and bots. Instead of *remote access Trojan*, RAT is sometimes interpreted as *remote administration tool* because it can be used for legitimate purposes by systems administrators. Either way, RAT refers to a type of software usually consisting of server and client parts designed to enable covert communications with a remote controller. The client part is installed on a victim host and mainly listens for instructions from the server part, located at the controller. Notorious examples include Back Orifice, Netbus, and Sub7.

Bots are remote-control programs installed covertly on innocent hosts [14]. Bots are typically programmed to listen to IRC channels for instructions from a "bot herder." All bots under control of the same bot herder form a botnet. Botnets have been known to be rented out for purposes of sending spam or launching a distributed DoS (DDoS) attack [15]. The power of a botnet is proportional to its size, but exact sizes have been difficult to discover.

One of the most publicized bots is the Storm worm, which has various aliases. Storm was launched in January 2007 as spam with a Trojan horse attachment. As a botnet, Storm has shown unusual resilience by working in a distributed peer-to-peer manner without centralized control. Each compromised host connects to a small subset of the entire botnet. Each infected host shares lists of other infected hosts, but no single host has a full list of the entire botnet. The size of the Storm botnet has been estimated at more than 1 million compromised hosts, but an exact size has been impossible to determine because of the many bot variants and active measures to avoid detection. Its creators have been persistent in continually updating its lures with current events and evolving tactics to spread and avoid detection.

Another major class of malware is designed for data theft. This class includes keyloggers and spyware. A keylogger can be a Trojan horse or other form of malware. It is designed to record a user's keystrokes and perhaps report them to a remote attacker. Keyloggers are planted by criminals on unsuspecting hosts to steal passwords and other valuable personal information. It has also been rumored that the Federal Bureau of Investigation (FBI) has used a keylogger called Magic Lantern.

As the name implies, *spyware* is stealthy software designed to monitor and report user activities for the purposes of learning personal information without the user's knowledge or consent. Surveys have found that spyware is widely prevalent on consumer PCs, usually without knowledge of the owners. Adware is viewed by some as a mildly objectionable form of spyware that spies on Web browsing behavior to target online advertisements to a user's apparent interests. More objectionable forms of spyware are more invasive of privacy and raise other objections related to stealthy installation, interference with normal Web browsing, and difficulty of removal.

Spyware can be installed in a number of stealthy ways: disguised as a Trojan horse, bundled with a legitimate software program, delivered in the payload of a worm or virus, or downloaded through deception. For instance, a deceptive Web site might pop up a window appearing to be a standard Windows dialog box, but clicking any button will cause spyware to be downloaded. Another issue is that spyware might or might not display an end-user license agreement (EULA) before installation. If an EULA is displayed, the mention of spyware is typically unnoticeable or difficult to find.

More pernicious forms of spyware can change computer settings, reset homepages, and redirect the browser to unwanted sites. For example, the notorious CoolWebSearch changed homepages to Coolwebsearch.com, rewrote search engine results, and altered host files, and some variants added links to pornographic and gambling sites to the browser's bookmarks.

Lures and "Pull" Attacks

Traditional network attacks can be viewed as an "active" approach in which the attacker takes the initiative of a series of actions directed at a target. Attackers face the risk of revealing their malicious intentions through these actions. For instance, port scanning, password guessing, or exploit attempts can be readily detected by an IDS as suspicious activities. Sending malware through email can only be seen as an attack attempt.

Security researchers have observed a trend away from direct attacks toward more stealthy attacks that wait for victims to visit malicious Web sites, as shown in Figure 4.2 [16]. The Web has become the primary vector for infecting computers, in large part because email has become better secured. Sophos discovers a new malicious Webpage every 14 seconds, on average [17].

Web-based attacks have significant advantages for attackers. First, they are stealthier and not as "noisy" as active attacks, making it easier to continue undetected for a longer time. Second, Web servers have the intelligence to be stealthy. For instance, Web servers have been found that serve up an attack only once per IP address, and otherwise serve up legitimate content. The malicious server remembers the IP addresses of visitors. Thus, a visitor will be attacked only once, which makes the attack harder to detect. Third, a Web server can serve up different attacks, depending on the visitor's operating system and browser.

As mentioned earlier, a common type of attack carried out through the Web is phishing. A phishing site is typically disguised as a legitimate financial organization or ecommerce business. During the month of December 2007, the Anti-Phishing Working Group found 25,328 new unique phishing sites hijacking 144 brands (www.antiphishing.org).

Another type of Web-based attack is a malicious site that attempts to download malware through a visitor's browser, called a *drive-by download*. A Web page usually loads a malicious script by means of an iframe (inline frame). It has been reported that most drive-by

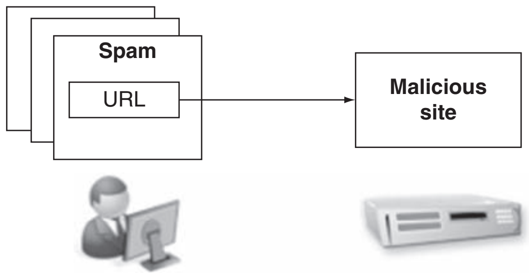


Figure 4.2: Stealthy attacks lure victims to malicious servers.

downloads are hosted on legitimate sites that have been compromised. For example, in June 2007 more than 10,000 legitimate Italian Web sites were discovered to be compromised with malicious code loaded through iframes. Many other legitimate sites are regularly compromised.

Drive-by downloading through a legitimate site holds certain appeal for attackers. First, most users would be reluctant to visit suspicious and potentially malicious sites but will not hesitate to visit legitimate sites in the belief that they are always safe. Even wary Web surfers may be caught off-guard. Second, the vast majority of Web servers run Apache (approximately 50%) or Microsoft IIS (approximately 40%), both of which have vulnerabilities that can be exploited by attackers. Moreover, servers with database applications could be vulnerable to SQL injection attacks. Third, if a legitimate site is compromised with an iframe, the malicious code might go unnoticed by the site owner for some time.

Pull-based attacks pose one challenge to attackers: They must attract visitors to the malicious site somehow while avoiding detection by security researchers. One obvious option is to send out lures in spam. Lures have been disguised as email from the Internal Revenue Service, a security update from Microsoft, or a greeting card. The email attempts to entice the reader to visit a link. On one hand, lures are easier to get through spam filters because they only contain links and not attachments. It is easier for spam filters to detect malware attachments than to determine whether links in email are malicious. On the other hand, spam filters are easily capable of extracting and following links from spam. The greater challenge is to determine whether the linked site is malicious.

3. Defense in Depth

Most security experts would agree with the view that perfect network security is impossible to achieve and that any single defense can always be overcome by an attacker with sufficient resources and motivation. The basic idea behind the *defense-in-depth strategy* is to hinder the attacker as much as possible with multiple layers of defense, even though each layer might be surmountable. More valuable assets are protected behind more layers of defense. The combination of multiple layers increases the cost for the attacker to be successful, and the cost is proportional to the value of the protected assets. Moreover, a combination of multiple layers will be more effective against unpredictable attacks than will a single defense optimized for a particular type of attack.

The cost for the attacker could be in terms of additional time, effort, or equipment. For instance, by delaying an attacker, an organization would increase the chances of detecting and reacting to an attack in progress. The increased costs to an attacker could deter some attempts if the costs are believed to outweigh the possible gain from a successful attack.

Defense in depth is sometimes said to involve people, technology, and operations. Trained security people should be responsible for securing facilities and information assurance. However, every computer user in an organization should be made aware of security policies and practices. Every Internet user at home should be aware of safe practices (such as avoiding opening email attachments or clicking suspicious links) and the benefits of appropriate protection (antivirus software, firewalls).

A variety of technological measures can be used for layers of protection. These should include firewalls, IDSs, routers with access control lists (ACLs), antivirus software, access control, spam filters, and so on. These topics are discussed in more depth later.

The term *operations* refers to all preventive and reactive activities required to maintain security. Preventive activities include vulnerability assessments, software patching, system hardening (closing unnecessary ports), and access controls. Reactive activities should detect malicious activities and react by blocking attacks, isolating valuable resources, or tracing the intruder.

Protection of valuable assets can be a more complicated decision than simply considering the value of the assets. Organizations often perform a risk assessment to determine the value of assets, possible threats, likelihood of threats, and possible impact of threats. Valuable assets facing unlikely threats or threats with low impact might not need much protection. Clearly, assets of high value facing likely threats or high-impact threats merit the strongest defenses. Organizations usually have their own risk management process for identifying risks and deciding how to allocate a security budget to protect valuable assets under risk.

4. Preventive Measures

Most computer users are aware that Internet use poses security risks. It would be reasonable to take precautions to minimize exposure to attacks. Fortunately, several options are available to computer users to fortify their systems to reduce risks.

Access Control

In computer security, *access control* refers to mechanisms to allow users to perform functions up to their authorized level and restrict users from performing unauthorized functions [18]. Access control includes:

- Authentication of users
- Authorization of their privileges
- Auditing to monitor and record user actions

All computer users will be familiar with some type of access control.

Authentication is the process of verifying a user's identity. Authentication is typically based on one or more of these factors:

- Something the user knows, such as a password or PIN
- Something the user has, such as a smart card or token
- Something personal about the user, such as a fingerprint, retinal pattern, or other biometric identifier

Use of a single factor, even if multiple pieces of evidence are offered, is considered weak authentication. A combination of two factors, such as a password and a fingerprint, called *two-factor* (or *multifactor*) *authentication,* is considered strong authentication.

Authorization is the process of determining what an authenticated user can do. Most operating systems have an established set of permissions related to read, write, or execute access. For example, an ordinary user might have permission to read a certain file but not write to it, whereas a root or superuser will have full privileges to do anything.

Auditing is necessary to ensure that users are accountable. Computer systems record actions in the system in audit trails and logs. For security purposes, they are invaluable forensic tools to recreate and analyze incidents. For instance, a user attempting numerous failed logins might be seen as an intruder.

Vulnerability Testing and Patching

As mentioned earlier, vulnerabilities are weaknesses in software that might be used to compromise a computer. Vulnerable software includes all types of operating systems and application programs. New vulnerabilities are being discovered constantly in different ways. New vulnerabilities discovered by security researchers are usually reported confidentially to the vendor, which is given time to study the vulnerability and develop a path. Of all vulnerabilities disclosed in 2007, 50% could be corrected through vendor patches [19]. When ready, the vendor will publish the vulnerability, hopefully along with a patch.

It has been argued that publication of vulnerabilities will help attackers. Though this might be true, publication also fosters awareness within the entire community. Systems administrators will be able to evaluate their systems and take appropriate precautions. One might expect systems administrators to know the configuration of computers on their network, but in large organizations, it would be difficult to keep track of possible configuration changes made by users. Vulnerability testing offers a simple way to learn about the configuration of computers on a network.

Vulnerability testing is an exercise to probe systems for known vulnerabilities. It requires a database of known vulnerabilities, a packet generator, and test routines to generate a sequence of packets to test for a particular vulnerability. If a vulnerability is found and a software patch is available, that host should be patched.

Penetration testing is a closely related idea but takes it further. Penetration testing simulates the actions of a hypothetical attacker to attempt to compromise hosts. The goal is, again, to learn about weaknesses in the network so that they can be remedied.

Closing Ports

Transport layer protocols, namely Transmission Control Protocol (TCP) and User Datagram Protocol (UDP), identify applications communicating with each other by means of port numbers. Port numbers 1 to 1023 are well known and assigned by the Internet Assigned Numbers Authority (IANA) to standardized services running with root privileges. For example, Web servers listen on TCP port 80 for client requests. Port numbers 1024 to 49151 are used by various applications with ordinary user privileges. Port numbers above 49151 are used dynamically by applications.

It is good practice to close ports that are unnecessary, because attackers can use open ports, particularly those in the higher range. For instance, the Sub7 Trojan horse is known to use port 27374 by default, and Netbus uses port 12345. Closing ports does not by itself guarantee the safety of a host, however. Some hosts need to keep TCP port 80 open for HyperText Transfer Protocol (HTTP), but attacks can still be carried out through that port.

Firewalls

When most people think of network security, firewalls are one of the first things to come to mind. Firewalls are a means of perimeter security protecting an internal network from external threats. A firewall selectively allows or blocks incoming and outgoing traffic. Firewalls can be standalone network devices located at the entry to a private network or personal firewall programs running on PCs. An organization's firewall protects the internal community; a personal firewall can be customized to an individual's needs.

Firewalls can provide separation and isolation among various network zones, namely the public Internet, private intranets, and a demilitarized zone (DMZ), as shown in Figure 4.3. The semiprotected DMZ typically includes public services provided by a private organization. Public servers need some protection from the public Internet so they usually sit behind a firewall. This firewall cannot be completely restrictive because the public servers must be externally accessible. Another firewall typically sits between the DMZ and private internal network because the internal network needs additional protection.

There are various types of firewalls: packet-filtering firewalls, stateful firewalls, and proxy firewalls. In any case, the effectiveness of a firewall depends on the configuration of its rules. Properly written rules require detailed knowledge of network protocols. Unfortunately, some firewalls are improperly configured through neglect or lack of training.

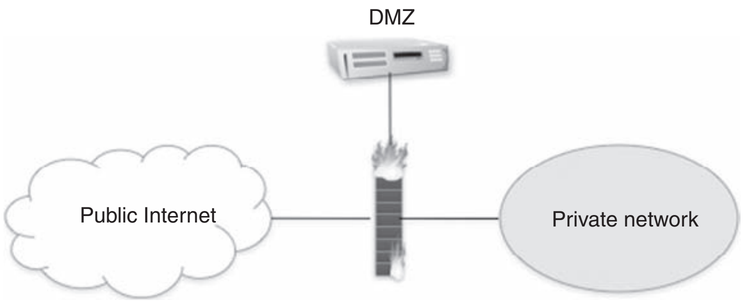


Figure 4.3: A firewall isolating various network zones.

Packet-filtering firewalls analyze packets in both directions and either permit or deny passage based on a set of rules. Rules typically examine port numbers, protocols, IP addresses, and other attributes of packet headers. There is no attempt to relate multiple packets with a flow or stream. The firewall is stateless, retaining no memory of one packet to the next.

Stateful firewalls overcome the limitation of packet-filtering firewalls by recognizing packets belonging to the same flow or connection and keeping track of the connection state. They work at the network layer and recognize the legitimacy of sessions.

Proxy firewalls are also called application-level firewalls because they process up to the application layer. They recognize certain applications and can detect whether an undesirable protocol is using a nonstandard port or an application layer protocol is being abused. They protect an internal network by serving as primary gateways to proxy connections from the internal network to the public Internet. They could have some impact on network performance due to the nature of the analysis.

Firewalls are essential elements of an overall defensive strategy but have the drawback that they only protect the perimeter. They are useless if an intruder has a way to bypass the perimeter. They are also useless against insider threats originating within a private network.

Antivirus and Antispyware Tools

The proliferation of malware prompts the need for antivirus software [20]. Antivirus software is developed to detect the presence of malware, identify its nature, remove the malware (disinfect the host), and protect a host from future infections. Detection should ideally minimize false positives (false alarms) and false negatives (missed malware) at the same time. Antivirus software faces a number of difficult challenges:

- Malware tactics are sophisticated and constantly evolving.
- Even the operating system on infected hosts cannot be trusted.
- Malware can exist entirely in memory without affecting files.

- Malware can attack antivirus processes.
- The processing load for antivirus software cannot degrade computer performance such that users become annoyed and turn the antivirus software off.

One of the simplest tasks performed by antivirus software is file scanning. This process compares the bytes in files with known signatures that are byte patterns indicative of a known malware. It represents the general approach of signature-based detection. When new malware is captured, it is analyzed for unique characteristics that can be described in a signature. The new signature is distributed as updates to antivirus programs. Antivirus looks for the signature during file scanning, and if a match is found, the signature identifies the malware specifically. There are major drawbacks to this method, however: New signatures require time to develop and test; users must keep their signature files up to date; and new malware without a known signature may escape detection.

Behavior-based detection is a complementary approach. Instead of addressing what malware is, behavior-based detection looks at what malware tries to do. In other words, anything attempting a risky action will come under suspicion. This approach overcomes the limitations of signature-based detection and could find new malware without a signature, just from its behavior. However, the approach can be difficult in practice. First, we must define what is suspicious behavior, or conversely, what is normal behavior. This definition often relies on heuristic rules developed by security experts, because normal behavior is difficult to define precisely. Second, it might be possible to *discern* suspicious behavior, but it is much more difficult to *determine* malicious behavior, because malicious intention must be inferred. When behavior-based detection flags suspicious behavior, more follow-up investigation is usually needed to better understand the threat risk.

The ability of malware to change or disguise appearances can defeat file scanning. However, regardless of its form, malware must ultimately perform its mission. Thus, an opportunity will always arise to detect malware from its behavior if it is given a chance to execute. Antivirus software will monitor system events, such as hard-disk access, to look for actions that might pose a threat to the host. Events are monitored by intercepting calls to operating system functions.

Although monitoring system events is a step beyond file scanning, malicious programs are running in the host execution environment and could pose a risk to the host. The idea of emulation is to execute suspected code within an isolated environment, presenting the appearance of the computer resources to the code, and to look for actions symptomatic of malware.

Virtualization takes emulation a step further and executes suspected code within a real operating system. A number of virtual operating systems can run above the host operating system. Malware can corrupt a virtual operating system, but for safety reasons a virtual operating system has limited access to the host operating system. A "sandbox" isolates the virtual environment from tampering with the host environment, unless a specific action is requested and permitted. In contrast, emulation does not offer an operating system to suspected code; the code is allowed to execute step by step, but in a controlled and restricted way, just to discover what it will attempt to do.

Antispyware software can be viewed as a specialized class of antivirus software. Somewhat unlike traditional viruses, spyware can be particularly pernicious in making a vast number of changes throughout the hard drive and system files. Infected systems tend to have a large number of installed spyware programs, possibly including certain cookies (pieces of text planted by Web sites in the browser as a means of keeping them in memory).

Spam Filtering

Every Internet user is familiar with spam email. There is no consensus on an exact definition of spam, but most people would agree that spam is unsolicited, sent in bulk, and commercial in nature. There is also consensus that the vast majority of email is spam. Spam continues to be a problem because a small fraction of recipients do respond to these messages. Even though the fraction is small, the revenue generated is enough to make spam profitable because it costs little to send spam in bulk. In particular, a large botnet can generate an enormous amount of spam quickly.

Users of popular Webmail services such as Yahoo! and Hotmail are attractive targets for spam because their addresses might be easy to guess. In addition, spammers harvest email addresses from various sources: Web sites, newsgroups, online directories, data-stealing viruses, and so on. Spammers might also purchase lists of addresses from companies who are willing to sell customer information.

Spam is more than an inconvenience for users and a waste of network resources. Spam is a popular vehicle to distribute malware and lures to malicious Web sites. It is the first step in phishing attacks.

Spam filters work at an enterprise level and a personal level. At the enterprise level, mail gateways can protect an entire organization by scanning incoming messages for malware and blocking messages from suspicious or fake senders. A concern at the enterprise level is the rate of false positives, which are legitimate messages mistaken for spam. Users may become upset if their legitimate mail is blocked. Fortunately, spam filters are typically customizable, and the rate of false positives can be made very low. Additional spam filtering at the personal level can customize filtering even further, to account for individual preferences.

Various spam-filtering techniques are embodied in many commercial and free spam filters, such as DSPAM and SpamAssassin, to name two. Bayesian filtering is one of the more popular techniques [21]. First, an incoming message is parsed into tokens, which are single words or word combinations from the message's header and body. Second, probabilities are assigned to tokens through a training process. The filter looks at a set of known spam messages compared to a set of known legitimate messages and calculates token probabilities based on Bayes' theorem (from probability theory). Intuitively, a word such as *Viagra* would appear more often in spam, and therefore the appearance of a Viagra token would increase the probability of that message being classified as spam.

The probability calculated for a message is compared to a chosen threshold; if the probability is higher, the message is classified as spam. The threshold is chosen to balance the rates of false positives and false negatives (missed spam) in some desired way. An attractive feature of Bayesian filtering is that its probabilities will adapt to new spam tactics, given continual feedback, that is, correction of false positives and false negatives by the user.

It is easy to see why spammers have attacked Bayesian filters by attempting to influence the probabilities of tokens. For example, spammers have tried filling messages with large amounts of legitimate text (e.g., drawn from classic literature) or random innocuous words. The presence of legitimate tokens tends to decrease a message's score because they are evidence counted toward the legitimacy of the message.

Spammers are continually trying new ways to get through spam filters. At the same time, security companies respond by adapting their technologies.

Honeypots

The basic idea of a *honeypot* is to learn about attacker techniques by attracting attacks to a seemingly vulnerable host [22]. It is essentially a forensics tool rather than a line of defense. A honeypot could be used to gain valuable information about attack methods used elsewhere or imminent attacks before they happen. Honeypots are used routinely in research and production environments.

A honeypot has more special requirements than a regular PC. First, a honeypot should not be used for legitimate services or traffic. Consequently, every activity seen by the honeypot will be illegitimate. Even though honeypots typically record little data compared to IDS, for instance, their data has little "noise," whereas the bulk of IDS data is typically uninteresting from a security point of view.

Second, a honeypot should have comprehensive and reliable capabilities for monitoring and logging all activities. The forensic value of a honeypot depends on the detailed information it can capture about attacks.

Third, a honeypot should be isolated from the real network. Since honeypots are intended to attract attacks, there is a real risk that the honeypot could be compromised and used as a launching pad to attack more hosts in the network.

Honeypots are often classified according to their level of interaction, ranging from low to high. Low-interaction honeypots, such as Honeyd, offer the appearance of simple services. An attacker could try to compromise the honeypot but would not have much to gain. The limited interactions pose a risk that an attacker could discover that the host is a honeypot. At the other end of the range, high-interaction honeypots behave more like real systems. They have more capabilities to interact with an attacker and log activities, but they offer more to gain if they are compromised.

Honeypots are related to the concepts of black holes or network telescopes, which are monitored blocks of unused IP addresses. Since the addresses are unused, any traffic seen at those addresses is naturally suspicious (although not necessarily malicious).

Traditional honeypots suffer a drawback in that they are passive and wait to see malicious activity. The idea of honeypots has been extended to active clients that search for malicious servers and interact with them. The active version of a honeypot has been called a *honeymonkey* or *client honeypot*.

Network Access Control

A vulnerable host might place not only itself but an entire community at risk. For one thing, a vulnerable host might attract attacks. If compromised, the host could be used to launch attacks on other hosts. The compromised host might give information to the attacker, or there might be trust relationships between hosts that could help the attacker. In any case, it is not desirable to have a weakly protected host on your network.

The general idea of *network access control* (NAC) is to restrict a host from accessing a network unless the host can provide evidence of a strong security posture. The NAC process involves the host, the network (usually routers or switches, and servers), and a security policy, as shown in Figure 4.4.

The details of the NAC process vary with various implementations, which unfortunately currently lack standards for interoperability. A host's security posture includes its IP address, operating system, antivirus software, personal firewall, and host intrusion detection system.

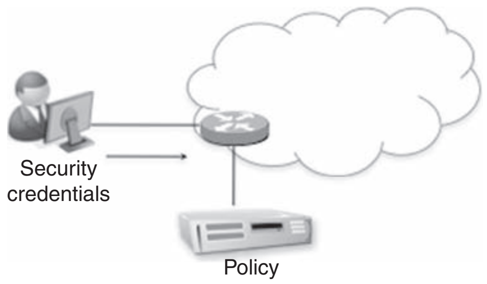


Figure 4.4: Network access control.

In some implementations, a software agent runs on the host, collects information about the host's security posture, and reports it to the network as part of a request for admission to the network. The network refers to a policy server to compare the host's security posture to the security policy, to make an admission decision.

The admission decision could be anything from rejection to partial admission or full admission. Rejection might be prompted by out-of-date antivirus software, an operating system needing patches, or firewall misconfiguration. Rejection might lead to quarantine (routing to an isolated network) or forced remediation.

5. Intrusion Monitoring and Detection

Preventive measures are necessary and help reduce the risk of attacks, but it is practically impossible to prevent all attacks. Intrusion detection is also necessary to detect and diagnose malicious activities, analogous to a burglar alarm. Intrusion detection is essentially a combination of monitoring, analysis, and response [23]. Typically an IDS supports a console for human interface and display. Monitoring and analysis are usually viewed as passive techniques because they do not interfere with ongoing activities. The typical IDS response is an alert to systems administrators, who might choose to pursue further investigation or not. In other words, traditional IDSs do not offer much response beyond alerts, under the presumption that security incidents need human expertise and judgment for follow-up.

Detection accuracy is the critical problem for intrusion detection. Intrusion detection should ideally minimize false positives (normal incidents mistaken for suspicious ones) and false negatives (malicious incidents escaping detection). Naturally, false negatives are contrary to the essential purpose of intrusion detection. False positives are also harmful because they are troublesome for systems administrators who must waste time investigating false alarms. Intrusion detection should also seek to more than identify security incidents. In addition to relating the facts of an incident, intrusion detection should ascertain the nature of the incident, the perpetrator, the seriousness (malicious vs. suspicious), scope, and potential consequences (such as stepping from one target to more targets).

IDS approaches can be categorized in at least two ways. One way is to differentiate host-based and network-based IDS, depending on where sensing is done. A host-based IDS monitors an individual host, whereas a network-based IDS works on network packets. Another way to view IDS is by their approach to analysis. Traditionally, the two analysis approaches are misuse (signature-based) detection and anomaly (behavior-based) detection. As shown in Figure 4.5, these two views are complementary and are often used in combination.

In practice, intrusion detection faces several difficult challenges: signature-based detection can recognize only incidents matching a known signature; behavior-based detection relies on an understanding of normal behavior, but "normal" can vary widely. Attackers are intelligent

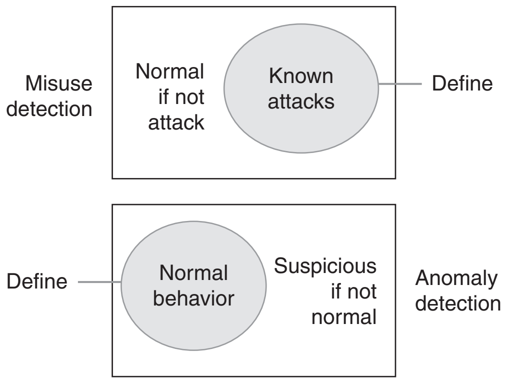


Figure 4.5: Misuse detection and anomaly detection.

and evasive; attackers might try to confuse IDS with fragmented, encrypted, tunneled, or junk packets; an IDS might not react to an incident in real time or quickly enough to stop an attack; and incidents can occur anywhere at any time, which necessitates continual and extensive monitoring, with correlation of multiple distributed sensors.

Host-Based Monitoring

Host-based IDS runs on a host and monitors system activities for signs of suspicious behavior. Examples could be changes to the system Registry, repeated failed login attempts, or installation of a backdoor. Host-based IDSs usually monitor system objects, processes, and regions of memory. For each system object, the IDS will usually keep track of attributes such as permissions, size, modification dates, and hashed contents, to recognize changes.

A concern for a host-based IDS is possible tampering by an attacker. If an attacker gains control of a system, the IDS cannot be trusted. Hence, special protection of the IDS against tampering should be architected into a host.

A host-based IDS is not a complete solution by itself. Though monitoring the host is logical, it has three significant drawbacks: visibility is limited to a single host; the IDS process consumes resources, possibly impacting performance on the host; and attacks will not be seen until they have already reached the host. Host-based and network-based IDS are often used together to combine strengths.

Traffic Monitoring

Network-based IDSs typically monitor network packets for signs of reconnaissance, exploits, DoS attacks, and malware. They have strengths to complement host-based IDSs: network-based IDSs can see traffic for a population of hosts; they can recognize patterns shared by multiple hosts; and they have the potential to see attacks before they reach the hosts.

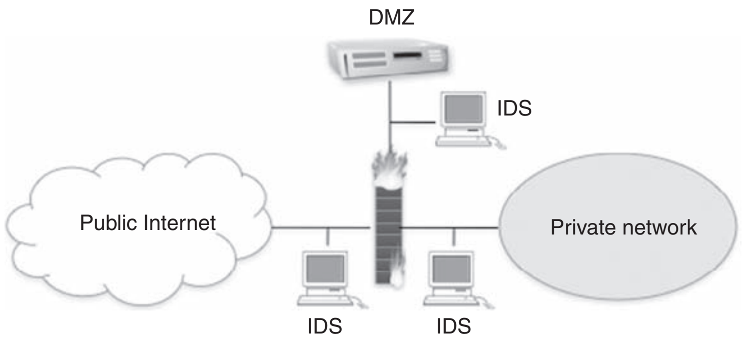


Figure 4.6: IDSs monitoring various network zones.

IDSs are placed in various locations for different views, as shown in Figure 4.6. An IDS outside a firewall is useful for learning about malicious activities on the Internet. An IDS in the DMZ will see attacks originating from the Internet that are able to get through the outer firewall to public servers. Lastly, an IDS in the private network is necessary to detect any attacks that are able to successfully penetrate perimeter security.

Signature-Based Detection

Signature-based intrusion detection depends on patterns that uniquely identify an attack. If an incident matches a known signature, the signature identifies the specific attack. The central issue is how to define signatures or model attacks. If signatures are too specific, a change in an attack tactic could result in a false negative (missed alarm). An attack signature should be broad enough to cover an entire class of attacks. On the other hand, if signatures are too general, it can result in false positives.

Signature-based approaches have three inherent drawbacks: new attacks can be missed if a matching signature is not known; signatures require time to develop for new attacks; and new signatures must be distributed continually.

Snort is a popular example of a signature-based IDS (www.snort.org). Snort signatures are rules that define fields that match packets of information about the represented attack. Snort is packaged with more than 1800 rules covering a broad range of attacks, and new rules are constantly being written.

Behavior Anomalies

A behavior-based IDS is appealing for its potential to recognize new attacks without a known signature. It presumes that attacks will be different from normal behavior. Hence the critical issue is how to define normal behavior, and anything outside of normal (anomalous) is classified as suspicious. A common approach is to define normal behavior in statistical terms, which allows for deviations within a range.

Behavior-based approaches have considerable challenges. First, normal behavior is based on past behavior. Thus, data about past behavior must be available for training the IDS. Second, behavior can and does change over time, so any IDS approach must be adaptive. Third, anomalies are just unusual events, not necessarily malicious ones. A behavior-based IDS might point out incidents to investigate further, but it is not good at discerning the exact nature of attacks.

Intrusion Prevention Systems

IDSs are passive techniques. They typically notify the systems administrator to investigate further and take the appropriate action. The response might be slow if the systems administrator is busy or the incident is time consuming to investigate.

A variation called an *intrusion prevention system* (IPS) seeks to combine the traditional monitoring and analysis functions of an IDS with more active automated responses, such as automatically reconfiguring firewalls to block an attack. An IPS aims for a faster response than humans can achieve, but its accuracy depends on the same techniques as the traditional IDS. The response should not harm legitimate traffic, so accuracy is critical.

6. Reactive Measures

When an attack is detected and analyzed, systems administrators must exercise an appropriate response to the attack. One of the principles in security is that the response should be proportional to the threat. Obviously, the response will depend on the circumstances, but various options are available. Generally, it is possible to block, slow, modify, or redirect any malicious traffic. It is not possible to delineate every possible response. Here we describe only two responses: quarantine and traceback.

Quarantine

Dynamic quarantine in computer security is analogous to quarantine for infectious diseases. It is an appropriate response, particularly in the context of malware, to prevent an infected host from contaminating other hosts. Infectious malware requires connectivity between an infected host and a new target, so it is logical to disrupt the connectivity between hosts or networks as a means to impede the malware from spreading further.

Within the network, traffic can be blocked by firewalls or routers with access control lists. ACLs are similar to firewall rules, allowing routers to selectively drop packets.

Traceback

One of the critical aspects of an attack is the identity or location of the perpetrator. Unfortunately, discovery of an attacker in IP networks is almost impossible because:

- The source address in IP packets can be easily spoofed (forged).
- Routers are stateless by design and do not keep records of forwarded packets.
- Attackers can use a series of intermediary hosts (called *stepping stones* or *zombies*) to carry out their attacks.

Intermediaries are usually innocent computers taken over by an exploit or malware and put under control of the attacker. In practice, it might be possible to trace an attack back to the closest intermediary, but it might be too much to expect to trace an attack all the way back to the real attacker.

To trace a packet's route, some tracking information must be either stored at routers when the packet is forwarded or carried in the packet, as shown in Figure 4.7. An example of the first approach is to store a hash of a packet for some amount of time. If an attack occurs, the target host will query routers for a hash of the attack packet. If a router has the hash, it is evidence that the packet had been forwarded by that router. To reduce memory consumption, the hash is stored instead of storing the entire packet. The storage is temporary instead of permanent so that routers will not run out of memory.

An example of the second approach is to stamp packets with a unique router identifier, such as an IP address. Thus the packet carries a record of its route. The main advantage here is that routers can remain stateless. The problem is that there is no space in the IP packet header for this scheme.

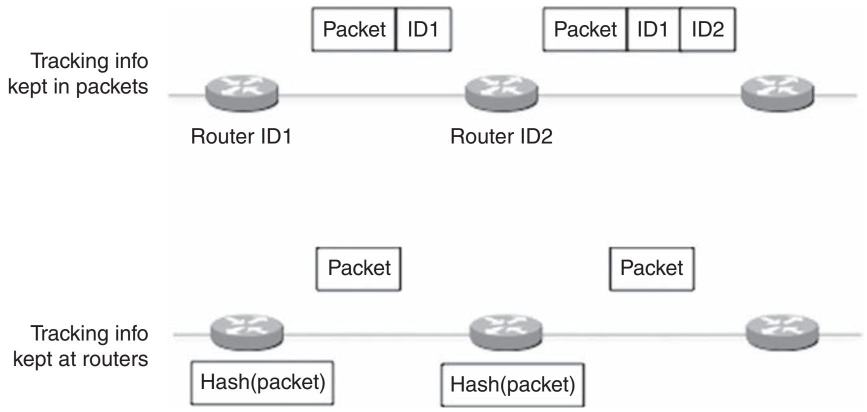


Figure 4.7: Tracking information stored at routers or carried in packets to enable packet traceback.

7. Conclusions

To guard against network intrusions, we must understand the variety of attacks, from exploits to malware to social engineering. Direct attacks are prevalent, but a class of *pull attacks* has emerged, relying on lures to bring victims to a malicious Web site. Pull attacks are much more difficult to uncover and in a way defend against. Just about anyone can become victimized.

Much can be done to fortify hosts and reduce their risk exposure, but some attacks are unavoidable. Defense in depth is a most practical defense strategy, combining layers of defenses. Although each defensive layer is imperfect, the cost becomes harder to surmount for intruders.

One of the essential defenses is *intrusion detection*. Host-based and network-based intrusion detection systems have their respective strengths and weaknesses. Research continues to be needed to improve intrusion detection, particularly behavior-based techniques. As more attacks are invented, signature-based techniques will have more difficulty keeping up.

References

[1] Turner D, et al. Symantec Global Internet Security Threat Report: Trends for July–December 2007, available at www.symantec.com (date of access: July, 1, 2008).

[2] Skoudis E. Counter Hack Reloaded: A Step-by-Step Guide to Computer Attacks and Effective Defenses. 2nd ed. Prentice Hall; 2006.

[3] McClure S, Scambray J, Kutz G. Hacking Exposed. 3rd ed. McGraw-Hill; 2001.

[4] Turner D, et al. Symantec Global Internet Security Threat Report: Trends for July–December 2007, available at www.symantec.com (date of access: July, 1, 2008).

[5] Turner D, et al. Symantec Global Internet Security Threat Report: Trends for July–December 2007, available at www.symantec.com (date of access: July, 1, 2008).

[6] Arsenault B, Gullutto V. Microsoft Security Intelligence Report: July–December 2007, available at www.microsoft.com (date of access: July 1, 2008).

[7] Foster J, Osipov V, Bhalla N. Buffer Overflow Attacks: Detect, Exploit, Prevent. Syngress; 2005.

[8] Litchfield D. SQL Server Security. McGraw-Hill Osborne; 2003.

[9] Jakobsson M, Meyers S, editors. Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft. Wiley-Interscience; 2006.

[10] Hoglund G, Butler J. Rootkits: Subverting the Windows Kernel. Addison-Wesley Professional; 2005.

[11] Harley D, Slade D. Viruses Revealed. McGraw-Hill; 2001.

[12] Skoudis E. Malware: Fighting Malicious Code. Prentice Hall PTR; 2004.

[13] Szor P. The Art of Computer Virus Research and Defense. Addison-Wesley; 2005.

[14] Schiller C, et al. Botnets: the Killer Web App. Syngress Publishing; 2007.

[15] Dittrich D. Distributed denial of service (DDoS) attacks/tools, Available at http://staff.washington.edu/dittrich/misc/ddos/ (date of access: July 1, 2008).

[16] Scambray J, Shema M, Sima C. Hacking Exposed Web Applications. 2nd ed. McGraw-Hill; 2006.

[17] Sophos. Security Threat Report 2008, Available at http://research.sophos.com/sophosthreatreport08 (date of access: July 1, 2008).

[18] Carroll B. Cisco Access Control Security: AAA Administration Services. Cisco Press; 2004.
[19] IBM Internet Security Systems. X-Force 2007 Trend Statistics; January 2008 (date of access: July 1, 2008).
[20] Szor P. The Art of Computer Virus Research and Defense. Addison-Wesley; 2005.
[21] Zdziarski J. Ending Spam: Bayesian Content Filtering and the Art of Statistical Language Classification. No Starch Press; 2005.
[22] The Honeynet Project. Know Your Enemy. Learning About Security Threats. 2nd ed. Addison-Wesley; 2004.
[23] Bejtlich R. The Tao of Network Security Monitoring: Beyond Intrusion Detection. Addison-Wesley; 2005.

This page intentionally left blank

CHAPTER 5

Unix and Linux Security

Gerald Beuchelt
Independent Security Consultant

When Unix was first booted on a PDP-8 computer at Bell Labs, it already had a basic notion of user isolation, separation of kernel and user memory space, and pro-cess security. It was originally conceived as a multiuser system, and as such, security could not be added on as an afterthought. In this respect, Unix was different from a whole class of computing machinery that had been targeted at single-user environments.

1. Unix and Security

The examples in this chapter refer to the Solaris operating system and Debian-based Linux distributions, a commercial and a community developed operating system.

Solaris is freely available in open source and binary distributions. It derives directly from AT&T System V R4.2 and is one of the few operating systems that can legally be called Unix. It is distributed by Sun Microsystems, but there are independent distributions built on top of the open source version of Solaris.

The Aims of System Security

Linux is mostly a GNU software-based operating system with a kernel originally written by Linus Torvalds. Debian is a distribution originally developed by Ian Murdock of Purdue University. Debian's express goal is to use only open and free software, as defined by its guidelines.

Authentication

When a user is granted access to resources on a computing system, it is of vital importance to verify that he was granted permission for access. This process—establishing the identity of the user—is commonly referred to as *authentication* (sometimes abbreviated *AuthN*).

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00005-1

Authorization

As we mentioned, Unix was originally devised as a multiuser system. To protect user data from other users and nonusers, the operating system has to put up safeguards against unauthorized access. Determining the eligibility of an authenticated (or anonymous) user to access a resource is usually called *authorization* (*AuthZ*).

Availability

Guarding a system (including all its subsystems, such as the network) against security breaches is vital to keep the system available for its intended use. *Availability* of a system must be properly defined: Any system is physically available, even if it is turned off—however, a shutdown system would not be too useful. In the same way, a system that has only the core operating system running but not the services that are supposed to run on the system is considered not available.

Integrity

Similar to availability, a system that is compromised cannot be considered available for regular service. Ensuring that the Unix system is running in the intended way is most crucial, especially since the system might otherwise be used by a third party for malicious uses, such as a relay or member in a botnet.

Achieving Unix Security

Prior to anything else, it is vitally important to emphasize the need to keep Unix systems up to date. No operating system or other program can be considered safe without being patched up; this point cannot be stressed enough. Having a system with the latest security patches is the first and most often the best line of defense against intruders.

All Unix systems have a patching mechanism; this is a way to get the system up to date. Depending on the vendor and the mechanism used, it is possible to "back out" the patches. For example, on Solaris it is usually possible to remove a patch through the `patchrm(1 m)` command. On Debian-based systems this is not quite as easy, since in a patch the software package to be updated is replaced by a new version. Undoing this is only possible by installing the earlier package.

Detecting Intrusions with Audits and Logs

By default, most Unix systems log kernel messages and important system events from core services. The most common logging tool is the syslog facility, which is controlled from the /etc/syslog.conf file.

2. Basic Unix Security

Unix security has a long tradition, and though many concepts of the earliest Unix systems still apply, there have been a large number of changes that fundamentally altered the way the operating system implements these security principles.

One of the reasons that it's complicated to talk about Unix security is that there are a lot of variants of Unix and Unix-like operating systems on the market. In fact, if you only look at some of the core Portable Operating System Interface (POSIX) standards that have been set forth to guarantee a minimal consistency across different Unix flavors (see Figure 5.1), almost every operating system on the market qualifies as Unix (or, more precisely, POSIX compliant). Examples include not only the traditional Unix operating systems such as Solaris, HP-UX, or AIX but also Windows NT-based operating systems (such as Windows XP, either through the native POSIX subsystem or the Services for Windows extensions) or even z/OS.

Traditional Unix Systems

Most traditional Unix systems do share some internal features, though: Their authentication and authorization approaches are similar, their delineation between kernel space and user space goes along the same lines, and their security-related kernel structures are roughly comparable. In the last few years, however, there have been major advancements in extending the original security model by adding role-based access control (RBAC) models to some operating systems.

Kernel Space Versus User Land

Unix systems typically execute instructions in one of two general contexts: the kernel or the user space. Code executed in a kernel context has (at least in traditional systems) full access to the entire hardware and software capabilities of the computing environment. Though there are some systems that extend security safeguards into the kernel, in most cases, not only can a rogue kernel execution thread cause massive data corruption, it can effectively bring down the entire operating system.

The term POSIX stands (loosely) for "Portable Operating System Interface for uniX". From the IEEE 1003.1 Standard, 2004 Edition:

"This standard defines a standard operating system interface and environment, including a command interpreter (or "shell"), and common utility programs to support applications portability at the source code level. This standard is the single common revision to IEEE Std 1003.1-1996, IEEE Std 1003.2-1992, and the Base Specifications of The Open Group Single UNIX Specification, Version 2."

Partial or full POSIX compliance is often required for government contracts.

Figure 5.1: Various Unix and POSIX standards.

Obviously, a normal user of an operating system should not wield so much power. To prevent this, user execution threads in Unix systems are not executed in the context of the kernel but in a less privileged context, the user space—sometimes also facetiously called "user land." The Unix kernel defines a structure called *process* (see Figure 5.2) that associates metadata about the user as well as, potentially, other environmental factors with the execution thread and its data. Access to computing resources such as memory, I/O subsystems, and so on is safeguarded by the kernel; if a user process wants to allocate a segment of memory or access a device, it has to make a system call, passing some of its metadata as parameters to the kernel. The kernel then performs an authorization decision and either grants the request or returns an error. It is then the process's responsibility to properly react to either the results of the access or the error.

If this model of user space process security is so effective, why not implement it for all operating system functions, including the majority of kernel operations? The answer to this question is that to a large extent the overhead of evaluating authorization metadata is very compute expensive. If most or all operations (that are, in the classical kernel space, often hardware-related device access operations) are run in user space or a comparable way, the performance of the OS would severely suffer. There is a class of operating system with a microkernel that implements this approach; the kernel implements only the most rudimentary functions (processes, scheduling, basic security), and all other operations, including device access and other operations that are typically carried out by the kernel, run in separate user processes. The advantage is a higher level of security and better safeguards against rogue device drivers. Furthermore, new device drivers or other operating system functionality can be added or removed without having to reboot the kernel. The performance penalties are so severe, however, that no major commercial operating system implements a microkernel architecture.

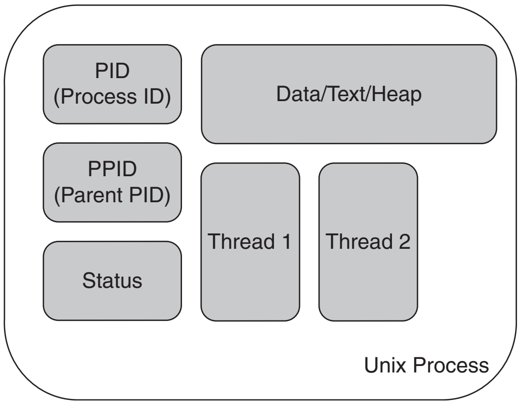


Figure 5.2: Kernel structure of a typical Unix process.

User Space Security

In traditional Unix systems, security starts with access control to resources. Since users interact with the systems through processes, it is important to know that every user space process structure has two important security fields: the user identifier, or UID, and the group identifier, or GID. These identifiers are typically positive integers, which are unique for each user [1]. Every process that is started by (or on behalf of) a user inherits the UID and GID values for that user account. These values are usually immutable for the live time of the process.

Access to system resources must go through the kernel by calling the appropriate function that is accessible to user processes. For example, a process that wants to reserve some system memory for data access will use the `malloc()` system call and pass the requested size and an (uninitialized) pointer as parameters. The kernel then evaluates this request, determines whether enough virtual memory (physical memory plus swap space) is available, reserves a section of memory, and sets the pointer to the address where the block starts.

Users who have the UID zero have special privileges: They are considered *superusers*, able to override many of the security guards that the kernel sets up. The default Unix superuser is named *root*.

Standard File and Device Access Semantics

File access is a very fundamental task, and it is important that only authorized users get read or write access to a given file. If any user was able to access any file, there would be no privacy at all, and security could not be maintained, since the operating system would not be able to protect its own permanent records, such as configuration information or user credentials.

The metadata describing who may access or modify files and directories is commonly referred to as an *access control list* (ACL). Note that there is more than just one type of ACL; the standard Unix ACLs are very well known, but different Unix variants or POSIX-like operating systems might implement different ACLs and only define a mapping to the simple POSIX 1003 semantics. A good example is the Windows NTFS ACL or the NFS v4 ACLs.

Read, Write, Execute

From its earliest days, Unix implemented a simple but effective way to set access rights for users. Normal files can be accessed in three fundamental ways: read, write, and execute. The first two ways are obvious; the execution requires a little more explanation. A file on disk may only be executed as either a binary program or a script if the user has the right to execute this file. If the execute permission is not set, the system call `exec()` will fail.

In addition to a user's permissions, there must be a notion of ownership of files and sometimes other resources. In fact, each file on a traditional Unix file system is associated with a user and a group. The user and group are not identified by their name but by UID and GID instead.

In addition to setting permissions for the user owning the file, two other sets of permissions are set for files: for the group and for all others. Similar to being owned by a user, a file is also associated with one group. All members of this group [2] can access the file with the permissions set for the group. In the same way, the other set of permissions applies to all users of the system.

Special Permissions

In addition to standard permissions, there are a few special permissions, discussed here.

Set-ID Bit

This permission only applies to executable files, and it can only be set for the user or the group. If this bit is set, the process for this program is not set to the UID or GID of the invoking user but instead the UID or GID of the file. For example, a program owned by the superuser can have the Set-ID bit set and execution allowed for all users. This way a normal user can execute a specific program with elevated privileges.

Sticky Bit

When the sticky bit is set on an executable file, its data (specifically the text segment) is kept in memory, even after the process exits. This is intended to speed execution of commonly used programs. A major drawback of setting the sticky bit is that when the executable file changes (e.g., through a patch), the permission must be unset and the program started once more. When this process exits, the executable is unloaded from memory and the file can be changed.

Mandatory Locking

Mandatory file and record locking refers to a file's ability to have its reading or writing permissions locked while a program is accessing that file.

In addition, there might be additional, implementation-specific permissions. These depend on the capabilities of the core operating facilities, including the kernel, but also on the type of file system. For example, most Unix operating systems can mount FAT-based file systems, which do not support any permissions or user and group ownership. Since the internal semantics require some values for ownership and permissions, these are typically set for the entire file system.

Permissions on Directories

The semantics of permissions on directories (see Figure 5.3) are different from those on files.

Making a directory readable for everyone:

```
# chmod o+r /tmp/mydir
# ls -ld /tmp/mydir

drwxr-xr-x   2 root        root        117 Aug   9 12:12 /tmp/mydir
```

Setting the SetID bit on an executable, thus enabling it to be run with super-user privileges:

```
# chmod u+s specialprivs
# ls -ld specialprivs

-rwsr-xr-x   2 root        root        117 Aug   9 12:12 specialprivs
```

Figure 5.3: Examples of chmod for files and directories.

Read and Write

Mapping these permissions to directories is fairly straightforward: The read permission allows listing files in the directory, and the write permission allows us to create files. For some applications it can be useful to allow writing but not reading.

Execute

If this permission is set, a process can set its working directory to this directory. Note that with the basic permissions, there is no limitation on traversing directories, so a process might change its working directory to a child of a directory, even if it cannot do so for the directory itself.

SetID

Semantics may differ here. For example, on Solaris this changes the behavior for default ownership of newly created files from the System V to the BSD semantics.

Other File Systems

As mentioned, the set of available permissions and authorization policies depends on the underlying operating system capabilities, including the file system. For example, the UFS file system in Solaris since version 2.5 allows additional ACLs on a per-user basis. Furthermore, NFS version 4 defines additional ACLs for file access; it is obvious that the NFS server must have an underlying files system that is capable of recording this additional metadata.

4. Protecting User Accounts and Strengthening Authentication

For any interactive session, Unix systems require the user to log into the system. To do so, the user must present a valid credential that identifies him (he must authenticate to the system).

Establishing Secure Account Use

The type of credentials a Unix system uses depends on the capabilities of the OS software itself and on the configuration set forth by the systems administrator. The most traditional user credential is a username and a text password, but there are many other ways to authenticate to the operating system, including Kerberos, secure shell (SSH), or security certificates.

The Unix Login Process

Depending on the desired authentication mechanism (see Figure 5.4), the user will have to use different access protocols or processes. For example, console or directly attached terminal sessions usually supports only password credentials or smart card logins, whereas a secure shell connection supports only RSA- or DSA-based cryptographic tokens over the SSH protocol.

The login process is a system daemon that is responsible for coordinating authentication and process setup for interactive users. To do this, the login process does the following:

1. Draw or display the login screen.
2. Collect the credential.
3. Present the user credential to any of the configured user databases [typically these can be files, network information system (NIS), Kerberos servers, or LDAP directories] for authentication.
4. Create a process with the user's default command-line shell, with the home directory as working directory.
5. Execute systemwide, user, and shell-specific startup scripts.

The commonly available X11 windowing system does not use the text-oriented login process but instead provides its own facility to perform roughly the same kind of login sequence.

Overview of UNIX authentication methods

- Simple: a username and a password are used to login to the operating system. The login process must receive both is cleartext. For the password, the UNIX crypt hash is calculated and compared to the value in the password or shadow file.
- Kerberos: The user is supposed to have a ticket-granting ticket from the Kerberos Key Distribution Server (KDC). Using the ticket-granting ticket, he obtains a service ticket for an interactive login to the UNIX host. This service ticket (encrypted, time limited) is then presented to the login process, and the UNIX host validates it with the KDC.
- PKI based Smartcard: the private key on the smart card is used to authenticate with the system.

Figure 5.4: Various authentication mechanisms for Unix systems.

Access to interactive sessions using the SSH protocol follows a similar general pattern, but the authentication is significantly different from the traditional login process.

Controlling Account Access

Simple files were the first method available to store user account data. Over the course of years many other user databases have been implemented. We examine these here.

The Local Files

Originally, Unix only supported a simple password file for storing account information. The username and the information required for the login process (UID, GID, shell, home directory, and GECOS information) are stored in this file, which is typically at `/etc/passwd`. This approach is highly insecure, since this file needs to be readable by all for a number of different services, thus exposing the password hashes to potential hackers. In fact, a simple dictionary or even brute-force attack can reveal simple or even more complex passwords.

To protect against an attack like this, most Unix variants use a separate file for storing the password hashes (`/etc/shadow`) that is only readable and writable by the system.

Network Information System

The NIS was introduced to simplify the administration of small groups of computers. Originally, Sun Microsystems called this service Yellow Pages, but the courts decided that this name constituted a trademark infringement on the British Telecom Yellow Pages. However, most commands that are used to administer the NIS still start with the `yp` prefix (such as `ypbind`, `ypcat`, etc.).

Systems within the NIS are said to belong to a NIS domain. Although there is absolutely no correlation between the NIS domain and the DNS domain of the system, it is quite common to use DNS-style domain names for naming NIS domains. For example, a system with DNS name `system1.sales.example.com` might be a member of the NIS domain `nis.sales.Example.COM`. Note that NIS domains—other than DNS domains—are case sensitive.

The NIS uses a simple master/slave server system: The master NIS server holds all authoritative data and uses an ONC-RPC-based protocol to communicate with the slave servers and clients. Slave servers cannot be easily upgraded to a master server, so careful planning of the infrastructure is highly recommended.

Client systems are bound to one NIS server (master or slave) during runtime. The addresses for the NIS master and the slaves must be provided when joining a system to the NIS domain. Clients (and servers) can always be members of only one NIS domain.

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch functionality.
#
passwd:        files nis

group:         files nis
shadow:        files nis

hosts:         files nis dns
networks:      files

protocols:     db files
services:      db files
ethers:        db files
rpc:           db files

netgroup:      nis
```

Figure 5.5: Sample nsswitch.conf for a Debian system.

To use the NIS user database (and other NIS resources, such as automount maps, netgroups, and host tables) after the system is bound, use the name service configuration file (`/etc/nsswitch.conf`), as shown in Figure 5.5.

Using PAMs to Modify AuthN

These user databases can easily be configured for use on a given system through the `/etc/nsswitch.conf` file. However, in more complex situations, the administrator might want to fine-tune the types of acceptable authentication methods, such as Kerberos, or even configure multifactor authentication. On many Unix systems, this is typically achieved through the pluggable authentication mechanism (PAM), as shown in Figure 5.6. Traditionally, the PAM is configured through the `/etc/pam.conf` file, but more modern implementations use a directory structure, similar to the System V init scripts. For these systems the administrator needs to modify the configuration files in the `/etc/pam.d/` directory.

Noninteractive Access

The security configuration of noninteractive services can vary quite significantly. Especially popular network services, such as LDAP, HTTP, or NFS, can use a wide variety of authentication and authorization mechanisms that do not even need to be provided by the operating system. For example, an Apache Web server or a MySQL database server might use its own user database, without relying on any operating system services.

```
#
# /etc/pam.d/common-password - password-related modules common to all services
#

# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define the services to be
# used to change user passwords. The default is pam_unix.

# Explanation of pam_unix options:
#
# The "nullok" option allows users to change an empty password, else
# empty passwords are treated as locked accounts.
#

# The "md5" option enables MD5 passwords. Without this option, the
# default is Unix crypt.
#

# The "obscure" option replaces the old 'OBSCURE_CHECKS_ENAB' option in
# login.defs.
#

# You can also use the "min" option to enforce the length of the new
# password.
#
# See the pam_unix manpage for other options.

password   requisite   pam_unix.so nullok obscure md5

# Alternate strength checking for password. Note that this
# requires the libpam-cracklib package to be installed.
# You will need to comment out the password line above and
# uncomment the next two in order to use this.
# (Replaces the 'OBSCURE_CHECKS_ENAB', 'CRACKLIB_DICTPATH')
#

password required   pam_cracklib.so retry=3 minlen=6 difok=3

password required   pam_unix.so use_authtok nullok md5
```

Figure 5.6: Setting the password strength on a Debian-based system through the PAM system.

Other Network Authentication Mechanisms

In 1983, BSD introduced the rlogin service. Unix administrators have been using RSH, RCP, and other tools from this package for a long time; they are very easy to use and configure and provide simple access across a small network of computers. The login was facilitated through a very simple trust model: Any user could create a .rhosts file in her home directory and specify foreign hosts and users from which to accept logins without proper credential checking. Over the rlogin protocol (TCP 513), the username of the rlogin client would be transmitted to the host system, and in lieu of an authentication, the rshd daemon would simply verify the preconfigured values. To prevent access from untrusted hosts, the administrator could use the /etc/hosts.equiv file to allow or deny individual hosts or groups of hosts (the latter through the use of NIS netgroups).

Risks of Trusted Hosts and Networks

Since no authentication ever takes place, this trust mechanism should not be used. Not only does this system rely entirely on the correct functioning of the hostname resolution system, but in addition, there is no way to determine whether a host was actually replaced [3]. Also, though rlogin-based trust systems might work for very small deployments, they become extremely hard to set up and operate with large numbers of machines.

Replacing Telnet, rlogin, and FTP Servers and Clients with SSH

The most sensible alternative to the traditional interactive session protocols such as Telnet is the secure shell system. It is very popular on Unix systems, and pretty much all versions ship with a version of SSH. Where SSH is not available, the open source package OpenSSH can easily be used instead [4].

SSH combines the ease-of-use features of the rlogin tools with a strong cryptographic authentication system. On one hand, it is fairly easy for users to enable access from other systems; on the other hand, the secure shell protocol uses strong cryptography to:

- Authenticate the connection, that is, establish the authenticity of the user
- Protect the privacy of the connection through encryption
- Guarantee the integrity of the channel through signatures

This is done using either the RSA or DSA security algorithm, which are both available for the SSH v2[5] protocol. The cipher (see Figure 5.7) used for encryption can be explicitly selected.

The user must first create a public/private key pair through the `ssh-keygen(1)` tool. The output of the key generator is placed in the .ssh subdirectory of the user's home directory. This output consists of a private key file called `id_dsa` or `id_rsa`. This file must be owned by the user and can only be readable by the user. In addition, a file containing the public key is created, named in the same way, with the extension .pub appended. The public key file is then placed into the .ssh subdirectory of the user's home directory on the target system.

Once the public and private keys are in place and the SSH daemon is enabled on the host system, all clients that implement the SSH protocol can create connections. There are four common applications using SSH:

```
$ ssh host -luser1 -c aes192-cbc
```

Figure 5.7: Create an interactive session on Solaris to host for user1 using the AES cipher with 192 bits.

- Interactive session is the replacement for Telnet and rlogin. Using the `ssh(1)` command line, the sshd daemon creates a new shell and transfers control to the user.
- In a remotely executed script/command, `ssh(1)` allows a single command with arguments to pass. This way, a single remote command (such as a backup script) can be executed on the remote system as long as this command is in the default path for the user.
- An SSH-enabled file transfer program can be used to replace the standard FTP or FTP over SSL protocol.
- Finally, the SSH protocol is able to tunnel arbitrary protocols. This means that any client can use the privacy and integrity protection offered by SSH. In particular, the X-Window system protocol can tunnel through an existing SSH connection by using the -X command-line switch.

5. Reducing Exposure to Threats by Limiting Superuser Privileges

The superuser has almost unlimited power on a Unix system, which can be a significant problem.

Controlling Root Access

There are a number of ways to limit access for the root user.

Configuring Secure Terminals

Most Unix systems allow us to restrict root logins to special terminals, typically the system console. This approach is quite effective, especially if the console or the allowed terminals are under strict physical access control. The obvious downside of this approach is that remote access to the system can be very limited: using this approach, access through any TCP/IP-based connection cannot be configured, thus requiring a direct connection, such as a directly attached terminal or a modem.

Configuration is quite different for the various Unix systems. Figure 5.8 shows the comparison between Solaris and Debian.

Gaining Root Privileges With `su`

The `su(1)` utility allows changing the identity of an interactive session. This is an effective mediation of the issues that come with restricting root access to secure terminals: Though only normal users can get access to the machine through the network (ideally by limiting the access protocols to those that protect the privacy of the communication, such as SSH), they can change their interactive session to a superuser session.

Using Groups Instead of Root

If users should be limited to executing certain commands with superuser privileges, it is possible and common to create special groups of users. For these groups, we can set the

On Solaris simply edit the file /etc/default/login:

```
# Copyright 2004 Sun Microsystems, Inc. All rights reserved.
# Use is subject to license terms.

# If CONSOLE is set, root can only login on that device.
# Comment this line out to allow remote login by root.
#

CONSOLE=/dev/console

# PASSREQ determines if login requires a password.
#

PASSREQ=YES

# SUPATH sets the initial shell PATH variable for root
#

SUPATH=/usr/sbin:/usr/bin

# SYSLOG determines whether the syslog(3) LOG_AUTH facility should be used
# to log all root logins at level LOG_NOTICE and multiple failed login
# attempts at LOG_CRIT.
#

SYSLOG=YES

# The SYSLOG_FAILED_LOGINS variable is used to determine how many failed
# login attempts will be allowed by the system before a failed login
# message is logged, using the syslog(3) LOG_NOTICE facility. For example,
# if the variable is set to 0, login will log -all- failed login attempts.
#

SYSLOG_FAILED_LOGINS=5
```

On Debian:

```
# The PAM configuration file for the Shadow 'login' service
#

# Disallows root logins except on tty's listed in /etc/securetty
# (Replaces the 'CONSOLE' setting from login.defs)

auth      requisite   pam_securetty.so

# Disallows other than root logins when /etc/nologin exists
# (Replaces the 'NOLOGINS_FILE' option from login.defs)

auth      requisite   pam_nologin.so

# Standard Un*x authentication.

@include common-auth

# This allows certain extra groups to be granted to a user
# based on things like time of day, tty, service, and user.
# Please edit /etc/security/group.conf to fit your needs
# (Replaces the 'CONSOLE_GROUPS' option in login.defs)
```

Figure 5.8: Restricting root access.

Continued

```
auth      optional   pam_group.so

# Uncomment and edit /etc/security/time.conf if you need to set
# time restrainst on logins.
# (Replaces the 'PORTTIME_CHECKS_ENAB' option from login.defs
# as well as /etc/porttime)

account   requisite     pam_time.so

# Uncomment and edit /etc/security/access.conf if you need to
# set access limits.
# (Replaces /etc/login.access file)

account   required     pam_access.so

# Sets up user limits according to /etc/security/limits.conf
# (Replaces the use of /etc/limits in old login)

session   required      pam_limits.so

# Prints the last login info upon succesful login
# (Replaces the 'LASTLOG_ENAB' option from login.defs)

session   optional      pam_lastlog.so

# Standard Un*x account and session

@include common-account
@include common-session
@include common-password
```

Figure 5.8—Cont'd

execution bit on programs (while disabling execution for all others) and the SetID bit for the owner, in this case the superuser. Therefore, only users of such a special group can execute the given utility with superuser privileges.

Using the `sudo(1)` *Mechanism*

By far more flexible and easier to manage than the approach for enabling privileged execution based on groups is the `sudo(1)` mechanism. Originally an open source program, `sudo(1)` is available for most Unix distributions. The detailed configuration is quite complex, and the manual page is quite informative.

6. Safeguarding Vital Data by Securing Local and Network File Systems

For production systems, there is a very effective way of preventing the modification of system-critical resources by unauthorized users or malicious software. Critical portions of the file systems (such as the locations of binary files, system libraries, and some configuration files) do not necessarily change very often.

The following scheme is a good start for partitioning with read-only partitions:

- Binaries and Libraries: /bin, /lib, /sbin, /usr – read-only
- Logs and frequently changing system data: /var, /usr/var – writable
- User home directories: /home, /export/home – writable
- Additional software packages: /opt, /usr/local – read-only
- System configuration: /etc, /usr/local/etc – writable
- Everything else: Root (/) - read-only

Obviously, this can only be a start and should be evaluated for each system and application. Updating operating system files, including those on the root file system, should be updated in single-user mode with all partitions mounted writable.

Figure 5.9: Secure partitioning.

Directory Structure and Partitioning for Security

In fact, any systemwide binary code should probably only be modified by the systems administrators. In these cases, it is very effective to properly partition the file system.

Employing Read-Only Partitions

The reason to properly partition the file system (see Figure 5.9) is so that only frequently changing files (such as user data, log files, and the like) are hosted on readable file systems. All other storage can then be mounted on read-only partitions.

Ownership and Access Permissions

To prevent inadvertent or malicious access to critical data, it is vitally important to verify the correct ownership and permission set for all critical files in the file system. The Unix `find(1)` command is an effective way to locate files with certain characteristics. In the following, a number of sample command-line options for this utility are given to locate files.

Locate SetID Files

Since executables with the SetID bit set are often used to allow the execution of a program with superuser privileges, it is vitally important to monitor these files on a regular basis.

Another critical permission set is that of world-writable files; there should be no system-critical files in this list, and users should be aware of any files in their home directories that are world-writable (see Figure 5.10).

```
find / \( -perm -04000 -o -perm -02000\) -type f -xdev -print
```

Figure 5.10: Finding files with SUID and SGID set.

Finally, files and directories that are not owned by current users can be found by the code shown in Figure 5.11.

For groups, just use `-nogroup` instead.

```
find / -nouser
```

Figure 5.11: Finding files without users.

References

[1] If two usernames are associated with the same UID, the operating system will treat them as the same user. Their authentication credentials (username and password) are different, but their authorization with respect to system resources is the same.

[2] It should be noted that users belong to one primary group, identified by the GID set in the password database. However, group membership is actually determined separately through the `/etc/group` file. As such, user can be (and often is) a member of more than one group.

[3] This could actually be addressed through host authentication, but it is not a feature of the rlogin protocol.

[4] See [IEEE04]. www.opengroup.org/onlinepubs/009695399/.

This page intentionally left blank

CHAPTER 6

Eliminating the Security Weakness of Linux and UNIX Operating Systems

Mario Santana
Terremark

Linux and other Unix-like operating systems are prevalent on the Internet for a number of reasons. As an operating system designed to be flexible and robust, Unix lends itself to providing a wide array of host- and network-based services. Unix also has a rich culture from its long history as a fundamental part of computing research in industry and academia. Unix and related operating systems play a key role as platforms for delivering the key services that make the Internet possible.

For these reasons, it is important that information security practitioners understand fundamental Unix concepts in support of practical knowledge of how Unix systems might be securely operated. This chapter is an introduction to Unix in general and to Linux in particular, presenting some historical context and describing some fundamental aspects of the operating system architecture. Considerations for hardening Unix deployments will be contemplated from network-centric, host-based, and systems management perspectives. Finally, proactive considerations are presented to identify security weaknesses to correct them and to deal effectively with security breaches when they do occur.

1. Introduction to Linux and Unix

A simple Google search for "define:unix" yields many definitions, including this one from Microsoft: "A powerful multitasking operating system developed in 1969 for use in a minicomputer environment; still a widely used network operating system." [1]

What Is Unix?

Unix is many things. Officially, it is a brand and an operating system specification. In common usage the word *Unix* is often used to refer to one or more of many operating

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00006-3

systems that derive from or are similar to the operating system designed and implemented about 40 years ago at AT&T Bell Laboratories. Throughout this chapter, we'll use the term *Unix* to include official Unix-branded operating systems as well as Unix-like operating systems such as BSD, Linux, and even Macintosh OS X.

History

Years after AT&T's original implementation, there followed decades of aggressive market wars among many operating system vendors, each claiming that its operating system was Unix. The ever-increasing incompatibilities between these different versions of Unix were seen as a major deterrent to the marketing and sales of Unix. As personal computers grew more powerful and flexible, running inexpensive operating systems like Microsoft Windows and IBM OS/2, they threatened Unix as the server platform of choice. In response to these and other marketplace pressures, most major Unix vendors eventually backed efforts to standardize the Unix operating system.

Unix is a Brand

Since the early 1990s, the Unix brand has been owned by The Open Group. This organization manages a set of specifications with which vendors must comply to use the Unix brand in referring to their operating system products. In this way, The Open Group provides a guarantee to the marketplace that any system labeled as Unix conforms to a strict set of standards.

Unix is a Specification

The Open Group's standard is called the Single Unix Specification. It is created in collaboration with the Institute of Electrical and Electronics Engineers (IEEE), the International Standards Organization (ISO), and others. The specification is developed, refined, and updated in an open, transparent process.

The Single Unix Specification comprises several components, covering core system interfaces such as system calls as well as commands, utilities, and a development environment based on the C programming language. Together, these describe a "functional superset of consensus-based specifications and historical practice." [2]

Lineage

The phrase *historical practice* in the description of the Single Unix Specification refers to the many operating systems historically referring to themselves as Unix. These include everything from AT&T's original releases to the versions released by the University of California at Berkeley and major commercial offerings by the likes of IBM, Sun, Digital Equipment Corporation (DEC), Hewlett-Packard (HP), the Santa Cruz Operation (SCO),

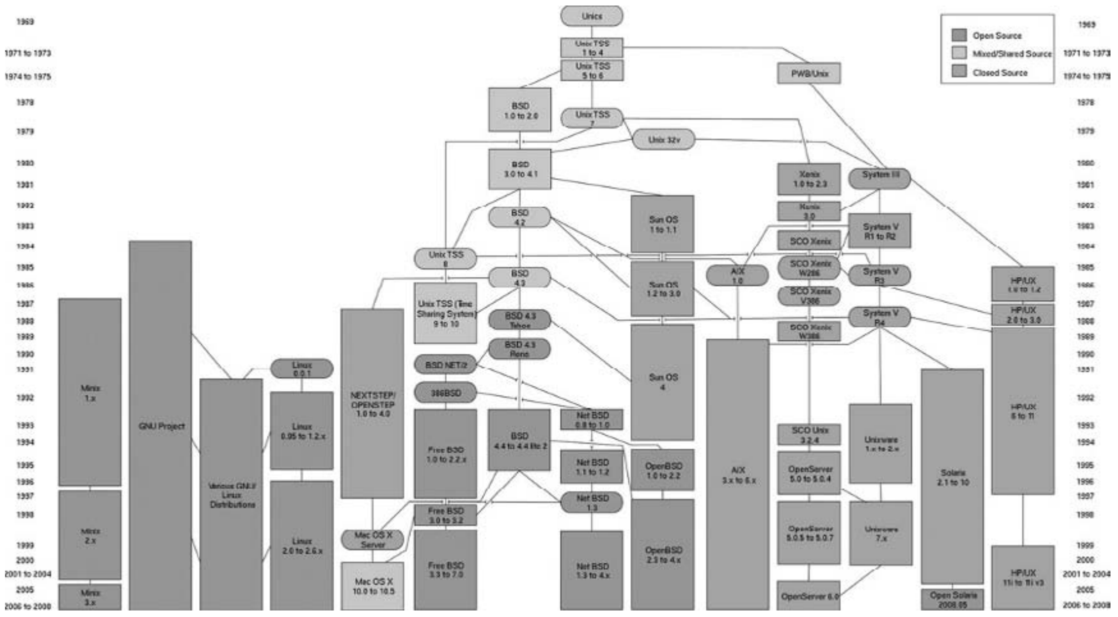


Figure 6.1: The simplified Unix family tree presents a timeline of some of today's most successful Unix variants [10].

Novell, and even Microsoft. But any list of Unix operating systems would be incomplete if it didn't mention Linux (see Figure 6.1).

What Is Linux?

Linux is a bit of an oddball in the Unix operating system lineup. That's because, unlike the Unix versions released by the major vendors, Linux did not reuse any existing source code. Instead, Linux was developed from scratch by a Finnish university student named Linus Torvalds.

Most Popular Unix-Like OS

Linux was written from the start to function very similarly to existing Unix products. And because Torvalds worked on Linux as a hobby, with no intention of making money, it was distributed for free. These factors and others contributed to making Linux the most popular Unix operating system today.

Linux is a Kernel

Strictly speaking, Torvalds' pet project has provided only one part of a fully functional Unix operating system: the kernel. The other parts of the operating system, including the commands, utilities, development environment, desktop environment, and other aspects of a full Unix operating system, are provided by other parties, including GNU, XOrg, and others.

Linux is a Community

Perhaps the most fundamentally different thing about Linux is the process by which it is developed and improved. As the hobby project that it was, Linux was released by Torvalds on the Internet in the hopes that someone out there might find it interesting. A few programmers saw Torvalds' hobby kernel and began working on it for fun, adding features and fleshing out functionality in a sort of unofficial partnership with Torvald. At this point, everyone was just having fun, tinkering with interesting concepts. As more and more people joined the unofficial club, Torvalds' pet project ballooned into a worldwide phenomenon.

Today, Linux is developed and maintained by hundreds of thousands of contributors all over the world. In 1996, Eric S. Raymond [3] famously described the distributed development methodology used by Linux as a bazaar—a wild, uproarious collection of people, each developing whatever feature they most wanted in an operating system, or improving whatever shortcoming most impacted them; yet somehow, this quick-moving community resulted in a development process that was stable as a whole, and that produced an amazing amount of progress in a very short time.

This is radically different from the way in which Unix systems have typically been developed. If the Linux community is like a bazaar, then other Unix systems can be described as a cathedral—carefully preplanned and painstakingly assembled over a long period of time, according to specifications handed down by master architects from previous generations. Recently, however, some of the traditional Unix vendors have started moving toward a more decentralized, bazaar-like development model similar in many ways to the Linux methodology.

Linux is Distributions

The Open Source movement in general is very important to the success of Linux. Thanks to GNU, XOrg, and other open-source contributors, there was an almost complete Unix already available when the Linux kernel was released. Linux only filled in the final missing component of a no-cost, open source Unix. Because the majority of the other parts of the operating system came from the GNU project, Linux is also known as GNU/Linux.

To actually install and run Linux, it is necessary to collect all the other operating system components. Because of the interdependency of the operating system components—each component must be compatible with the others—it is important to gather the right versions of all these components. In the early days of Linux, this was quite a challenge!

Soon, however, someone gathered up a self-consistent set of components and made them all available from a central download location. The first such efforts include H. J. Lu's "boot/root" floppies and MCC Interim Linux. These folks did not necessarily develop any of these components; they only redistributed them in a more convenient package. Other people did the same, releasing new bundles called *distributions* whenever a major upgrade was available.

Some distributions touted the latest in hardware support; others specialized in mathematics or graphics or another type of computing; still others built a distribution that would provide the simplest or most attractive user experience. Over time, distributions have become more robust, offering important features such as package management, which allows a user to safely upgrade parts of the system without reinstalling everything else.

Linux Standard Base

Today there are dozens of Linux distributions. Different flavors of distributions have evolved over the years. A primary distinguishing feature is the package management system. Some distributions are primarily volunteer community efforts; others are commercial offerings. See Figure 6.2 for a timeline of Linux development [4].

The explosion in the number of different Linux distributions created a situation reminiscent of the Unix wars of previous decades. To address this issue, the Linux Standard Base was created to specify certain key standards of behavior for conforming Linux distributions. Most major distributions comply with the Linux Standard Base specifications.

System Architecture

The architecture of Unix operating systems is relatively simple. The kernel interfaces with hardware and provides core functionality for the system. File systems provide permanent storage and access to many other kinds of functionality. Processes embody programs as their instructions are being executed. Permissions describe the actions that users may take on files and other resources.

Kernel

The operating system kernel manages many of the fundamental details that an operating system needs to deal with, including memory, disk storage, and low-level networking. In general, the kernel is the part of the operating system that talks directly to hardware; it presents an abstracted interface to the rest of the operating system components.

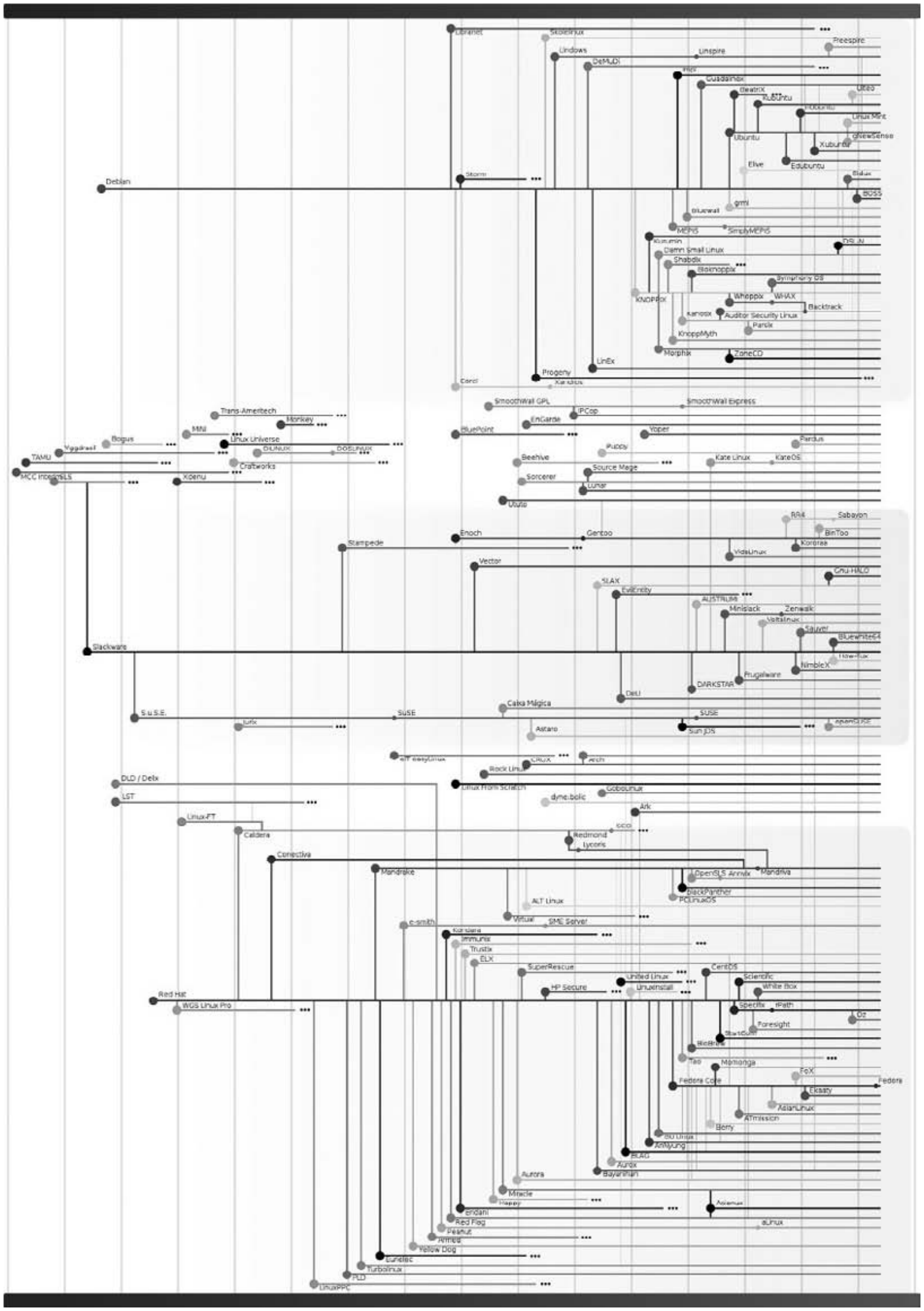


Figure 6.2: History of Linux distributions.

Because the kernel understands all the different sorts of hardware that the operating system deals with, the rest of the operating system is freed from needing to understand all those underlying details. The abstracted interface presented by the kernel allows other parts of the operating system to read and write files or communicate on the network without knowing or caring what kinds of disks or network adapter are installed.

File System

A fundamental aspect of Unix is its file system. Unix pioneered the hierarchical model of directories that contain files and/or other directories to allow the organization of data into a tree structure. Multiple file systems could be accessed by connecting them to empty directories in the root file system. In essence, this is very much like grafting one hierarchy onto an unused branch of another. There is no limit to the number of file systems that can be mounted in this way.

The file system hierarchy is also used to provide more than just access to and organization of local files. Network data shares can also be mounted, just like file systems on local disks. And special files such as device files, first in/first out (FIFO) or pipe files, and others give direct access to hardware or other system features.

Users and Groups

Unix was designed to be a time-sharing system, and as such has been multiuser since its inception. Users are identified in Unix by their usernames, but internally each is represented as a unique identifying integer called a *user ID*, or *UID*. Each user can also belong to one or more groups. Like users, groups are identified by their names, but they are represented internally as a unique integer called a *group ID*, or *GID*. Each file or directory in a Unix file system is associated with a user and a group.

Permissions

Unix has traditionally had a simple permissions architecture, based on the user and group associated with files in the file system. This scheme makes it possible to specify read, write, and/or execute permissions, along with a special permission setting whose effect is context-dependent. Furthermore, it's possible to set these permissions independently for the file's owner; the file's group, in which case the permission applies to all users, other than the owner, who are members of that group; and to all other users. The `chmod` command is used to set the permissions by adding up the values of each permission, as shown in Table 6.1.

The Unix permission architecture has historically been the target of criticism for its simplicity and inflexibility. It is not possible, for example, to specify a different permission setting for more than one user or more than one group. These limitations have been addressed in more recent file system implementations using extended file attributes and access control lists.

Table 6.1: Unix Permissions and Chmod

`chmod` Usage	Read	Write	Execute	Special
User	u + r or 0004	u + w or 0002	u + x or 0001	u + s or 4000
Group	u + r or 0040	u + w or 0020	u + x or 0010	u + s or 2000
Other	u + r or 0400	u + w or 0200	u + x or 0100	u + s or 1000

Processes

When a program is executed, it is represented in a Unix system as a process. The kernel keeps track of many pieces of information about each process. This information is required for basic housekeeping and advanced tasks such as tracing and debugging. This information represents the user, group, and other data used for making security decisions about a process's access rights to files and other resources.

2. Hardening Linux and Unix

With a basic understanding of the fundamental concepts of the Unix architecture, let's take a look at the practical work of securing a Unix deployment. First we'll review considerations for securing Unix machines from network-borne attacks. Then we'll look at security from a host-based perspective. Finally, we'll talk about systems management and how different ways of administering a Unix system can impact security.

Network Hardening

Defending from network-borne attacks is arguably the most important aspect of Unix security. Unix machines are used heavily to provide network-based services, running Web sites, DNS, firewalls, and many more. To provide these services, Unix systems must be connected to hostile networks, such as the Internet, where legitimate users can easily access and make use of these services.

Unfortunately, providing easy access to legitimate users makes the system easily accessible to bad actors who would subvert access controls and other security measures to steal sensitive information, change reference data, or simply make services unavailable to legitimate users. Attackers can probe systems for security weaknesses, identify and exploit vulnerabilities, and generally wreak digital havoc with relative impunity from anywhere around the globe.

Minimizing Attack Surface

Every way in which an attacker can interact with the system poses a security risk. Any system that makes available a large number of network services, especially complex services such as the custom Web applications of today, suffers a higher likelihood that inadequate permissions or a software bug or some other error will present attackers with an opportunity to compromise security. In contrast, even a very insecure service cannot be compromised if it is not running.

A pillar of any security architecture is the concept of minimizing the attack surface. By reducing the number of enabled network services and by reducing the available functionality of those services that are enabled, a system presents a smaller set of functions that can be subverted by an attacker. Other ways to reduce attackable surface area are to deny network access from unknown hosts when possible and to limit the privileges of running services, to limit the extent of the damage they might be subverted to cause.

Eliminate Unnecessary Services

The first step in reducing attack surface is to disable unnecessary services provided by a server. In Unix, services are enabled in one of several ways. The "Internet daemon," or *inetd*, is a historically popular mechanism for managing network services. Like many Unix programs, inetd is configured by editing a text file. In the case of inetd, this text file is /etc/inetd.conf; unnecessary services should be commented out of this file. Today a more modular replacement for inetd, called *xinetd,* is gaining popularity. The configuration for xinetd is not contained in any single file but in many files located in the /etc/xinetd.d/ directory. Each file in this directory configures a single service, and a service may be disabled by removing the file or by making the appropriate changes to the file.

Many Unix services are not managed by inetd or xinetd, however. Network services are often started by the system's initialization scripts during the boot sequence. Derivatives of the BSD Unix family historically used a simple initialization script located in /etc/rc. To control the services that are started during the boot sequence, it is necessary to edit this script.

Recent Unices (the plural of Unix), even BSD derivatives, use something similar to the initialization scheme of the System V family. In this scheme, a "run level" is chosen at boot time. The default run level is defined in /etc/inittab; typically, it is 3 or 5. The initialization scripts for each run level are located in /etc/rc*X*.d, where *X* represents the run-level number. The services that are started during the boot process are controlled by adding or removing scripts in the appropriate run-level directory. Some Unices provide tools to help manage these scripts, such as the `chkconfig` command in Red Hat Linux and derivatives. There are also other methods of managing services in Unix, such as the Service Management Facility of Solaris 10.

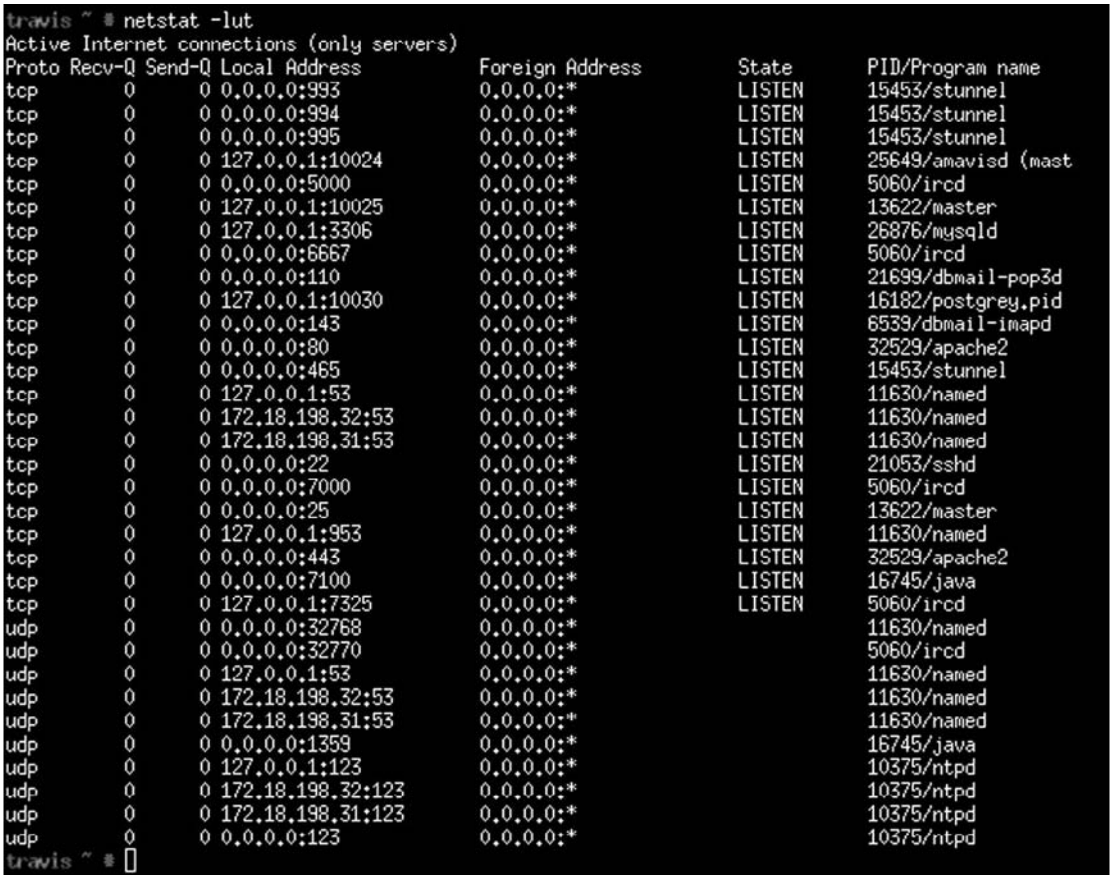
No matter how a network service is started or managed, however, it must necessarily listen for network connections to make itself available to users. This fact makes it possible to positively identify all running network services by looking for processes that are listening for network connections. Almost all versions of Unix provide a command that makes this a trivial task. The `netstat` command can be used to list various kinds of information about the network environment of a Unix host. Running this command with the appropriate flags (usually `–lut`) will produce a listing of all open network ports, including those that are listening for incoming connections (see Figure 6.3).

Every such listening port should correspond to a necessary service that is well understood and securely configured.

Host-Based

Obviously, it is impossible to disable all the services provided by a server. However, it is possible to limit the hosts that have access to a given service. Often it is possible to identify a well-defined list of hosts or subnets that should be granted access to a network service. There are several ways in which this restriction can be configured.

A classical way of configuring these limitations is through the *tcpwrappers* interface. The tcpwrappers functionality is to limit the network hosts that are allowed to access services



```
travis ~ # netstat -lut
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:993             0.0.0.0:*               LISTEN      15453/stunnel
tcp        0      0 0.0.0.0:994             0.0.0.0:*               LISTEN      15453/stunnel
tcp        0      0 0.0.0.0:995             0.0.0.0:*               LISTEN      15453/stunnel
tcp        0      0 127.0.0.1:10024         0.0.0.0:*               LISTEN      25649/amavisd (mast
tcp        0      0 0.0.0.0:5000            0.0.0.0:*               LISTEN      5060/ircd
tcp        0      0 127.0.0.1:10025         0.0.0.0:*               LISTEN      13622/master
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      26876/mysqld
tcp        0      0 0.0.0.0:6667            0.0.0.0:*               LISTEN      5060/ircd
tcp        0      0 0.0.0.0:110             0.0.0.0:*               LISTEN      21699/dbmail-pop3d
tcp        0      0 127.0.0.1:10030         0.0.0.0:*               LISTEN      16182/postgrey.pid
tcp        0      0 0.0.0.0:143             0.0.0.0:*               LISTEN      6539/dbmail-imapd
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      32529/apache2
tcp        0      0 0.0.0.0:465             0.0.0.0:*               LISTEN      15453/stunnel
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      11630/named
tcp        0      0 172.18.198.32:53        0.0.0.0:*               LISTEN      11630/named
tcp        0      0 172.18.198.31:53        0.0.0.0:*               LISTEN      11630/named
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      21053/sshd
tcp        0      0 0.0.0.0:7000            0.0.0.0:*               LISTEN      5060/ircd
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN      13622/master
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      11630/named
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      32529/apache2
tcp        0      0 0.0.0.0:7100            0.0.0.0:*               LISTEN      16745/java
tcp        0      0 127.0.0.1:7325          0.0.0.0:*               LISTEN      5060/ircd
udp        0      0 0.0.0.0:32768           0.0.0.0:*                           11630/named
udp        0      0 0.0.0.0:32770           0.0.0.0:*                           5060/ircd
udp        0      0 127.0.0.1:53            0.0.0.0:*                           11630/named
udp        0      0 172.18.198.32:53        0.0.0.0:*                           11630/named
udp        0      0 172.18.198.31:53        0.0.0.0:*                           11630/named
udp        0      0 0.0.0.0:1359            0.0.0.0:*                           16745/java
udp        0      0 127.0.0.1:123           0.0.0.0:*                           10375/ntpd
udp        0      0 172.18.198.32:123       0.0.0.0:*                           10375/ntpd
udp        0      0 172.18.198.31:123       0.0.0.0:*                           10375/ntpd
udp        0      0 0.0.0.0:123             0.0.0.0:*                           10375/ntpd
travis ~ #
```

Figure 6.3: Output of netstat –lut.

provided by the server. These controls are configured in two text files, /etc/hosts.allow and /etc/hosts.deny. This interface was originally designed to be used by inetd and xinetd on behalf of the services they manage. Today most service-providing software directly supports this functionality.

Another, more robust method of controlling network access is through firewall configurations. Most modern Unices include some form of firewall capability: IPFilter, used by many commercial Unices; IPFW, used by most of the BSD variants, and IPTables, used by Linux. In all cases, the best way to arrive at a secure configuration is to create a default rule to deny all traffic, and to then create the fewest, most specific exceptions possible.

Modern firewall implementations are able to analyze every aspect of the network traffic they filter as well as aggregate traffic into logical connections and track the state of those connections. The ability to accept or deny connections based on more than just the originating network address and to end a conversation when certain conditions are met makes modern firewalls a much more powerful control for limiting attack surface than tcpwrappers.

`chroot` and Other Jails

Eventually some network hosts must be allowed to access a service if it is to be useful at all. In fact, it is often necessary to allow anyone on the Internet to access a service, such as a public Web site. Once a malicious user can access a service, there is a risk that the service will be subverted into executing unauthorized instructions on behalf of the attacker.
The potential for damage is limited only by the permissions that the service pro-cess has to access resources and to make changes on the system. For this reason, an important security measure is to limit the power of a service to the bare minimum necessary to allow it to perform its duties.

A primary method of achieving this goal is to associate the service process with a user who has limited permissions. In many cases, it's possible to configure a user with very few permissions on the system and to associate that user with a service process. In these cases, the service can only perform a limited amount of damage, even if it is subverted by attackers.

Unfortunately, this is not always very effective or even possible. A service must often access sensitive server resources to perform its work. Configuring a set of permissions to allow access to only the sensitive information required for a service to operate can be complex or impossible.

In answer to this challenge, Unix has long supported the `chroot` and `ulimit` interfaces as ways to limit the access that a powerful process has on a system. The `chroot` interface limits a process's access on the file system. Regardless of actual permissions, a process run under a `chroot` jail can only access a certain part of the file system. Common practice is to run sensitive or powerful services in a `chroot` jail and make a copy of only those file system resources that the service needs in order to operate. This allows a service to run with a high level of system access, yet be unable to damage the contents of the file system outside the portion it is allocated [5].

The `ulimit` interface is somewhat different in that it can configure limits on the amount of system resources a process or user may consume. A limited amount of disk space, memory, CPU utilization, and other resources can be set for a service process. This can curtail the possibility of a denial-of-service attack because the service cannot exhaust all system resources, even if it has been subverted by an attacker [6].

Access Control

Reducing the attack surface area of a system limits the ways in which an attacker can interact and therefore subvert a server. Access control can be seen as another way to reduce the attack surface area. By requiring all users to prove their identity before making any use of a service, access control reduces the number of ways in which an anonymous attacker can interact with the system.

In general, access control involves three phases. The first phase is identification, where a user asserts his identity. The second phase is authentication, where the user proves his identity. The third phase is authorization, where the server allows or disallows particular actions based on permissions assigned to the authenticated user.

Strong Authentication

It is critical, therefore, that a secure mechanism is used to prove the user's identity. If this mechanism were to be subverted, an attacker would be able to impersonate a user to access resources or issue commands with whatever authorization level has been granted to that user. For decades, the primary form of authentication has been through the use of passwords. However, passwords suffer from several weaknesses as a form of authentication, presenting attackers with opportunities to impersonate legitimate users for illegitimate ends. Bruce Schneier has argued for years that "passwords have outlived their usefulness as a serious security device." [7]

More secure authentication mechanisms include two-factor authentication and PKI certificates.

Two-Factor Authentication Two-factor authentication involves the presentation of two of the following types of information by users to prove their identity: something they know, something they have, or something they are. The first factor, something they know, is typified by a password or a PIN—some shared secret that only the legitimate user should know. The second factor, something they have, is usually fulfilled by a unique physical token

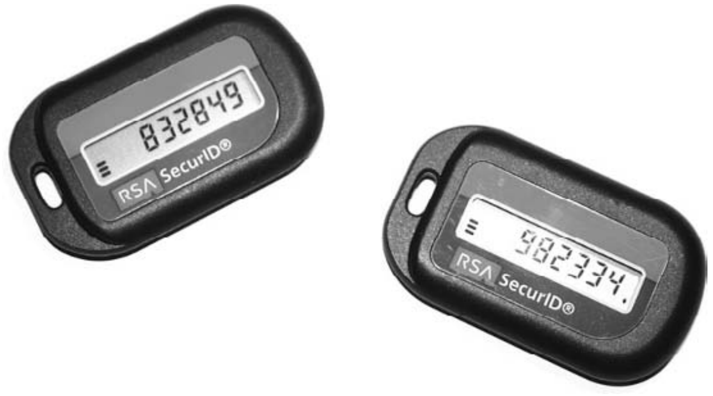


Figure 6.4: Physical tokens used for two-factor authentication.

(see Figure 6.4). RSA makes a popular line of such tokens, but cell phones, matrix cards, and other alternatives are becoming more common. The third factor, something they are, usually refers to biometrics.

Unix supports various ways to implement two-factor authentication into the system. Pluggable Authentication Modules, or PAMs, allow a program to use arbitrary authentication mechanisms without needing to manage any of the details. PAMs are used by Solaris, Linux, and other Unices. BSD authentication serves a similar purpose and is used by several major BSD derivatives.

With PAM or BSD authentication, it is possible to configure any combination of authentication mechanisms, including simple passwords, biometrics, RSA tokens, Kerberos, and more. It's also possible to configure a different combination for different services. This kind of flexibility allows a Unix security administrator to implement a very strong authentication requirement as a prerequisite for access to sensitive services.

PKI Strong authentication can also be implemented using a Private Key Infrastructure, or PKI. Secure Socket Layer, or SSL, is a simplified PKI designed for secure communications, familiar from its use in securing traffic on the Web. Using a similar foundation of technologies, it's possible to issue and manage certificates to authenticate users rather than Web sites. Additional technologies, such as a trusted platform module or a smart card, simplify the use of these certificates in support of two-factor authentication.

Dedicated Service Accounts

After strong authentication, limiting the complexity of the authorization phase is the most important part of access control. User accounts should not be authorized to perform sensitive tasks. Services should be associated with dedicated user accounts, which should then be authorized to perform only those tasks required for providing that service.

Additional Controls

In addition to minimizing the attack surface area and implementing strong access controls, there are several important aspects of securing a Unix network server.

Encrypted Communications

One of the ways an attacker can steal sensitive information is to eavesdrop on network traffic. Information is vulnerable as it flows across the network, unless it is encrypted. Sensitive information, including passwords and intellectual property, are routinely transmitted over the network. Even information that is seemingly useless to an attacker can contain important clues to help a bad actor compromise security.

File Transfer Protocol (FTP), World Wide Web (WWW), and many other services that transmit information over the network support the Secure Sockets Layer standard, or SSL, for encrypted communications. For server software that doesn't support SSL natively, wrappers like *stunnel* provide transparent SSL functionality.

No discussion of Unix network encryption can be complete without mention of Secure Shell, or SSH. SSH is a replacement for Telnet and RSH, providing remote command-line access to Unix systems as well as other functionality. SSH encrypts all network communications using SSL, mitigating many of the risks of Telnet and RSH.

Log Analysis

In addition to encrypting network communications, it is important to keep a detailed activity log to provide an audit trail in case of anomalous behavior. At a minimum, the logs should capture system activity such as logon and logoff events as well as service program activity, such as FTP, WWW, or Structured Query Language (SQL) logs.

Since the 1980s, the *syslog* service has historically been used to manage log entries in Unix. Over the years, the original implementation has been replaced by more feature-rich implementations, such as *syslog-ng* and *rsyslog*. These systems can be configured to send log messages to local files as well as remote destinations, based on independently defined verbosity levels and message sources.

The syslog system can independently route messages based on the facility, or message source, and the level, or message importance. The facility can identify the message as pertaining to the kernel, the email system, user activity, an authentication event, or any of various other services. The level denotes the criticality of the message and can typically be one of *emergency, alert, critical, error, warning, notice, informational,* and *debug*. Under Linux, the *klog* process is responsible for handling log messages generated by the kernel; typically, klog is configured to route these messages through syslog, just like any other process.

Some services, such as the Apache Web server, have limited or no support for syslog. These services typically include the ability to log activity to a file independently. In these cases, simple scripts can redirect the contents of these files to syslog for further distribution and/or processing.

Relevant logs should be copied to a remote, secure server to ensure that they cannot be tampered with. Additionally, file hashes should be used to identify any attempt to tamper with the logs. In this way, the audit trail provided by the log files can be depended on as a source of uncompromised information about the security status of the system.

IDS/IPS

Intrusion detection systems (IDSs) and intrusion prevention systems (IPSs) have become commonplace security items on today's networks. Unix has a rich heritage of such software, including Snort, Prelude, and OSSEC. Correctly deployed, an IDS can provide an early warning of probes and other precursors to attack.

Host Hardening

Unfortunately, not all attacks originate from the network. Malicious users often gain access to a system through legitimate means, bypassing network-based defenses. There are various steps that can be taken to harden a Unix system from a host-based attack such as this.

Permissions

The most obvious step is to limit the permissions of user accounts on the Unix host. Recall that every file and directory in a Unix file system is associated with a single user and a single group. User accounts should each have permissions that allow full control of their respective home directories. Together with permissions to read and execute system programs, this allows most of the typical functionality required of a Unix user account. Additional permissions that might be required include mail spool files and directories as well as crontab files for scheduling tasks.

Administrative Accounts

Setting permissions for administrative users is a more complicated question. These accounts must access very powerful system-level commands and resources in the routine discharge of their administrative functions. For this reason, it's difficult to limit the tasks these users may perform. It's possible, however, to create specialized administrative user accounts and then authorize these accounts to access a well-defined subset of administrative resources. Printer management, Web site administration, email management, database administration, storage management, backup administration, software upgrades, and other specific administrative functions common to Unix systems lend themselves to this approach.

Groups

Often it is convenient to apply permissions to a set of users rather than a single user or all users. The Unix group mechanism allows for a single user to belong to one or more groups and for file system permissions and other access controls to be applied to a group.

File System Attributes and ACLs

It can become unfeasibly complex to implement and manage anything more than a simple permissions scheme using the classical Unix file system permission capabilities. To overcome this issue, modern Unix file systems support access control lists, or ACLs. Most Unix file systems support ACLs using extended attributes that could be used to store arbitrary information about any given file or directory. By recognizing authorization information in these extended attributes, the file system implements a comprehensive mechanism to specify arbitrarily complex permissions for any file system resource.

ACLs contain a list of *access control entries*, or ACEs, which specify the permissions that a user or group has on the file system resource in question. On most Unices, the chacl command is used to view and set the ACEs of a given file or directory. The ACL support in modern Unix file systems provides a fine-grained mechanism for managing complex permissions requirements. ACLs do not make the setting of minimum permissions a trivial matter, but complex scenarios can now be addressed effectively.

Intrusion Detection

Even after hardening a Unix system with restrictive user permissions and ACLs, it's important to maintain logs of system activity. As with activity logs of network services, host-centric activity logs track security-relevant events that could show symptoms of compromise or evidence of attacks in the reconnaissance or planning stages.

Audit Trails

Again, as with network activity logs, Unix has leaned heavily on syslog to collect, organize, distribute, and store log messages about system activity. Configuring syslog for system messages is the same as for network service messages. The kernel's messages, including those messages generated on behalf of the kernel by klogd under Linux, are especially relevant from a host-centric point of view.

An additional source of audit trail data about system activity is the history logs kept by a login shell such as *bash*. These logs record every command the user issued at the command line. The bash shell and others can be configured to keep these logs in a secure location and to attach time stamps to each log entry. This information is invaluable in identifying malicious activity, both as it is happening as well as after the fact.

File Changes

Besides tracking activity logs, monitoring file changes can be a valuable indicator of suspicious system activity. Attackers often modify system files to elevate privileges, capture passwords or other credentials, establish backdoors to ensure future access to the system, and support other illegitimate uses. Identifying these changes early can often foil an attack in progress before the attacker is able to cause significant damage or loss.

Programs such as Tripwire and Aide have been around for decades; their function is to monitor the file system for unauthorized changes and raise an alert when one is found. Historically, they functioned by scanning the file system and generating a unique *hash*, or fingerprint, of each file. On future runs, the tool would recalculate the hashes and identify changed files by the difference in the hash. Limitations of this approach include the need to regularly scan the entire file system, which can be a slow operation, as well as the need to secure the database of file hashes from tampering.

Today many Unix systems support file change monitoring: Linux has dnotify and inotify; Mac OS X has FSEvents, and other Unices have File Alteration Monitor. All these present an alternative method of identifying file changes and reviewing them for security implications.

Specialized Hardening

Many Unices have specialized hardening features that make it more difficult to exploit software vulnerabilities or to do so without leaving traces on the system and/or to show that the system is so hardened. Linux has been a popular platform for research in this area; even the National Security Agency (NSA) has released code to implement its strict security requirements under Linux. Here we outline two of the most popular Linux hardening packages. Other such packages exist for Linux and other Unices, some of which use innovative techniques such as virtualization to isolate sensitive data, but they are not covered here.

GRSec/PAX

The grsecurity package provides several major security enhancements for Linux. Perhaps the primary benefit is the flexible policies that define fine-grained permissions it can control. This role-based access control capability is especially powerful when coupled with grsecurity's ability to monitor system activity over a period of time and generate a minimum set of privileges for all users. Additionally, through the PAX subsystem, grsecurity manipulates program memory to make it very difficult to exploit many kinds of security vulnerabilities. Other benefits include a very robust auditing capability and other features that strengthen existing security features, such as `chroot` jails.

SELinux

Security Enhanced Linux, or SELinux, is a package developed by the NSA. It adds Mandatory Access Control, or MAC, and related concepts to Linux. MAC involves assigning security attributes as well as system resources such as files and memory to users. When a user attempts to read, write, execute, or perform any other action on a system resource, the security attributes of the user and the resource are both used to determine whether the action is allowed, according to the security policies configured for the system.

Systems Management Security

After hardening a Unix host from network-borne attacks and hardening it from attacks performed by an authorized user of the machine, we will take a look at a few systems management issues. These topics arguably fall outside the purview of security as such; however, by taking certain considerations into account, systems management can both improve and simplify the work of securing a Unix system.

Account Management

User accounts can be thought of as keys to the "castle" of a system. As users require access to the system, they must be issued keys, or accounts, so they can use it. When a user no longer requires access to the system, her key should be taken away or at least disabled.

This sounds simple in theory, but account management in practice is anything but trivial. In all but the smallest environments, it is infeasible to manage user accounts without a centralized account directory where necessary changes can be made and propagated to every server on the network. Through PAM, BSD authentication, and other mechanisms, modern Unices support LDAP, SQL databases, Windows NT and Active Directory, Kerberos, and myriad other centralized account directory technologies.

Patching

Outdated software is perhaps the number-one cause of easily preventable security incidents. Choosing a modern Unix with a robust upgrade mechanism and history of timely updates, at least for security fixes, makes it easier to keep software up to date and secure from well-known exploits.

Backups

When all else fails—especially when attackers have successfully modified or deleted data in ways that are difficult or impossible to positively identify—good backups will save the day. When backups are robust, reliable, and accessible, they put a ceiling on the amount of damage an attacker can do. Unfortunately, good backups don't help if the greatest damage comes from disclosure of sensitive information; in fact, backups could exacerbate the problem if they are not taken and stored in a secure way.

3. Proactive Defense for Linux and Unix

As security professionals, we devote ourselves to defending systems from attack. However, it is important to understand the common tools, mindsets, and motivations that drive attackers. This knowledge can prove invaluable in mounting an effective defense against attack. It's also important to prepare for the possibility of a successful attack and to consider organizational issues so that you can develop a secure environment.

Vulnerability Assessment

A vulnerability assessment looks for security weaknesses in a system. Assessments have become an established best practice, incorporated into many standards and regulations. They can be network-centric or host-based.

Network-Based Assessment

Network-centric vulnerability assessment looks for security weaknesses a system presents to the network. Unix has a rich heritage of tools for performing network vulnerability assessments. Most of these tools are available on most Unix flavors.

nmap is a free, open source tool for identifying hosts on a network and the services running on those hosts. It's a powerful tool for mapping out the true services being provided on a network. It's also easy to get started with nmap.

Nessus is another free network security tool, though its source code isn't available. It's designed to check for and optionally verify the existence of known security vulnerabilities. It works by looking at various pieces of information about a host on the network, such as detailed version information about the operating system and any software providing services on the network. This information is compared to a database that lists vulnerabilities known to exist in certain software configurations. In many cases, Nessus is also capable of confirming a match in the vulnerability database by attempting an exploit; however, this is likely to crash the service or even the entire system.

Many other tools are available for performing network vulnerability assessments. Insecure.Org, the folks behind the nmap tool, also maintain a great list of security tools [8].

Host-Based Assessment

Several tools can examine the security settings of a system from a host-based perspective. These tools are designed to be run on the system that's being checked; no network connections are necessarily initiated. They check things such as file permissions and other insecure configuration settings on Unix systems.

One such tool, *lynis*, is available for various Linux distributions as well as some BSD variants. Another tool is the Linux Security Auditing Tool, or *lsat*. Ironically, lsat supports more versions of Unix than lynis does, including Solaris and AIX.

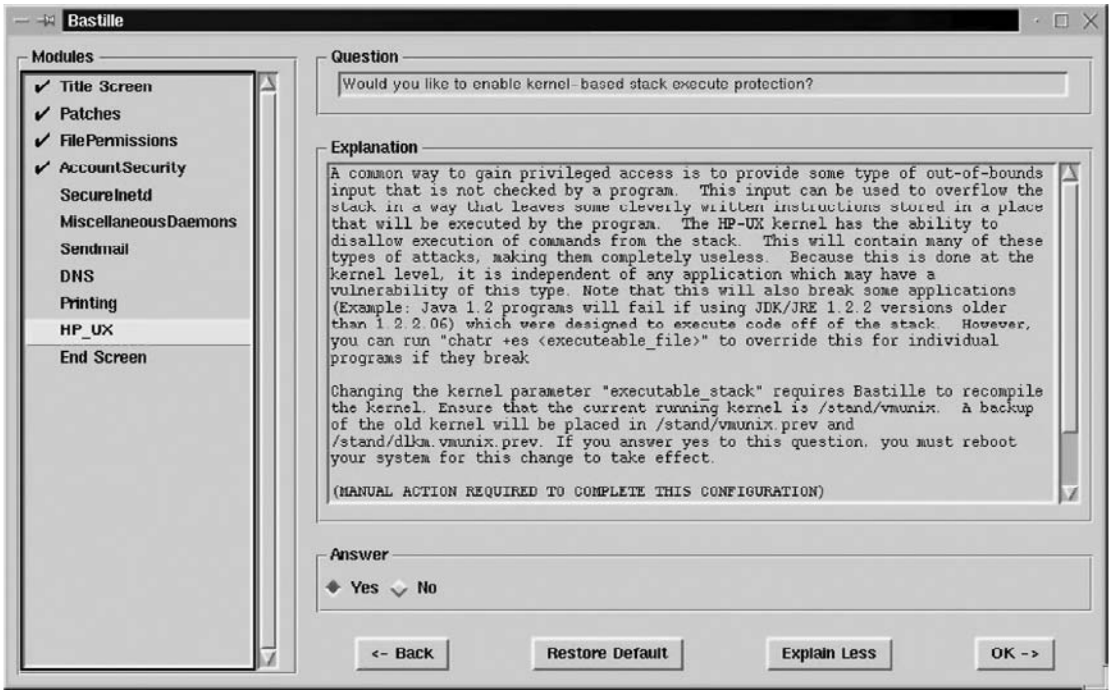
No discussion of host-based Unix security would be complete without mentioning *Bastille* (see Figure 6.5). Though lynis and lsat are pure auditing tools that report on the status of various security-sensitive host configuration settings, Bastille was designed to help remediate these issues. Recent versions have a reporting-only mode that makes Bastille work like a pure auditing tool.

Incident Response Preparation

Regardless of how hardened a Unix system is, there is always a possibility that an attacker—whether it's a worm, a virus, or a sophisticated custom attack—will successfully compromise the security of the system. For this reason, it is important to think about how to respond to a wide variety of security incidents.

Predefined Roles and Contact List

A fundamental part of incident response preparation is to identify the roles that various personnel will play in the response scenario. The manual, hands-on gestalt of Unix systems



Would you like to enable kernel-based stack execute protection?

```
A common way to gain privileged access is to provide some type of out-of-bounds
input that is not checked by a program.  This input can be used to overflow the
stack in a way that leaves some cleverly written instructions stored in a place
that will be executed by the program.  The HP-UX kernel has the ability to
disallow execution of commands from the stack.  This will contain many of these
types of attacks, making them completely useless.  Because this is done at the
kernel level, it is independent of any application which may have a
vulnerability of this type. Note that this will also break some applications
(Example: Java 1.2 programs will fail if using JDK/JRE 1.2.2 versions older
than 1.2.2.06) which were designed to execute code off of the stack.  However,
you can run "chatr +es <executeable_file>" to override this for individual
programs if they break

Changing the kernel parameter "executable_stack" requires Bastille to recompile
the kernel. Ensure that the current running kernel is /stand/vmunix.  A backup
of the old kernel will be placed in /stand/vmunix.prev and
/stand/dlkm.vmunix.prev. If you answer yes to this question, you must reboot
your system for this change to take effect.

(MANUAL ACTION REQUIRED TO COMPLETE THIS CONFIGURATION)
```

[✓ Yes] No

Figure 6.5: Bastille screenshot.

administration has historically forced Unix systems administrators to be familiar with all aspects of the Unix systems they manage. These should clearly be on the incident response team. Database, application, backup, and other administrators should be on the team as well, at least as secondary personnel that can be called on as necessary.

Simple Message for End Users

Incident response is a complicated process that must deal with conflicting requirements to bring the systems back online while ensuring that any damage caused by the attack—as well as whatever security flaws were exploited to gain initial access—is corrected. Often, end users without incident response training are the first to handle a system after a security incident has been identified. It is important that these users have clear, simple instructions in this case, to avoid causing additional damage or loss of evidence. In most situations, it is appropriate to simply unplug a Unix system from the network as soon as a compromise of its security is confirmed. It should not be used, logged onto, logged off from, turned off, disconnected from electrical power, or otherwise tampered with in any way. This simple action has the best chance, in most cases, to preserve the status of the incident for further investigation while minimizing the damage that could ensue.

Blue Team/Red Team Exercises

Any incident response plan, no matter how well designed, must be practiced to be effective. Regularly exercising these plans and reviewing the results are important parts of incident response preparation. A common way of organizing such exercises is to assign some personnel (the Red Team) to simulate a successful attack, while other personnel (the Blue Team) are assigned to respond to that attack according to the established incident response plan. These exercises, referred to as Red Team/Blue Team exercises, are invaluable for testing incident response plans. They are also useful in discovering security weaknesses and in fostering a sense of *esprit des corps* among the personnel involved.

Organizational Considerations

Various organizational and personnel management issues can also impact the security of Unix systems. Unix is a complex operating system. Many different duties must be performed in the day-to-day administration of Unix systems. Security suffers when a single individual is responsible for many of these duties; however, that is commonly the skill set of Unix system administration personnel.

Separation of Duties

One way to counter the insecurity of this situation is to force different individuals to perform different duties. Often, simply identifying independent functions, such as backups and log monitoring, and assigning appropriate permissions to independent individuals is enough.

Log management, application management, user management, system monitoring, and backup operations are just some of the roles that can be separated.

Forced Vacations

Especially when duties are appropriately separated, unannounced forced vacations are a powerful way to bring fresh perspectives to security tasks. It's also an effective deterrent to internal fraud or mismanagement of security responsibilities. A more robust set of requirements for organizational security comes from the Information Security Management Maturity Model, including its concepts of transparency, partitioning, separation, rotation, and supervision of responsibilities [9].

References

[1] Microsoft. Glossary of Networking Terms for Visio IT Professionals, http://technet.microsoft.com/en-us/library/cc751329.aspx#XSLTsection142121120120; n.d. [accessed September 22, 2008, from Microsoft TechNet].

[2] The Open Group. The Single Unix Specification, www.unix.org/what_is_unix/single_unix_specification.html; n.d. [accessed September 22, 2008, from What Is Unix].

[3] Raymond ES. The Cathedral and the Bazaar, www.catb.org/esr/writings/cathedral-bazaar/cathedral-bazaar/index.html; September 11, 2000 [accessed September 22, 2008, from Eric S. Raymond's homepage].

[4] Lundqvist A. Image:Gldt, http://en.wikipedia.org/wiki/Image:Gldt.svg; May 12, 2008 [accessed October 6, 2008, from Wikipedia].

[5] Richard Stevens W. Advanced Programming in the UNIX Environment. Reading: Addison-Wesley; 1992.

[6] Richard Stevens W. Advanced Programming in the UNIX Environment. Reading: Addison-Wesley; 1992.

[7] Schneier B. Real-World Passwords, www.schneier.com/blog/archives/2006/12/realworld_passw.html; December 14, 2006 [accessed October 9, 2008, from Schneier on Security].

[8] Insecure.Org. Top 100 Network Security Tools, http://sectools.org; 2008 [accessed October 9, 2008].

[9] ISECOM. Security Operations Maturity Architecture, www.isecom.org/soma; 2008 [accessed October 9, 2008, from ISECOM].

[10] Hutton M. Image: Unix History, retrieved October 6, 2008, from Wikipedia: http://enwikipedia.org/wiki/Image:Unix_history-simple.svg; July 9, 2008.

CHAPTER 7

Internet Security

Jesse Walker
Intel Corporation

The Internet, and all its accompanying complications, has become integral to our lives. The security problems besetting the Internet are legendary and have been daily annoyances to many users. Given the Net's broad impact on our lives and the widespread security issues associated with it, it is worthwhile understanding what can be done to improve the immunity of our communications from attack.

The Internet can serve as a laboratory for studying network security issues; indeed, we can use it to study nearly every kind of security issue. We will pursue only a modest set of questions related to this theme. The goal of this chapter is to understand how cryptography can be used to address some of the security issues besetting communications protocols. To do so, it will be helpful to first understand the Internet architecture. After that we will survey the types of attacks that are possible against communications. With this background we will be in a position to understand how cryptography can be used to preserve the confidentiality and integrity of messages.

Our goal is modest. It is only to describe the network architecture and its cryptographic-based security mechanisms sufficiently to understand some of the major issues confronting security systems designers and to appreciate some of the major design decisions they have to make to address these issues.

1. Internet Protocol Architecture

The Internet was designed to create standardized communication between computers. Computers communicate by exchanging messages. The Internet supports message exchange through a mechanism called *protocols*. Protocols are very detailed and stereotyped rules explaining exactly how to exchange a particular set of messages. Each protocol is defined as a set of finite state automata and a set of message formats. Each protocol specification defines one automaton for sending a message and another for receiving a message. The

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00007-5

automata specify the message timing; they play the role of grammar, indicating whether any particular message is meaningful or is interpreted by the receiver as gibberish. The protocol formats restrict the information that the protocol can express.

Security has little utility as an abstract, disembodied concept. What the word *security* should mean depends very much on the context in which it is applied. The architecture, design, and implementation of a system each determine the kind of vulnerabilities and opportunities for exploits that exist and which features are easy or hard to attack or defend.

It is fairly easy to understand why this is true. An attack on a system is an attempt to make the system act outside its specification. An attack is different from "normal" bugs that afflict computers and that occur through random interactions between the system's environment and undetected flaws in the system architecture, design, or implementation. An attack, on the other hand, is an explicit and systematic attempt by a party to search for flaws that make the computer act in a way its designers did not intend.

Computing systems consist of a large number of blocks or modules assembled together, each of which provides an intended set of functions. The system architecture hooks the modules together through *interfaces*, through which the various modules exchange information to activate the functions provided by each module in a coordinated way. An attacker exploits the architecture to compromise the computing system by interjecting inputs into these interfaces that do not conform to the specification for inputs of a specific module. If the targeted module has not been carefully crafted, unexpected inputs can cause it to behave in unintended ways. This implies that the security of a system is determined by its decomposition into modules, which an adversary exploits by injecting messages into the interfaces the architecture exposes. Accordingly, no satisfying discussion of any system is feasible without an understanding of the system architecture. Our first goal, therefore, is to review the architecture of the Internet communication protocols in an effort to gain a deeper understanding of its vulnerabilities.

Communications Architecture Basics

Since communication is an extremely complex activity, it should come as no surprise that the system components providing communication decompose into modules. One standard way to describe each communication module is as a black box with a well-defined service interface. A minimal communications service interface requires four primitives:

- A *send* primitive, which an application using the communications module uses to send a message via the module to a peer application executing on another networked device. The *send* primitive specifies a message payload and a destination. The communication module responding to the *send* transmits the message to the specified destination, reporting its requester as the message source.

- A *confirm* primitive, to report that the module has sent a message to the designated destination in response to a *send* request or to report when the message transmission failed, along with any failure details that might be known. It is possible to combine the *send* and *confirm* primitives, but network architectures rarely take this approach. The *send* primitive is normally defined to allow the application to pass a message to the communications module for transmission by transferring control of a buffer containing the message. The *confirm* primitive then releases the buffer back to the calling application when the message has indeed been sent. This scheme effects "a conservation of buffers" and enables the communications module and the application using it to operate in parallel, thus enhancing the overall communication performance.
- A *listen* primitive, which the receiving application uses to provide the communications module with buffers into which it should put messages arriving from the network. Each buffer the application posts must be large enough to receive a message of the maximum expected size.
- A *receive* primitive, to deliver a received message from another party to the receiving application. This releases a posted buffer back to the application and usually generates a signal to notify the application of message arrival. The released buffer contains the received message and the (alleged) message source.

Sometimes the *listen* primitive is replaced with a *release* primitive. In this model the receive buffer is owned by the receiving communications module instead of the application, and the application must recycle buffers containing received messages back to the communication module upon completion. In this case the buffer size selected by the receiving module determines the maximum message size. In a moment we will explain how network protocols work around this restriction.

It is customary to include a fifth service interface primitive for communications modules:

- A status primitive, to report diagnostic and performance information about the underlying communications. This might report statistics, the state of active associations with other network devices, and the like.

Communications is effected by providing a communications module black box on systems, connected by a signaling medium. The medium connecting the two devices constitutes the network communications path. The media can consist of a direct link between the devices or, more commonly, several intermediate relay systems between the two communicating endpoints. Each relay system is itself a communicating device with its own communications module, which receives and then forward messages from the initiating system to the destination system.

Under this architecture, a message is transferred from an application on one networked system to an application on a second networked system as follows:

First the application sourcing the message invokes the *send* primitive exported by its communications module. This causes the communications module to (attempt) to transmit the message to a destination provided by the application in the *send* primitive.

The communications module encodes the message onto the network's physical medium representing a link to another system. If the communications module implements a *best-effort* message service, it generates the *confirm* primitive as soon as the message has been encoded onto the medium. If the communication module implements a *reliable* message service, the communication delays generation of the *confirm* until it receives an acknowledgment from the message destination. If it has not received an acknowledgment from the receiver after some period of time, it generates a *confirm* indicating that the message delivery failed.

The encoded message traverses the network medium and is placed into a buffer by the receiving communications module of another system attached to the medium. This communications module examines the destination. The module then examines the destination specified by the message. If the module's local system is not the destination, the module reencodes the message onto the medium representing another link; otherwise the module uses the *deliver* primitive to pass the message to the receiving application.

Getting More Specific

This stereotyped description of networked communications is overly simplified. Communications are actually torturously more difficult in real network modules. To tame this complexity, communications modules are themselves partitioned further into layers, each providing a different networking function. The Internet decomposes communications into five layers of communications modules:

- The PHY layer
- The MAC layer
- The network layer
- The transport layer
- The sockets layer

These layers are also augmented by a handful of cross-layer coordination modules. The Internet depends on the following cross-layer modules:

- ARP
- DHCP
- DNS
- ICMP
- Routing

An application using networking is also part of the overall system design, and the way it uses the network has to be taken into consideration to understand system security.

We next briefly describe each of these in turn.

The PHY Layer

The PHY (pronounced *fie*) layer is technically not part of the Internet architecture per se, but Ethernet jacks and cables, modems, Wi-Fi adapters, and the like represent the most visible aspect of networking, and no security treatment of the Internet can ignore the PHY layer entirely.

The PHY layer module is medium dependent, with a different design for each type of medium: Ethernet, phone lines, Wi-Fi, cellular phone, OC-48, and the like are based on different PHY layer designs. It is the job of the PHY layer to translate between digital bits as represented on a computing device and the analog signals crossing the specific physical medium used by the PHY. This translation is a physics exercise.

To send a message, the PHY layer module encodes each bit of each message from the sending device as a media-specific signal, representing the bit value 1 or 0. Once encoded, the signal propagates along the medium from the sender to the receiver. The PHY layer module at the receiver decodes the medium-specific signal back into a bit.

It is possible for the encoding step at the transmitting PHY layer module to fail, for a signal to be lost or corrupted while it crosses the medium, and for the decoding step to fail at the receiving PHY layer module. It is the responsibility of higher layers to detect and recover from these potential failures.

The MAC Layer

Like the PHY layer, the MAC (pronounced *mack*) layer is not properly a part of the Internet architecture, but no satisfactory security discussion is possible without considering it. The MAC module is the "application" that uses and controls a particular PHY layer module. A MAC layer is always designed in tandem with a specific PHY (or vice versa), so a PHY-MAC pair together is often referred to as the *data link* layer.

MAC is an acronym for *media access control*. As its name suggests, the MAC layer module determines when to send and receive *frames*, which are messages encoded in a media-specific format. The job of the MAC is to pass frames over a link between the MAC layer modules on different systems.

Although not entirely accurate, it is useful to think of a MAC module as creating *links*, each of which is a communication channel between different MAC modules. It is further useful to distinguish physical links and virtual links. A *physical link* is a direct point-to-point channel between the MAC layers in two endpoint devices. A *virtual link* can be thought of as a shared medium to which more than two devices can connect at the same time. There are no physical endpoints per se; the medium acts as though it is multiplexing links between each pair of attached devices. Some media such as Ethernet are implemented as physical point-to-point links but act more like virtual links in that more than a single destination is reachable via the link. This is accomplished by MAC layer switching, which is also called *bridging*. Timing requirements for coordination among communicating MAC layer modules make it difficult to build worldwide networks based on MAC layer switching, however.

A MAC frame consists of a header and a data payload. The frame header typically specifies information such as the source and destination for the link endpoints. Devices attached to the medium via their MAC + PHY modules are identified by *MAC addresses*. Each MAC module has its own MAC address assigned by its manufacturer and is supposed to be a globally unique identifier. The destination MAC address in a frame allows a particular MAC module to identify frames intended for it, and the destination address allows it to identify the purported frame source. The frame header also usually includes a preamble, which is a set of special PHY timing signals used to synchronize the interpretation of the PHY layer data signals representing the frame bits.

The payload portion of a frame is the data to be transferred across the network. The maximum payload size is always fixed by the medium type. It is becoming customary for most MACs to support a maximum payload size of 1500 bytes = 12,000 bits, but this is not universal. The maximum fixed size allows the MAC to make efficient use of the underlying physical medium. Since messages can be of an arbitrary length exceeding this fixed size, a higher-layer function is needed to partition messages into segments of the appropriate length.

As we have seen, it is possible for bit errors to creep into communications as signals representing bits traverse the PHY medium. MAC layers differ a great deal in how they respond to errors. Some PHY layers, such as the Ethernet PHY, experience exceedingly low error rates, and for this reason, the MAC layers for these PHYs make no attempt to more than detect errors and discard the mangled frames. Indeed, with these MACs it is less expensive for the Internet to resend message segments at a higher layer than at the MAC layer. These are called *best-effort MACs*. Others, such as the Wi-Fi MAC, experience high error rates due to the shared nature of the channel and natural interference among radio sources, and experience has shown that these MACs can deliver better performance by retransmitting damaged or lost frames. It is customary for most MAC layers to append a checksum computed over the entire frame, called a *frame check sequence* (FCS). The FCS allows the receiver to detect bit errors accumulated due to random noise and other physical phenomena during transmission and due to decoding errors. Most MACs discard frames with FCS errors. Some MAC layers also perform error correction on the received bits to remove random bit errors rather than relying on retransmissions.

The Network Layer

The purpose of the network layer module is to represent messages in a media-independent manner and forward them between various MAC layer modules representing different links. The media-independent message format is called an *Internet Protocol*, or *IP, datagram*. The network layer implements the IP layer and is the lowest layer of the Internet architecture per se.

As well as providing media independence, the network layer provides a vital forwarding function that works even for a worldwide network like the Internet. It is impractical to form a link directly between each communicating system on the planet; indeed, the cabling costs alone are prohibitive—no one wants billions, or even dozens, of cables connecting their computer to other computers—and too many MAC + PHY interfaces can quickly exhaust the power budget for a single computing system. Hence, each machine is attached by a small number of links to other devices, and some of the machines with multiple links comprise a *switching fabric*. The computing systems constituting the switching fabric are called *routers*.

The forwarding function supported by the network layer module is the key component of a router and works as follows: When a MAC module receives a frame, it passes the frame payload to the network layer module. The payload consists of an *IP datagram*, which is the media-independent representation of the message. The receiving network layer module examines the datagram to see whether to deliver it locally or to pass it on toward the datagram's ultimate destination. To accomplish the latter, the network layer module consults a *forwarding table* to identify some neighbor router closer to the ultimate destination than itself. The forwarding table also identifies the MAC module to use to communicate with the selected neighbor and passes the datagram to that MAC layer module. The MAC module in turn retransmits the datagram as a frame encoded for its medium across its link to the neighbor. This process happens recursively until the datagram is delivered to its ultimate destination.

The network layer forwarding function is based on *IP addresses*, a concept that is critical to understanding the Internet architecture. An IP address is a media-independent name for one of the MAC layer modules within a computing system. Each IP address is structured to represent the "location" of the MAC module within the entire Internet. This notion of location is relative to the graph comprising routers and their interconnecting links, called the *network topology*, not to actual geography. Since this name represents a location, the forwarding table within each IP module can use the IP address of the ultimate destination as a sort of signpost pointing at the MAC module with the greatest likelihood of leading to the ultimate destination of a particular datagram.

An IP address is different from the corresponding MAC address already described. A MAC address is a permanent, globally unique identifier, whereas an IP address can be dynamic due to device mobility; an IP address cannot be assigned by the equipment manufacturer, since a computing device can change locations frequently. Hence, IP addresses are administered and blocks allocated to different organizations with an Internet presence. It is common, for instance, for an Internet service provider (ISP) to acquire a large block of IP addresses for use by its customers.

An IP datagram has a structure similar to that of a frame: It consists of an IP header, which is "extra" overhead used to control the way a datagram passes through the Internet, and a data payload, which contains the message being transferred. The IP header indicates the ultimate source and destinations, represented as IP addresses.

The IP header format limits the size of an IP datagram payload to 64K ($2^{16} = 65{,}536$) bytes. It is common to limit datagram sizes to the underlying media size, although datagrams larger than this do occur. This means that normally each MAC layer frame can carry a single IP datagram as its data payload. IP version 4, still the dominant version deployed on the Internet today, allows fragmentation of larger datagrams, to split large datagrams into chunks small enough to fit the limited frame size of the underlying MAC layer medium. IPv4 reassembles any fragmented datagrams at the ultimate destination.

Network layer forwarding of IP datagrams is best effort, not reliable. Network layer modules along the path taken by any message can lose and reorder datagrams. It is common for the network layer in a router to recover from congestion—that is, when the router is overwhelmed by more receive frames than it can process—by discarding late-arriving frames until the outer has caught up with its forwarding workload. The network layer can reorder datagrams when the Internet topology changes, because a new path between source and destination might be shorter or longer than an old path, so datagrams in flight before the change can arrive after frames sent after the change. The Internet architecture delegates recovery from these problems to high-layer modules.

The Transport Layer

The transport layer is implemented by TCP and similar protocols. Not all transport protocols provide the same level of service as TCP, but a description of TCP will suffice to help us understand the issues addressed by the transport layer. The transport layer provides a multitude of functions.

First, the transport layer creates and manages instances of two-way channels between communication endpoints. These channels are called *connections*. Each connection represents a virtual endpoint between a pair of communication endpoints. A connection is named by a pair of IP addresses and *port numbers*. Two devices can support simultaneous connections using different port numbers for each connection. It is common to differentiate applications on the same host through the use of port numbers.

A second function of the transport layer is to support delivery of messages of arbitrary length. The 64K byte limit of the underlying IP module is too small to carry really large messages, and the transport layer module at the message source chops messages into pieces called *segments* that are more easily digestible by lower-layer communications modules. The segment size is negotiated between the two transport endpoints during connection setup. The segment size is chosen by discovering the smallest maximum frame size supported by any MAC + PHY link on the path through the Internet used by the connection setup messages. Once this is known, the transmitter typically partitions a large message into segments no larger than this size, plus room for an IP header. The transport layer module passes each segment to the network layer module, where it becomes the payload for a single IP datagram. The destination network layer module extracts the payload from the IP datagram and passes it to the transport layer module, which interprets the information as a message segment. The destination transport reassembles this into the original message once all the necessary segments arrive.

Of course, as noted, MAC frames and IP datagrams can be lost in transit, so some segments can be lost. It is the responsibility of the transport layer module to detect this loss and retransmit the missing segments. This is accomplished by a sophisticated acknowledgment algorithm defined by the transport layer. The destination sends a special acknowledgment message, often piggybacked with a data segment being sent in the opposite direction, for each segment that arrives. Acknowledgments can be lost as well, and if the message source does not receive the acknowledgment within a time window, the source retransmits the unacknowledged segment. This process is repeated some number of times, and if the failure continues, the network layer tears down the connection because it cannot fulfill its reliability commitment.

One reason for message loss is congestion at routers, something blind retransmission of unacknowledged segments will only exacerbate. The network layer is also responsible for implementing congestion control algorithms as part of its transmit function. TCP, for instance, lowers its transmit rate whenever it fails to receive an acknowledgment message in time, and it slowly increases its rate of transmission until another acknowledgment is lost. This allows TCP to adapt to congestion in the network, helping to minimize frame loss.

It can happen that segments arrive at the destination out of order, since some IP datagrams for the same connection could traverse the Internet through different paths due to dynamic changes in the underlying network topology. The transport layer is responsible for delivering the segments in the order sent, so the receiver caches any segments that arrive out of order prior to delivery. The TCP reordering algorithm is closed tied to the acknowledgment and congestion control scheme so that the receiver never has to buffer too many out-of-order received segments and the sender not too many sent but unacknowledged segments.

Segment data arriving at the receiver can be corrupted due to undetected bit errors on the data link and copy errors within routers and the sending and receiving computing systems. Accordingly, all transport layers use a checksum algorithm called a *cyclic redundancy check* (CRC) to detect such errors. The receiving transport layer module typically discards segments with errors detected by the CRC algorithm, and recovery occurs through retransmission by the receiver when it fails to receive an acknowledgment from the receiver for a particular segment.

The Sockets Layer

The top layer of the Internet, the sockets layer, does not *per se* appear in the architecture at all. The sockets layer provides a set of sockets, each of which represents a logical communications endpoint. An application can use the sockets layer to create, manage, and destroy connection instances using a socket as well as send and receive messages over the connection. The sockets layer has been designed to hide much of the complexity of utilizing the transport layer. The sockets layer has been highly optimized over the years to deliver as much performance as possible, but it does impose a performance penalty. Applications with very demanding performance requirements tend to utilize the transport layer directly instead of through the sockets layer module, but this comes with a very high cost in terms of software maintenance.

In most implementations of these communications modules, each message is copied twice, at the sender and the receiver. Most operating systems are organized into user space, which is used to run applications, and kernel space, where the operating system itself runs. The sockets layer occupies the boundary between user space and kernel space. The sockets layer's *send* function copies a message from memory controlled by the sending application into a buffer controlled by the kernel for transmission. This copy prevents the application from changing a message it has posted to send, but it also permits the application and kernel to continue their activities in parallel, thus better utilizing the device's computing resources. The sockets layer invokes the transport layer, which partitions the message buffer into segments and passes the address of each segment to the network layer. The network layer adds its headers to form datagrams from the segments and invokes the right MAC layer module to transmit each datagram to its next hop. A second copy occurs at the boundary between the network layer and the MAC layer, since the data link must be able to asynchronously match transmit requests from the network layer to available transmit slots on the medium provided by its PHY. This process is reversed at the receiver, with a copy of datagrams across the MAC-network layer boundary and of messages between the socket layer and application.

Address Resolution Protocol

The network layer uses Address Resolution Protocol, or ARP, to translate IP addresses into MAC addresses, which it needs to give to the MAC layer in order to deliver frames to the appropriate destination.

The ARP module asks the question, “Who is using IP address *X*?” The requesting ARP module uses a request/response protocol, with the MAC layer broadcasting the ARP module’s requests to all the other devices on the same physical medium segment. A receiving ARP module generates a response only if its network layer has assigned the IP address to one of its MAC modules. Responses are addressed to the requester’s MAC address. The requesting ARP module inserts the response received in an address translation table used by the network layer to identify the next hop for all datagrams it forwards.

Dynamic Host Configuration Protocol

Remember that unlike MAC addresses, IP addresses cannot be assigned in the factory, because they are dynamic and must reflect a device’s current location within the Internet’s topology. A MAC module uses Dynamic Host Configuration Protocol, or DHCP, to acquire an IP address for itself, to reflect the device’s current location with respect to the Internet topology.

DHCP makes the request: “Please configure my MAC module with an IP address.” When one of a device’s MAC layer modules connects to a new medium, it invokes DHCP to make this request. The associated DHCP module generates such a request that conveys the MAC address of the MAC module, which the MAC layer module broadcasts to the other devices attached to the same physical medium segment. A DHCP server responds with a unicast DHCP response binding an IP address to the MAC address. When it receives the response, the requesting DHCP module passes the assigned IP address to the network layer to configure in its address translation table.

In addition to binding an IP address to the MAC module used by DHCP, the response also contains a number of network configuration parameters, including the address of one or more routers, to enable reaching arbitrary destinations, the maximum datagram size supported, and the addresses of other servers, such as DNS servers, that translate human-readable names into IP addresses.

Domain Naming Service

IP and MAC addresses are efficient means for identifying different network interfaces, but human beings are incapable of using these as reliably as computing devices can. Instead, human beings rely on names to identify the computing devices with which they want to communication. These names are centrally managed and called *domain names*. The Domain Naming Service, or DNS, is a mechanism for translating human-readable names into IP addresses.

The translation from human-readable names to IP addresses happens within the socket layer module. An application opens a socket with the name of the intended destination. As the first step of opening a connection to that destination, the socket sends a request to a DNS server, asking the server to translate the name into an IP address. When the server responds, the socket can open the connection to the right destination, using the IP address provided.

It is becoming common for devices to register their IP addresses under their names with DNS once DHCP has completed. This permits other devices to locate the registering device so that they can send messages to it.

Internet Control Message Protocol

Internet Control Message Protocol, or ICMP, is an important diagnostic tool for troubleshooting the Internet. Though ICMP provides many specialized message services, three are particularly important:

- *Ping*. Ping is a request/response protocol designed to determine reachability of another IP address. The requester sends a ping request message to a designated IP address. If it's delivered, the destination IP address sends a ping response message to the IP address that sourced the request. The responding ICMP module copies the contents of the ping request into the ping response so that the requester can match responses to requests. The requester uses pings to measure the roundtrip time to a destination.
- *Traceroute*. Traceroute is another request/response protocol. An ICMP module generates a traceroute request to discover the path it is using to traverse the Internet to a destination IP address. The requesting ICMP module transmits a destination. Each router that handles the traceroute request adds a description of its own IP address that received the message and then forwards the updated traceroute request. The destination sends all this information back to the message source in a traceroute response message.
- *Destination unreachable*. When a router receives a datagram for which it has no next hop, it generates a "destination unreachable" message and sends it back to the datagram source. When the message is delivered, the ICMP module marks the forwarding table of the message source so that its network layer will reject further attempts to send messages to the destination IP address. An analogous process happens at the ultimate destination when a message is delivered to a network layer, but the application targeted to receive the message is no longer on line. The purpose of "destination unreachable" messages is to suppress messages that will never be successfully delivered, to reduce network congestion.

Routing

The last cross-layer module we'll discuss is *routing*. Routing is a middleware application to maintain the forwarding tables used by the network layer. Each router advertises itself by periodically broadcasting "hello" messages through each of its MAC interfaces. This allows routers to discover the presence or loss of all neighboring routers, letting them construct the one-hop topology of the part of the Internet directly visible through their directly attached media. The routing application in a router then uses a sophisticated gossiping mechanism to exchange this mechanism with their neighbors. Since some of a router's neighbors are not its own direct neighbors, this allows each router to learn the two-hop topology of the Internet. This process repeats recursively until each router knows the entire topology of the Internet. The cost of using each link is part of the information gossiped. A routing module receiving this information uses all of it to compute a lowest-cost route to each destination. Once this is accomplished, the routing module reconfigures the forwarding table maintained by its network layer module. The routine module updates the forwarding table whenever the Internet topology changes, so each network layer can make optimal forwarding decisions in most situations and at the very worst at least reach any other device that is also connected to the Internet.

There are many different routing protocols, each of which are based on different gossiping mechanisms. The most widely deployed routing protocol between different administrative domains within the Internet is the Border Gateway Protocol (BGP). The most widely deployed routing protocols within wired networks controlled by a single administrative domain are OSPF and RIP. AODV, OLSR, and TBRPF are commonly used in Wi-Fi meshes. Different routing protocols are used in different environments because each one addresses different scaling and administrative issues.

Applications

Applications are the ultimate reason for networking, and the Internet architecture has been shaped by applications' needs. All communicating applications define their own language in which to express what they need to say. Applications generally use the sockets layer to establish communication channels, which they then use for their own purposes.

It is worth emphasizing that since the network modules have been designed to be a generic communications vehicle, that is, designed to meet the needs of all (or at least most) applications, it is rarely meaningful for the network to attempt to make statements on behalf of the applications. There is widespread confusion on this point around authentication and key management, which are the source of many exploitable security flaws.

2. An Internet Threat Model

Now that we have reviewed the architecture of the Internet protocol suite, it is possible to constructively consider security issues it raises. Before doing so, let's first set the scope of the discussion.

There are two general approaches to attacking a networked computer. The first is to compromise one of the communicating parties so that it responds to queries with lies or otherwise communicates in a manner not foreseen by the system designers of the receiver.

For example, it has become common to receive email with virus-infected attachments, whereby opening the attachment infects the receiver with the virus. These messages typically are sent by a machine that has already been compromised, so the sender is no longer acting as intended by the manufacturer of the computing system. Problems of this type are called *Byzantine failures*, named after the Byzantine Generals problem.

The Byzantine Generals problem imagines several armies surrounding Byzantium. The generals commanding these armies can communicate only by exchanging messages transported by couriers between them. Of course the couriers can be captured and the messages replaced by forgeries, but this is not really the issue, since it is possible to devise message schemes that detect lost messages or forgeries. All the armies combined are sufficient to overwhelm the defenses of Byzantium, but if even one army fails to participate in a coordinated attack, the armies of Byzantium have sufficient strength to repulse the attack. Each general must make a decision as to whether to participate in an attack on Byzantium at dawn or withdraw to fight another day. The question is how to determine the veracity of the messages received on which the decision to attack will be made—that is, whether it is possible to detect that one or more generals have become traitors so will say their armies will join the attack when in fact they plan to hold back so that their allies will be slaughtered by the Byzantines.

Practical solutions addressing Byzantine failures fall largely within the purview of platform rather than network architecture. For example, since viruses infect a platform by buffer overrun attacks, platform mechanisms to render buffer overrun attacks futile are needed. Secure logging, to make an accurate record of messages exchanged, is a second deterrent to these sorts of attacks; the way to accomplish secure logging is usually a question of platform design. Most self-propagating viruses and worms utilize the Internet to propagate, but they do not utilize any feature of the Internet architecture per se for their success. The success of these attacks instead depends on the architecture, design, implementation, and policies of the receiving system. Although these sorts of problems are important, we will rarely focus on security issues stemming from Byzantine failures.

What will instead be the focus of the discussion are attacks on the messages exchanged between computers themselves. As we will see, even with this more limited scope, there are plenty of opportunities for things to go wrong.

The Dolev–Yao Adversary Model

Security analyses of systems traditionally begin with a model of the attacker, and we follow this tradition. Dolev and Yao formulated the standard attack model against messages exchanged over a network. The *Dolev–Yao model* makes the following assumptions about an attacker:

- *Eavesdrop*. An adversary can listen to any message exchanged through the network.
- *Forge*. An adversary can create and inject entirely new messages into the datastream or change messages in flight; these messages are called *forgeries*.
- *Replay*. A special type of forgery, called a *replay*, is distinguished. To replay a message, the adversary resends legitimate messages that were sent earlier.
- *Delay and rush*. An adversary can delay the delivery of some messages or accelerate the delivery of others.
- *Reorder*. An adversary can alter the order in which messages are delivered.
- *Delete*. An adversary can destroy in-transit messages, either selectively or all the messages in a datastream.

This model assumes a very powerful adversary, and many people who do not design network security solutions sometime assert that the model grants adversaries an unrealistic amount of power to disrupt network communications. However, experience demonstrates that it is a reasonably realistic set of assumptions in practice; examples of each threat abound, as we will see. One of the reasons for this is that the environment in which the network operates is exposed; unlike memory or microprocessors or other devices comprising a computer, there is almost no assurance that the network medium will be deployed in a "safe" way. That is, it is comparatively easy for an attacker to anonymously access the physical network fabric, or at least the medium monitored to identify attacks against the medium and the networked traffic it carries. And since a network is intended as a generic communications vehicle, it becomes necessary to adopt a threat model that addresses the needs of all possible applications.

Layer Threats

With the Dolev–Yao model in hand, we can examine each of the architectural components of the Internet protocol suite for vulnerabilities. We next look at threats each component of the Internet architecture exposes through the prism of this model. The first Dolev–Yao assumption about adversaries is that they can eavesdrop on any communications.

Eavesdropping

An attacker can eavesdrop on a communications medium by connecting a receiver to the medium. Ultimately such a connection has to be implemented at the PHY layer because an adversary has to access some physical media somewhere to be able to listen to anything at all. This connection to the PHY medium might be legitimate, such as when an authorized device is compromised, or illegitimate, such as an illegal wiretap; it can be intentional, as when an eavesdropper installs a rogue device, or unintentional, such as a laptop with wireless capabilities that will by default attempt to connect to any Wi-Fi network within range.

With a PHY layer connection, the eavesdropper can receive the analog signals on the medium and decode them into bits. Because of the limited scope of the PHY layer function—there are no messages, only analog signals representing bits—the damage an adversary can do with only PHY layer functionality is rather limited. In particular, to make sense of the bits, an adversary has to impose the higher-layer frame and datagram formats onto the received bits. That is, any eavesdropping attack has to take into account at least the MAC layer to learn anything meaningful about the communications. Real eavesdroppers are more sophisticated than this: They know how to interpret the bits as a medium-specific encoding with regards to the frames that are used by the MAC layer. They also know how to extract the media-independent representation of datagrams conveyed within the MAC frames, as well as how to extract the transport layer segments from the datagrams, which can be reassembled into application messages.

The defenses erected against any threat give some insight into the perceived danger of the threat. People are generally concerned about eavesdropping, and it is easy to illicitly attach listening devices to most PHY media, but detection and removal of wiretaps has not evolved into a comparatively large industry. An apparent explanation of why this is so is that it is easier and more cost effective for an attacker to compromise a legitimate device on the network and configure it to eavesdrop than it is to install an illegitimate device. The evidence for this view is that the antivirus/antibot industry is gigantic by comparison.

There is another reason that an antiwiretapping industry has never developed for the Internet. Almost every MAC module supports a special mode of operation called *promiscuous mode*. A MAC module in promiscuous mode receives every frame appearing on the medium, not just the frames addressed to itself. This allows one MAC module to snoop on frames that are intended for other parties. Promiscuous mode was intended as a troubleshooting mechanism to aid network administrators in diagnosing the source of problems. However, it is also a mechanism that can be easily abused by anyone motivated to enable promiscuous mode.

Forgeries

A second Dolev–Yao assumption is that the adversary can forge messages. Eavesdropping is usually fairly innocuous compared to forgeries, because eavesdropping merely leaks information, whereas forgeries cause an unsuspecting receiver to take actions based on false information. Hence, the prevention or detection of forgeries is one of the central goals of network security mechanisms. Different kinds of forgeries are possible for each architectural component of the Internet. We will consider only a few for each layer of the Internet protocol suite, to give a taste for their variety and ingenuity.

Unlike the eavesdropping threat, where knowledge of higher layers is essential to any successful compromise, an attacker with only a PHY layer transmitter (and no higher-layer mechanisms) can disrupt communications by *jamming* the medium—that is, outputting noise onto the medium in an effort to disrupt communications. A jammer creates signals that do not necessarily correspond to any bit patterns. The goal of a pure PHY layer jammer is denial of service (DoS)—that is, to fill the medium so that no communications can take place.

Sometimes it is feasible to create a jamming device that is sensitive to the MAC layer formats above it, to selectively jam only some frames. Selective jamming requires a means to interpret bits received from the medium as a higher-layer frame or datagram, and the targeted frames to jam are recognized by some criterion, such as being sent from or to a particular address. So that it can enable its own transmitter before the frame has been entirely received by its intended destination, the jammer's receiver must recognize the targeted frames before they are fully transmitted. When this is done correctly, the jammer's transmitter interferes with the legitimate signals, thereby introducing bit errors in the legitimate receiver's decoder. This results in the legitimate receiver's MAC layer detecting the bit errors while trying to verify the frame check sequence, causing it to discard the frame. Selective jamming is harder to implement than continuous jamming, PHY layer jamming, but it is also much harder to detect, because the jammer's signal source transmits only when legitimate devices transmit as well, and only the targeted frames are disrupted. Successful selective jamming usually causes administrators to look for the source of the communications failure on one of the communicating devices instead of in the network for a jammer.

There is also a higher-layer analog to jamming, called *message flooding*. Denial of service (DoS) is also the goal of message flooding. The technique used by message flooding is to create and send messages at a rate high enough to exhaust some resource. It is popular today, for instance, for hackers to compromise thousands of unprotected machines, which they use to generate simultaneous messages to a targeted site. Examples of this kind of attack are to completely fill the physical medium connecting the targeted site to the Internet with network layer datagrams—this is usually hard or impossible—or to generate transport layer connection requests at a rate faster than the targeted site can respond. Other variants—request operations that lead to disk I/O or require expensive cryptographic operations—are also common. Message flooding attacks have the property that they are legitimate messages from authorized parties but simply timed so that collectively their processing exceeds the maximum capacity of the targeted system.

Let's turn away from resource-clogging forgeries and examine forgeries designed to cause a receiver to take an unintended action. It is possible to construct this type of forgery at any higher layer: forged frames, datagrams, network segments, or application messages.

To better understand how forgeries work, we need to more closely examine Internet "identities"—MAC addresses, IP addresses, transport port numbers, and DNS names—as well as the modules that use or support their use. The threats are a bit different at each layer.

Recall that each MAC layer module is manufactured with its own "hardware" address, which is supposed to be a globally unique identifier for the MAC layer module instance. The hardware address is configured in the factory into nonvolatile memory. At boot time the MAC address is transferred from nonvolatile memory into operational RAM maintained by the MAC module. A transmitting MAC layer module inserts the MAC address from RAM into each frame it sends, thereby advertising an "identity." The transmitter also inserts the MAC address of the intended receiver on each frame, and the receiving MAC layer matches the MAC address in its own RAM against the destination field in each frame sent over the medium. The receiver ignores the frame if the MAC addresses don't match and receives the frame otherwise.

In spite of this system, it is useful—even necessary sometimes—for a MAC module to change its MAC address. For example, sometimes a manufacturer recycles MAC addresses so that two different modules receive the same MAC address in the factory. If both devices are deployed on the same network, neither works correctly until one of the two changes its address. Because of this problem, all manufacturers provide a way for the MAC module to alter the address in RAM. This can always be specified by software via the MAC module's device driver, by replacing the address retrieved from hardware at boot time.

Since it can be changed, attacks will find it. A common attack in Wi-Fi networks, for instance, is for the adversary to put the MAC module of the attacking device into promiscuous mode, to receive frames from other nearby systems. It is usually easy to identify another client device from the received frames and extract its MAC address. The attacker then reprograms its own MAC module to transmit frames using the address of its victim. A goal of this attack is usually to "hijack" the session of a customer paying for Wi-Fi service; that is, the attacker wants free Internet access for which someone else has already paid. Another goal of such an attack is often to avoid attribution of the actions being taken by the attacker; any punishment for antisocial or criminal behavior will likely be attributed to the victim instead of the attacker because all the frames that were part of the behavior came from the victim's address.

A similar attack is common at the network layer. The adversary will snoop on the IP addresses appearing in the datagrams encoded in the frames and use these instead of their own IP addresses to source IP datagrams. This is a more powerful attack than that of utilizing only a MAC address, because IP addresses are global; an IP address is an Internet-wide locator, whereas a MAC address is only an identifier on the medium to which the device is physically connected.

Manipulation of MAC and IP addresses leads directly to a veritable menagerie of forgery attacks and enables still others. A very selective list of examples must suffice to illustrate the ingenuity of attackers.

- TCP uses sequence numbers as part of its reliability scheme. TCP is supposed to choose the first sequence number for a connection randomly. If an attacker can predict the first sequence number for a TCP connection, an attacker who spoofs the IP address of one of the parties to the connection can hijack the session by interjecting its own datagrams into the flow that use the correct sequence numbers. This desynchronizes the retry scheme for the device being spoofed, which then drops out from the conversation. This attack seems to have become relatively less common than other attacks over the past few years, since most TCP implementations have begun to utilize better random number generators to seed their sequence numbers.
- An attacker can generate an ARP response to any ARP request, thus claiming to use any requested IP address. This is a common method to hijack another machine's IP address; it is a very effective technique when the attacker has a fast machine and the victim machine responds more slowly.
- An attacker can generate DHCP response messages replying to DHCP requests. This technique is often used as part of a larger forgery, such as the evil twin attack, whereby an adversary masquerades as an access point for a Wi-Fi public hotspot. The receipt of DHCP response messages convinces the victim it is connecting to an access point operated by the legitimate hotspot.
- A variant is to generate a DHCP request with the hardware MAC address of another device. This method is useful when the attacker wants to ascribe action it takes over the Internet to another device.
- An attacker can impersonate the DNS server, responding to requests to resolve human-readable names into IP addresses. The IP address in the response messages point the victim to a site controlled by the attacker. This is becoming a common attack used by criminals attempting to commit financial fraud, such as stealing credit card numbers.

Replay

Replay is a special forgery attack. It occurs when an attacker records frames or datagrams and then retransmits them unchanged at a later time.

This might seem like an odd thing to do, but replay attacks are an especially useful way to attack stateful messaging protocols, such as a routing protocol. Since the goal of a routing protocol is to allow every router to know the current topology of the network, a replayed routing message can cause the routers receiving it to utilize out-of-date information.

An attacker might also respond to an ARP request sent to a sabotaged node or a mobile device that has migrated to another part of the Internet, by sending a replayed ARP response. This replay indicates the node is still present, thus masking the true network topology.

Replay is also often a valuable tool for attacking a message encryption scheme. By retransmitting a message, an attacker can sometimes learn valuable information from a message decrypted and then retransmitted without encryption on another link.

A primary use of replay, however, is to attack session startup protocols. Protocol startup procedures establish session state, which is used to operate the link or connection, and determine when some classes of failures occur. Since this state is not yet established when the session begins, startup messages replayed from prior instances of the protocol will fool the receiver into allocating a new session. This is a common DoS technique.

Delay and Rushing

Delay is a natural consequence of implementations of the Internet architecture. Datagrams from a single connection typically transit a path across the Internet in bursts. This happens because applications at the sender, when sending large messages, tend to send messages larger than a single datagram. The transport layer partitions these messages into segments to fit the maximum segment size along the path to the destination. The MAC tends to output all the frames together as a single blast after it has accessed the medium. Therefore, routers with many links can receive multiple datagram bursts at the same time. When this happens, a router has to temporarily buffer the burst, since it can output only one frame conveying a datagram per link at a time. Simultaneous arrival of bursts of datagrams is one source of congestion in routers. This condition usually manifests itself at the application by slow communications time over the Internet. Delay can also be introduced by routers intentionally, such as via traffic shaping.

There are several ways in which attackers can induce delays. We illustrate this idea by describing two different attacks. It is not uncommon for an attacker to take over a router, and when this happens, the attacker can introduce artificial delay, even when the router is uncongested. As a second example, attackers with bot armies can bombard a particular router with “filler” messages, the only purpose of which is to congest the targeted router.

Rushing is the opposite problem: a technique to make it appear that messages can be delivered sooner than can be reasonably expected. Attackers often employ rushing attacks by first hijacking routers that service parts of the Internet that are fairly far apart in terms of network topology. The attackers cause the compromised routers to form a *virtual link* between them. A virtual link emulates a MAC layer protocol but running over a transport layer connection between the two routers instead of a PHY layer. The virtual link, also called a *wormhole*, allows the routers to claim they are connected directly by a link and so are only one hop apart. The two compromised routers can therefore advertise the wormhole as a “low-cost” path between their respective regions of the Internet. The two regions then naturally exchange traffic through the compromised routers and the wormhole.

An adversary usually launches a rushing attack as a prelude to other attacks. By attracting traffic to the wormhole endpoints, the compromised routers can eavesdrop and modify the datagrams flowing through them. Compromised routers at the end of a wormhole are also an ideal vehicle for selective deletion of messages.

Reorder

A second natural event in the Internet is datagram *reordering*. The two most common reordering mechanisms are forwarding table updates and traffic-shaping algorithms. Reordering due to forwarding takes place at the network layer; traffic shaping can be applied at the MAC layer or higher.

The Internet reconfigures itself automatically as routers set up new links with neighboring routers and tear down links between routers. These changes cause the routing application on each affected router to send an update to its neighbors, describing the topology change. These changes are gossiped across the network until every router is aware of what happened. Each router receiving such an update modifies its forwarding table to reflect the new Internet topology.

Since the forwarding table updates take place asynchronously from datagram exchanges, a router can select a different forwarding path for each datagram between even the same two devices. This means that two datagrams sent in order at the message source can arrive in a different order at the destination, since a router can update its forwarding table between the selection of a next hop for different datagrams.

The second reordering mechanism is traffic shaping, which gets imposed on the message flow to make better use of the communication resources. One example is quality of service. Some traffic classes, such as voice or streaming video, might be given higher priority by routers than best-effort traffic, which constitutes file transfers. Higher-priority means the router will send datagrams carrying voice or video first while buffering the traffic longer. Endpoint systems also apply traffic-shaping algorithms in an attempt to make real-time applications work better, without gravely affecting the performance of applications that can wait for their data. Any layer of the protocol stack can apply traffic shaping to the messages it generates or receives.

An attacker can emulate reordering any messages it intercepts, but since every device in the Internet must recover from message reordering anyway, reordering attacks are generally useful only in very specific contexts. We will not discuss them further.

Message Deletion

Like reordering, *message deletion* can happen through normal operation of the Internet modules. A MAC layer will drop any frame it receives with an invalid frame check sequence. A network layer module will discard any datagram it receives with an IP header error.

A transport layer will drop any data segment received with a data checksum error. A router will drop perfectly good datagrams after receiving too many simultaneous bursts of traffic that lead to congestion and exhaustion of its buffers. For these reasons, TCP was designed to retransmit data segments in an effort to overcome errors.

The last class of attack possible with a Dolev–Yao adversary is message deletion. Two message deletion attacks occur frequently enough to be named: *black-hole attacks* and *gray-hole attacks*.

Black-hole attacks occur when a router deletes all messages it is supposed to forward. From time to time a router is misconfigured to offer a zero-cost route to every destination in the Internet. This causes all traffic to be sent to this router. Since no device can sustain such a load, the router fails. The neighboring routers cannot detect the failure rapidly enough to configure alternate routes, and they fail as well. This continues until a significant portion of the routers in the Internet fail, resulting in a black hole: Messages flow into the collapsed portion of the Internet and never flow out. A black-hole attack intentionally misconfigures a router. Black-hole attacks also occur frequently in small-scale sensor, mesh, and peer-to-peer file networks.

A gray-hole attack is a selective deletion attack. Targeted jamming is one type of selective message deletion attack. More generally, an adversary can discard any message it intercepts in the Internet, thereby preventing its ultimate delivery. An adversary intercepting and selectively deleting messages can be difficult to detect and diagnose, so is a powerful attack. It is normally accomplished via compromised routers.

A subtler, indirect form of message deletion is also possible through the introduction of *forwarding loops*. Each IP datagram header has a *time-to-live* (TTL) field, limiting the number of hops that a datagram can make. This field is set to 255 by the initiator and decremented by each router the datagram passes through. If a router decrements the TTL field to zero, it discards the datagram.

The reason for the TTL field is that the routing protocols that update the forwarding tables can temporarily cause forwarding loops because updates are applied asynchronously as the routing updates are gossiped through the Internet. For instance, if router A gets updated prior to router B, A might believe that the best path to some destination C is via B, whereas B believes the best route to C is via A as the next hop. Messages for C will ping-pong between A and B until one or both are updated with new topology information.

An attacker who compromises a router or forges its routing traffic can intentionally introduce forwarding routes. This causes messages addressed to the destinations affected by the forgery to circulate until the TTL field gets decremented to zero. These attacks are also difficult to detect, because all the routers are behaving according to their specifications, but messages are being mysteriously lost.

3. Defending Against Attacks on the Internet

Now that we have a model for thinking about the threats against communication and we understand how the Internet works, we can examine how its communications can be protected. Here we will explain how cryptography is used to protect messages exchanged between various devices on the Internet and illustrate the techniques with examples.

As might be expected, the techniques vary according to scenario. Methods that are effective for an active session do not work for session establishment. Methods that are required for session establishment are too expensive for an established session. It is interesting that similar methods are used at each layer of the Internet architecture for protecting a session and for session establishment and that each layer defines its own security protocols. Many find the similarity of security solutions at different layers curious and wonder why security is not centralized in a single layer. We will explain why the same mechanisms solve different problems at different layers of the architecture, to give better insight into what each is for.

Layer Session Defenses

A *session* is a series of one or more related messages. The easiest and most straightforward defenses protect the exchange of messages that are organized into sessions, so we will start with session-oriented defenses.

Cryptography, when used properly, can provide reliable defenses against eavesdropping. It can also be used to detect forgery and replay attacks, and the methods used also have some relevance to detecting reordering and message deletion attacks. We will discuss how this is accomplished and illustrate the techniques with *transport layer security* (TLS), IPsec, and 802.11i.

Defending Against Eavesdropping

The primary method used to defend against eavesdropping is encryption. Encryption was invented with the goal of making it infeasible for any computationally limited adversary to be able to learn anything useful about a message that cannot already be deduced by some other means, such as its length. Encryption schemes that appear to meet this goal have been invented and are in widespread use on the Internet. Here we will describe how they are used.

There are two forms of encryption: symmetric encryption, in which the same key is used to both encrypt and decrypt, and asymmetric encryption, in which encryption and decryption use distinct but related keys. The properties of each are different. Asymmetric encryption tends to be used only for applications related to session initiation and assertions about policy (although this is not universally true). The reason for this is that a single asymmetric key operation is generally too expensive to be applied to a message stream of arbitrary length. We therefore focus on symmetric encryption and how it is used by network security protocols.

A symmetric encryption scheme consists of three operations: *key generate*, *encrypt*, and *decrypt*. The key generate operation creates a *key,* which is a secret. The key generate procedure is usually application specific; we describe some examples of key generate operations in our discussion of session startup. Once generated, the key is used by the encrypt operation to transform *plaintext* messages—that is, messages that can be read by anyone—into *ciphertext*, which is messages that cannot be read by any computationally limited party who does not possess the key. The key is also used by the decrypt primitive to translate ciphertext messages back into plaintext messages.

There are two kinds of symmetric encryption algorithms. The first is type is called a *block cipher* and the second a *stream cipher*. Block and stream ciphers make different assumptions about the environment in which they operate, making each more effective than the other at different protocol layers.

A block cipher divides a message into chunks of a fixed size called *blocks* and encrypts each block separately. Block ciphers have the random access property, meaning that a block cipher can efficiently encrypt or decrypt any block utilizing an *initialization vector* in conjunction with the key. This property makes block ciphers a good choice for encrypting the content of MAC layer frames and network layer datagrams, for two reasons. First, the chunking behavior of a block cipher corresponds nicely to the packetization process used to form datagrams from segments and frames from datagrams. Second, and perhaps more important, the Internet architecture models the lower layers as "best-effort" services, meaning that it assumes that datagrams and frames are sent and then forgotten. If a transmitted datagram is lost due to congestion or bit error (or attack), it is up to the transport layer or application to recover. The random access property makes it easy to restart a block cipher anywhere it's needed in the data stream. Popular examples of block ciphers include AES, DES, and 3DES, used by Internet security protocols.

Block ciphers are used by the MAC and network layers to encrypt as follows: First, a block cipher mode of operation is selected. A block cipher itself encrypts and decrypts only single blocks. A mode of operation is a set of rules extending the encryption scheme from a single block to messages of arbitrary length. The most popular modes of operation used in the Internet are counter mode and cipher-block chaining (CBC) mode. Both require an initialization vector, which is a counter value for counter mode and a randomly generated bit vector for cipher-block chaining mode. To encrypt a message, the mode of operation first partitions the message into a sequence of blocks whose sizes equal that of the cipher's block size, padding if needed to bring the message length up to a multiple of the block size. The mode of operation then encrypts each block under the key while combining initialization vectors with the block in a mode-specific fashion.

For example, counter mode uses a counter as its initialization vector, which it increments, encrypts, and then exclusive-ORs the result with the block:

$$counter \rightarrow counter + 1; E \leftarrow Encrypt_{Key}(counter);$$
$$CipherTextBlock \leftarrow E \oplus PlainTextBlock$$

where $\oplus$ denotes exclusive OR. The algorithm output the new (unencrypted) counter value, which is used to encrypt the next block, and CipherTextBlock.

The process of assembling a message from a message encrypted under a mode of operation is very simple: Prepend the original initialization vector to the sequence of ciphertext blocks, which together replace the plaintext payload for the message. The right way to think of this is that the initialization vector becomes a new message header layer. Also prepended is a *key identifier*, which indicates to the receiver which key it should utilize to decrypt the payload. This is important because in many cases it is useful to employ multiple connections between the same pair of endpoints, and so the receiver can have multiple decryption keys to choose from for each message received from a particular source.

A receiver reverses this process: First it extracts the initialization vector from the data payload, then it uses this and the ciphertext blocks to recover the original plaintext message by reversing the steps in the mode of operation.

This paradigm is widely used in MAC and network layer security protocols, including 802.11i, 802.16e, 802.1ae, and IPsec, each of which utilizes AES in modes related to counter and cipher-block chaining modes.

A stream cipher treats the data as a continuous stream and can be thought of as encrypting and decrypting data one bit at a time. Stream ciphers are usually designed so that each encrypted bit depends on all previously encrypted ones, so decryption becomes possible only if all the bits arrive in order; most true stream ciphers lack the random access property. This means that in principle stream ciphers only work in network protocols when they're used on top of a reliable data delivery service such as TCP, and so they only work correctly below the transport layer when used in conjunction with reliable data links. Stream ciphers are attractive from an implementation perspective because they can often achieve much higher throughputs than block ciphers. RC4 is an example of a popular stream cipher.

Stream ciphers typically do not use a mode of operation or an initialization vector at all, or at least not in the same sense as a block cipher. Instead, they are built as pseudo-random number generators, the output of which is based on a key. The random number generator is used to create a sequence of bits that appear random, called a *key stream*, and the result is exclusive OR'd with the plaintext data to create ciphertext. Since XOR is an idempotent operation, decryption with a stream cipher is just the same operation: Generate the same key stream and exclusive OR it with the ciphertext to recover the plaintext. Since stream ciphers do not utilize initialization vectors, Internet protocols employing stream ciphers do not need the extra overhead of a header to convey the initialization vector needed by the decryptor in the block cipher case. Instead, these protocols rely on the sender and receiver being able to keep their respective key stream generators synchronized for each bit transferred. This implies that stream ciphers can only be used over a reliable medium such as TCP—that is, a transport that guarantees delivery of all bits in the proper order and without duplication.

Transport layer security is an example of an Internet security protocol that uses the stream cipher RC4. TLS runs on top of TCP.

Assuming that a symmetric encryption scheme is well designed, its efficacy against eavesdropping depends on four factors. Failing to consider *any* of these can cause the encryption scheme to fail catastrophically.

Independence of Keys

This is perhaps the most important consideration for the use of encryption. All symmetric encryption schemes assume that the encryption key for each and every session is generated independently of the encryption keys used for every other session. Let's parse this thought.

- *Independent* means selected or generated by a process that is indistinguishable by *any* polynomial time statistical test from the uniform distribution applied to the key space. One common failure is to utilize a key generation algorithm that is not random, such as using the MAC address or IP address of a device or time of session creation as the basis for a key. Schemes that use such public values instead of randomness for keys are easily broken using brute-force search techniques such as dictionary attacks. A second common failure is to pick an initial key randomly but create successive keys by some simple transformation, such as incrementing the initial key, exclusive OR'ing the MAC address of the device with the key, and so on. Encryptions using key generation schemes of this sort are easily broken using differential cryptanalysis and related key attacks.
- *Each and every* mean each and every. For a block cipher, reusing the same key twice with the same initialization vector can allow an adversary to recover the plaintext data from the ciphertext *without* using the key. Similarly, each key always causes the pseudo-random number generator at the heart of a stream cipher to generate the same key stream, and reuse of the same key stream again will leak the plaintext data from the ciphertext without using the key.
- Methods effective for the coordinated generation of random keys at the beginning of each session constitute a complicated topic. We address it in our discussion of session startup later in the chapter.

Limited Output

Perhaps the second most important consideration is to limit the amount of information encrypted under a single key. The modern definition of security for an encryption scheme revolves around the idea of indistinguishability of the scheme's output from random. This goes back to a notion of ideal security proposed by Shannon. This has a dramatic effect on how long an encryption key may be safely used before an adversary has sufficient information to begin to learn something about the encrypted data.

Every encryption scheme is ultimately a deterministic algorithm, and no deterministic algorithm can generate an infinite amount of output that is indistinguishable from random. This means that encryption keys must be replaced on a regular basis. The amount of data that can be safely encrypted under a single key depends very much on the encryption scheme. As usual, the limitations for block ciphers and stream ciphers are a bit different.

Let the block size for a block cipher be some integer $n > 0$. Then, for any key K, for every string S_1 there is another string S_2 so that:

$$Encrypt_K(S_2) = S_1 \text{ and } Decrypt_K(S_1) = S_2$$

This says that a block cipher's encrypt and decrypt operations are *permutations* of the set of all bit strings whose length equals the block size. In particular, this property says that every pair of distinct n bit strings results in distinct n bit ciphertexts for any block cipher. However, by an elementary theorem from probability called the birthday paradox, random selection of n bit strings should result in a 50% probability that some string is chosen at least twice after about $2^{n/2}$ selections. This has an important consequence for block ciphers. It says that an algorithm as simple as naïve guessing can distinguish the output of the block cipher from random after about $2^{n/2}$ blocks have been encrypted. This means that an encryption key should never be used to encrypt even close to $2^{n/2}$ blocks before a new, independent key is generated.

To make this specific, DES and 3DES have a block size of 64 bits; AES has a 128-bit block size. Therefore a DES or 3DES key should be used much less than to encrypt $2^{64/2} = 2^{32}$ blocks, whereas an AES key should never be used to encrypt as many as 2^{64} blocks; doing so begins to leak information about the encrypted data without use of the encryption key. As an example, 802.11i has been crafted to limit each key to encrypting 2^{48} before forcing generation of a new key.

This kind of arithmetic does not work for a stream cipher, since its block size is 1 bit. Instead, the length of time a key can be safely used is governed by the periodicity of the pseudorandom number generator at the heart of the stream cipher. RC4, for instance, becomes distinguishable from random after generating about 2^{31} bytes. Note that $31 \approx 32 = \sqrt{256}$, and 256 bytes is the size of the RC4 internal state. This illustrates the rule of thumb that there is a birthday paradox relation between the maximum number of encrypted bits of a stream cipher key and its internal state.

Key Size

The one "fact" about encryption that everyone knows is that larger keys result in stronger encryption. This is indeed true, provided that the generate keys operation is designed according to the independence condition. One common mistake is to properly generate a short key—say, 32 bits long—that is then concatenated to get a key of the length needed by the selected encryption scheme—say, 128 bits. Another similar error is to generate a short key and manufacture the remainder of the key with known public data, such as an IP address. These methods result in a key that is only as strong as the short key that was generated randomly.

Mode of Operation

The final parameter is the mode of operation—that is, the rules for using a block cipher to encrypt messages whose length is different than the block cipher width. The most common problem is failure to respect the document terms and conditions defined for using the mode of operation.

As an illustration of what can go wrong—even by people who know what they are doing—cipher-block chaining mode requires that the initialization vector be chosen randomly. The earliest version of the IPsec standard used cipher-block chaining mode exclusively for encryption. This standard recommended choosing initialization vectors as the final block of any prior message sent. The reasoning behind this recommendation was that, because an encrypted block cannot be distinguished from random if the number of blocks encrypted is limited, a block of a previously encrypted message ought to suffice. However, the advice given by the standard was erroneous because the initialization vector selection algorithm failed to have one property that a real random selection property has: The initialization vector is not unpredictable. A better way to meet the randomness requirement is to increment a counter, prepend it to the message to encrypt, and then encrypt the counter value, which becomes the initialization vector. This preserves the unpredictability property at a cost of encrypting one extra block.

A second common mistake is to design protocols using a mode of operation that was not designed to encrypt multiple blocks. For example, failing to use a mode of operation at all—using the naked encrypt and decrypt operations, with no initialization vector—is itself a mode of operation called *electronic code book* mode. Electronic code book mode was designed to encrypt messages that never span more than a single block—for example, encrypting keys to distribute for other operations. Using electronic code book mode on a message longer than a single block leaks a bit per block, however, because this mode allows an attacker to disguise when two plaintext blocks are the same or different. A classical example of this problem is to encrypt a photograph using electronic code book mode. The main outline of the photograph shows through plainly. This is not a failure of the encryption scheme; it is rather using encryption in a way that was never intended.

Now that we understand how encryption works and how it is used in Internet protocols, we should ask why is it needed at different layers. What does encryption at each layer of the Internet architecture accomplish? The best way to answer this question is to watch what it does.

Encryption applied at the MAC layer encrypts a single link. Data is encrypted prior to being put on a link and is decrypted again at the other end of a link. This leaves the IP datagrams conveyed by the MAC layer frames exposed inside each router as they wend their way across the Internet. Encryption at the MAC layer is a good way to transparently prevent data from leaking, since many devices never use encryption. For example, many organizations are distributed geographically and use direct point-to-point links to connect sites; encrypting the links connecting sites prevents an outsider from learning the organization's confidential information merely by eavesdropping. Legal wiretaps also depend on this arrangement because they monitor data inside routers. The case of legal wiretaps also illustrates the problem with link layer encryption only: If an unauthorized party assumes control of a router, they are free to read all the datagrams that traverse the router.

IPsec operates essentially at the network layer. Applying encryption via IPsec prevents exposure of the datagrams' payload end to end, so the data is still protected within routers. Since the payload of a datagram includes both the transport layer header as well as its data segments, applying encryption at the IPsec layer hides the applications being used as well as the data. This provides a big boost in confidentiality but also leads to more inefficient use of the Internet, since traffic-shaping algorithms in routers critically depend on having complete access to the transport headers. Using encryption at the IPsec layer also means the endpoints do not have to know whether each link a datagram traverses through the Internet applies encryption; using encryption at this layer simplifies the security analysis over encryption applied at the MAC layer alone. Finally, like MAC layer encryption, IPsec is a convenient tool for introducing encryption transparently to protect legacy applications, which by and large ignored confidentiality issues.

The transport layer encryption function can be illustrated by TLS. Like IPsec, TLS operates end to end, but TLS encrypts only the application data carried in the transport data segments, leaving the transport header exposed. Thus, with TLS, routers can still perform their traffic-shaping function, and we still have the simplified security analysis that comes with end-to-end encryption. The first downside of this method is that the exposure of the transport headers gives the attacker greater knowledge about what might be encrypted in the payload. The second downside is that it is somewhat more awkward to introduce encryption transparently at the transport layer; encryption at the transport layer requires cooperation by the application to perform properly.

This analysis says that it is reasonable to employ encryption at any one of the network protocol layers, because each solves a slightly different problem.

Before leaving the topic of encryption, it is worthwhile to emphasize what encryption does and does not do. Encryption, when properly used, is a *read access control*. If used properly, no one who lacks access to the encryption key can read the encrypted data. Encryption, however, is *not* a *write access control*; that is, it does not maintain the integrity of the encrypted data. Counter mode and stream ciphers are subject to bit-flipping attacks, for instance. An attacker launches a bit-flipping attack by capturing a frame or datagram, changing one or more bits from 0 to 1 (or vice versa) and retransmitting the altered frame. The resulting frame decrypts to some result—the altered message decrypts to something—and if bits are flipped judiciously, the result can be intelligible. As a second example, cipher-block chaining mode is susceptible to cut-and-paste attacks, whereby the attack cuts the final few blocks from one message in a stream and uses them to overwrite the final blocks of a later stream. At most one block decrypts to gibberish; if the attacker chooses the paste point judiciously, for example, so that it falls where the application ought to have random data anyway, this can be a powerful attack. The upshot is that even encrypted data needs an integrity mechanism to be effective, which leads us to the subject of defenses against forgeries.

Defending Against Forgeries and Replays

Forgery and replay detection are usually treated together because replays are a special kind of forgery. We follow this tradition in our own discussion. Forgery detection, not eavesdropping protection, is the central concern for designs to secure network protocol. This is because every accepted forgery of an encrypted frame or datagram is a question for which the answer can tell the adversary about the encryption key or plaintext data. Just as in school, an attacker can learn about the encrypted stream or encryption key faster by asking questions rather than sitting back and passively listening.

Since eavesdropping is a passive attack, whereas creating forgeries is active, turning from the subject of eavesdropping to that of forgeries changes the security goals subtly. Encryption has a security goal of prevention—to prevent the adversary from learning anything useful about the data that cannot be derived in other ways. The comparable security goal for forgeries is to prevent the adversary from creating forgeries, which is infeasible. This is because any device with a transmitter appropriate for the medium can send forgeries by creating frames and datagrams using addresses employed by other parties. What is feasible is a form of asking forgiveness instead of permission: Prevent the adversary from creating *undetected* forgeries.

The cryptographic tool underlying forgery detection is called a *message authentication code*. Like an encryption scheme, a message authentication code consists of three operations: a *key generation* operation, a *tagging* operation, and a *verification* operation. Also like encryption, the key generation operation, which generates a symmetric key shared between the sender and receiver, is usually application specific. The tagging and verification operations, however, are much different from encrypt and decrypt.

The tagging operation takes the symmetric key, called an *authentication key*, and a message as input parameters and outputs a *tag*, which is a cryptographic checksum depending on the key and message as its output.

The verification operation takes three input parameters: the symmetric key, the message, and its tag. The verification algorithm recomputes the tag from the key and message and compares the result against the tag input into the algorithm. If the two fail to match, the verify algorithm outputs a signal that the message is a forgery. If the input and locally computed tag match, the verify algorithm declares that the message is authenticated.

The conclusion drawn by the verify algorithm of a message authentication code is not entirely logically correct. Indeed, if the tag is n bits in length, an attacker could generate a random n bit string as its tag and it would have one chance in 2^n of being valid. A message authentication scheme is considered good if there are no polynomial time algorithms that are significantly better than random guessing at producing correct tags.

Message authentication codes are incorporated into network protocols in a manner similar to encryption. First, a sequence number is prepended to the data that is being forgery protected; the sequence number, we will see, is used to detect replays. Next, a message authentication code tagging operation is applied to the sequence number and message body to produce a tag. The tag is appended to the message, and a key identifier for the authentication key is prepended to the message. The message can then be sent. The receiver determines whether the message was a forgery by first finding the authentication key identified by the key identifier, then by checking the correctness of the tag using the message authentication code's verify operation. If these checks succeed, the receiver finally uses the sequence number to verify that the message is not a replay.

How does replay detection work? When the authentication key is established, the sender initializes to zero the counter that is used in the authenticated message. The receiver meanwhile establishes a replay window, which is a list of all recently received sequence numbers. The replay window is initially empty. To send a replay protected frame, the sender increments his counter by one and prepends this at the front of the data to be authenticated prior to tagging. The receiver extracts the counter value from the received message and compares this to the replay window. If the counter falls before the replay window, which means it is too old to be considered valid, the receiver flags the message as a replay. The receiver does the same thing if the counter is already represented in the replay window data structure. If the counter is greater than the bottom of the replay window and is a counter value that has not yet been received, the frame or datagram is considered "fresh" instead of a replay.

The process is simplest to illustrate for the MAC layer. Over a single MAC link it is ordinarily impossible for frames to be reordered, because a single device can access the medium at a time and, because of the speed of electrons or photons comprising the signals representing bits, at least some of the bits at the start of a frame are received prior to the final bits being transmitted (satellite links are an exception). If frames cannot be reordered by a correctly operating MAC layer, the replay window data structure records the counter for the last received frame, and the replay detection algorithm merely has to decide whether the replay counter value in a received frame is larger than that recorded in its replay window. If the counter is less than or equal to the replay window value, the frame is a forgery; otherwise it is considered genuine. 802.11i, 802.16, and 802.1ae all employ this approach to replay detection. This same approach can be used by a message authentication scheme operating above the transport layer, by protocols such as TLS and SSH (Secure Shell), since the transport eliminates duplicates and delivers bits in the order sent. The replay window is more complicated at the network layer, however, because some reordering is natural, given that the network reorders datagrams. Hence, for the network layer the replay window is usually sized to account for the maximum reordering expected in the "normal" Internet. IPsec uses this more complex replay window.

The reason that this works is the following: Every message is given a unique, incrementing sequence number in the form of its counter value. The transmitter computes the message authentication code tag over the sequence number and the message data. Since it is infeasible for a computationally bounded adversary to create a valid tag for the data with probability significantly greater than $1/2^n$, a tag validated by the receiver implies that the message, including its sequence number, was created by the transmitter. The worst thing that could have happened, therefore, is that the adversary has delayed the message. However, if the sequence number falls within the replay window, the message could not have been delayed longer than reordering due to the normal operation of forwarding and traffic shaping within the Internet.

A replay detection scheme limits an adversary's opportunities to delete and to reorder messages. If a message does not arrive at its destination, its sequence number is never set in the receive window, so it can be declared a lost message. It is easy to track the percentage of lost messages, and if this exceeds some threshold, then communications become unreliable, but more important, the cause of the unreliability can be investigated. Similarly, messages received outside the replay window can also be tracked, and if the percentage becomes too high, messages are arriving out of order more frequently than might be expected from normal operation of the Internet, pointing to a configuration problem, an equipment failure, or an attack. Again, the cause of the anomaly can be investigated. Mechanisms like these are often the way that attacks are discovered in the first place. The important lesson is that attacks and even faulty equipment or misconfigurations are often difficult to detect without collecting reliability statistics, and the forgery detection mechanisms can provide some of the best reliability statistics available.

Just like encryption, the correctness of this analysis depends critically on the design enforcing some fundamental assumptions, regardless of the quality of the message authentication code on which it might be based. If any of the following assumptions are violated, the forgery detection scheme can fail catastrophically to accomplish its mission.

Independence of Authentication Keys

This is absolutely paramount for forgery detection. If the message authentication keys are not independent, an attacker can easily create forged message authentication tags based on authentication keys learned in other ways. This assumption is so important that it is useful to examine in greater detail.

The first point is that a message authentication key utterly fails to accomplish its mission if it is shared among even three parties; only two parties must know any particular authentication key. This is very easy to illustrate. Suppose A, B, and C were to share a message authentication key, and suppose A creates a forgery-protected message it sends to C. What can C conclude when it receives this message? C cannot conclude that the message actually originated from A, even though its addressing indicates it did, because B could have produced the same message and used A's address. C cannot even conclude that B did not change some of the message in transit. Therefore, the algorithm loses all its efficacy for detecting forgeries if message authentication keys are known by more than two parties. They must be known by at least two parties or the receiver cannot verify that the message and its bits originated with the sender.

This is much different than encryption. An encryption/decryption key can be distributed to every member of a group, and as long as the key is not leaked from the group to a third party, the encryption scheme remains an effective read access control against parties that are not members of the group. Message authentication utterly fails if the key is shared beyond two parties. This is due to the active nature of forgery attacks and the fact that forgery handling, being a detection rather than a prevention scheme, already affords the adversary more latitude than encryption toward fooling the good guys.

So message authentication keys must be shared between exactly two communicating devices for forgery detection schemes to be effective. As with encryption keys, a message authentication key must be generated randomly because brute-force searches and related key attacks can recover the key by observing messages transiting the medium.

No Reuse of Replay Counter Values with a Key

Reusing a counter with a message authentication key is analogous to reusing an initialization vector with an encryption key. Instead of leaking data, however, replay counter value reuse leads automatically to trivial forgeries based on replayed messages. The attacker's algorithm is trivial: Using a packet sniffer, record each of the messages protected by the same key and file them in a database. If the attacker ever receives a key identifier and sequence number pair already in the database, the transmitter has begun to reuse replay counter values with a key. The attacker can then replay any message with a higher sequence number and the same key identifier. The receiver will be fooled into accepting the replayed message.

An implication of this approach is that known forgery detection schemes cannot be based on static keys. We could to the contrary attempt to design such a scheme. One could try to checkpoint in nonvolatile memory the replay counter at the transmitter and the replay window at the receiver. This approach does not work, however, in the presence of a Dolev–Yao adversary. The adversary can capture a forgery-protected frame in flight and then delete all successive messages. At its convenience later, the adversary resends the captured message. The receiver, using its static message authentication key, will verify the tag and, based on its replay window retrieved from nonvolatile storage, verify that the message is indeed in sequence and so accept the message as valid. This experiment demonstrates that forgery detection is not entirely satisfactory, because sequence numbers do not take timeliness into account. Secure clock synchronization, however, is a difficult problem with solutions that enjoy only partial success. The construction of better schemes that account for timing remains an open research problem.

Key Size

If message authentication keys must be randomly generated, they must also be of sufficient size to discourage brute-force attack. The key space has to be large enough to make exhaustive search for the message authentication key cost prohibitive. Key sizes for message authentication comparable with those for encryption are sufficient for this task.

Message Authentication Code Tag Size

We have seen many aspects that make message authentication codes somewhat more fragile encryption schemes. Message authentication code size is one in which forgery detection can on the contrary effectively utilize a smaller block size than an encryption scheme. Whereas an encryption scheme based on a 128-bit block size has to replace keys every 2^{48} or so blocks to avoid leaking data, an encryption scheme can maintain the same level of security with about a 48-bit message authentication code tag. The difference is that the block cipher-based encryption scheme leaks information about the encrypted data due to the birthday paradox, whereas an attacker has to create a valid forgery based on exhaustive search due to the active nature of a forgery attack. In general, to determine the size of a tag needed by a message authentication code, we have only to determine the maximum number of messages sent in the lifetime of the key. If this number of messages is bounded by 2^n, the tag need only be $n + 1$ bits long.

As with encryption, many find it confusing that forgery detection schemes are offered at nearly every layer of the Internet architecture. To understand this, it is again useful to ask the question about what message forgery detection accomplishes at each layer.

If a MAC module requires forgery detection for every frame received, physical access to the medium being used by the module's PHY layer affords an attacker no opportunity to create forgeries. This is a very strong property. It means that the only MAC layer messages attacking the receiver are either generated by other devices authorized to attach to the medium or else are forwarded by the network layer modules of authorized devices, because all frames received directly off the medium generated by unauthorized devices will be discarded by the forgery detection scheme. A MAC layer forgery detection scheme therefore essentially provides a write access control of the physical medium, closing it to unauthorized parties. Installing a forgery detection scheme at any other layer will not provide this kind of protection. Requiring forgery detection at the MAC layer is therefore desirable whenever feasible.

A different kind of assurance is provided by forgery detection at the network layer. IPsec is the protocol designed to accomplish this function. If a network layer module requires IPsec for every datagram received, this essentially cuts off attacks against the device hosting the module to other authorized machines in the entire Internet; datagrams generated by unauthorized devices will be dropped. With this forgery detection scheme it is still possible for an attacker on the same medium to generate frames attacking the device's MAC layer module, but attacks against higher layers become computationally infeasible. Installing a forgery detection scheme at any other layer will not provide this kind of protection. Requiring forgery detection at the network layer is therefore desirable whenever feasible as well.

Applying forgery detection at the transport layer offers different assurances entirely. Forgery detection at this level assures the receiving application that the arriving messages were generated by the peer application, not by some virus or Trojan-horse program that has linked itself between modules between protocol layers on the same or different machine. This kind of assurance cannot be provided by any other layer. Such a scheme at the network or MAC layers only defends against message injection by unauthorized devices on the Internet generally or directly attached to the medium, not against messages generated by unauthorized processes running on an authorized machine. Requiring forgery detection at the transport layer therefore is desirable whenever it is feasible.

The conclusion is that forgery detection schemes accomplish different desirable functions at each protocol layer. The security goals that are achievable are always architecturally dependent, and this sings through clearly with forgery detection schemes.

We began the discussion of forgery detection by noting that encryption by itself is subject to attack. One final issue is how to use encryption and forgery protection together to protect the same message. Three solutions could be formulated to this problem. One approach might be to add forgery detection to a message first—add the authentication key identifier, the replay sequence number, and the message authentication code tag—followed by encryption of the message data and forgery detection headers. TLS is an example Internet protocol that takes this approach. The second approach is to reverse the order of encryption and forgery detection: First encrypt and then compute the tag over the encrypted data and the encryption headers. IPsec is an example Internet protocol defined to use this approach. The last approach is to apply both simultaneously to the plaintext data. SSH is an Internet protocol constructed in this manner.

Session Startup Defenses

If encryption and forgery detection techniques are such powerful security mechanisms, why aren't they used universally for all network communications? The problem is that not everyone is your friend; everyone has enemies, and in every human endeavor there are those with criminal mindsets who want to prey on others. Most people do not go out of their way to articulate and maintain relationships with their enemies unless there is some compelling reason to do so, and technology is powerless to change this.

More than anything else, the keys used by encryption and forgery detection are relationship signifiers. Possession of keys is useful not only because they enable encryption and forgery detection but because their use assures the remote party that messages you receive will remain confidential and that messages the peer receives from you actually originated from you. They enable the accountable maintenance of a preexisting relationship. If you receive a message that is protected by a key that only you and I know, and you didn't generate the message yourself, it is reasonable for you to conclude that I sent the message to you and did so intentionally.

If keys are signifiers of preexisting relationships, much of our networked communications cannot be defended by cryptography, because we do not have preexisting relationships with everyone. We send and receive email to and from people we have never met. We buy products online from merchants we have never met. None of these relationships would be possible if we required all messages to be encrypted or authenticated. What is always required is an open, unauthenticated, risky channel to establish new relationships; cryptography can only assure us that communication from parties with whom we already have relationships is indeed occurring with the person with whom we think we are communicating.

A salient and central assumption for both encryption and forgery detection is that the keys these mechanisms use are fresh and independent across sessions. A session is an instance of exercising a relationship to effect communication. This means that secure communications require a state change, transitioning from a state in which two communicating parties are not engaged in an instance of communication to one in which they are. This state change is *session establishment*.

Session establishment is like a greeting between human beings. It is designed to synchronize two entities communicating over the Internet and establish and synchronize their keys, key identifiers, sequence numbers and replay windows, and, indeed, all the states to provide mutual assurance that the communication is genuine and confidential.

The techniques and data structures used to establish a secure session are different from those used to carry on a conversation. Our next goal is to look at some representative mechanisms in this area. The field is vast and it is impossible to do more than skim the surface briefly to give the reader a glimpse of the beauty and richness of the subject.

Secure session establishment techniques typically have three goals, as described in the following subsections.

Mutual Authentication

First, session establishment techniques seek to mutually authenticate the communicating parties to each other. *Mutually authenticate* means that both parties learn the "identity" of the other. It is not possible to know what is proper to discuss with another party without also knowing the identity of the other party. If only one party learns the identity of the other, it is always possible for an imposter to masquerade as the unknown party.

Key Secrecy

Second, session establishment techniques seek to establish a session key that can be maintained as a secret between the two parties and is known to no one else. The session key must be independent from all other keys for all other session instances and indeed from all other keys. This implies that no adversary with limited computational resources can distinguish the key from random. Generating such an independent session key is both harder and easier than it sounds; it is always possible to do so if a preexisting relationship already exists between the two communicating parties, and it is impossible to do so reliably if a preexisting relationship does not exist. Relationships begat other relationships, and nonrelationships are sterile with respect to the technology.

Session State Consistency

Finally, the parties need to establish a consistent view of the session state. This means that they both agree on the identities of both parties; they agree on the session key instance; they agree on the encryption and forgery detection schemes used, along with any associated state such as sequence counters and replay windows; and they agree on which instance of communication this session represents. If they fail to agree on a single shared parameter, it is always possible for an imposter to convince one of the parties that it is engaged in a conversation that is different from its peer's conversation.

Mutual Authentication

There are an enormous number of ways to accomplish the mutual authentication function needed to initiate a new session. Here we examine two that are used in various protocols within the Internet.

A Symmetric Key Mutual Authentication Method

Our old friend the message authentication code can be used with a static, long-lived key to create a simple and robust mutual authentication scheme. Earlier we stressed that the properties of message authentication are incompatible with the use of a static key to provide forgery detection of session-oriented messages. The incompatibility is due to the use of sequence numbers for replay detection. We will replace sequence numbers with unpredictable quantities in order to resocialize static keys. The cost of this resocialization effort will be a requirement to exchange extra messages.

Suppose parties A and B want to mutually authenticate. We will assume that ID_A is B's name for A, whereas ID_B is A's name for B. We will also assume that A and B share a long-lived message authentication key K, and that K is known only to A and B. We will assume that A initiates the authentication. A and B can mutually authenticate using a three-message exchange, as follows: For message 1, A generates a random number R_A and sends a message containing its identity ID_A and random number to B:

$$A \rightarrow B\text{: } ID_A, R_A \tag{1}$$

The notation A → B: *m* means that A sends message *m* to B. Here the message being passed is specified as ID_A, R_A, meaning it conveys A's identity ID_A and A's random number R_A. This message asserts B's name for A, to tell B which is the right long-lived key it should use in this instance of the authentication protocol. The random number R_A plays the role of the sequence number in the session-oriented case.

If B is willing to have a conversation with A at this time, it fetches the correct message authentication key K, generates its own random number R_B, and computes a message authentication code tag T over the message ID_B, ID_A, R_A, R_B, that is, over the message consisting of both names and both random numbers. B appends the tag to the message, which it then sends to A in response to message 1:

$$B \rightarrow A\text{: } ID_B, ID_A, R_A, R_B, T \tag{2}$$

B includes A's name in the message to tell A which key to use to authenticate the message. It includes A's random number R_A in the message to signal the protocol instance to which this message responds.

The magic begins when A validates the message authentication code tag T. Since independently generated random numbers are unpredictable, A knows that the second message could not have been produced before A sent the first, because it returns R_A to A. Since the authentication code tag T was computed over the two identities ID_B and ID_A and the two random numbers R_A and R_B using the key K known only to A and B, and since A did not create the second message itself, A knows that B must have created message 2. Hence, message 2 is a response from B to A's message 1 for this instance of the protocol. If the message were to contain some other random number than R_A, A would know the message is not a response to its message 1.

If A verifies message 2, it responds by computing a message authentication code tag T' computed over ID_A and B's random number RB, which it includes in message 3:

$$A \rightarrow B\text{: } ID_A, R_B, T' \tag{3}$$

Reasoning as before, B knows A produced message 3 in response to its message 2, because message 3 could not have been produced prior to message 2 and only A could have produced the correct tag T'. Thus, after message 3 is delivered, A and B both have been assured of each other's identity, and they also agree on the session instance, which is identified by the pair of random numbers R_A and R_B.

A deeper analysis of the protocol reveals that message 2 must convey both identities and both random numbers protected from forgery by the tag T. This construction binds A's view of the session with B's. This binding prevents interleaving or man-in-the-middle attacks. As an example, without this binding, a third party, C, could masquerade as B to A and as A to B.

It is worth noting that message 1 is not protected from either forgery or replay. This lack of any protections is an intrinsic part of the problem statement. During the protocol, A and B must transition from a state where they are unsure about the other's identity and have no communication instance instantiating the long-term relationship signified by the encryption key K to a state where they fully agree on each other's identities and a common instance of communication expressing their long-lived relationship. A makes the transition upon verifying message 2, and there are no known ways to reassure it about B until this point of the protocol. B makes the state transition once it has completed verification of message 3. The point of the protocol is to transition from a mutually suspicious state to a mutually trusted state.

An Asymmetric Key Mutual Authentication Method

Authentication based on asymmetric keys is also possible. In addition to asymmetric encryption there is also an asymmetric key analog of a message authentication code called a *signature scheme*. Just like a message authentication code, a signature scheme consists of three operations: *key generate*, *sign*, and *verify*. The key generate operation outputs two parameters, a signing key S and a related verification key V. S's key holder is never supposed to reveal S to another party, whereas V is meant to be a public value. Under these assumptions the sign operation takes the signing key S and a message M as input parameters and output a signature s of M. The verify operation takes the verification key V, message M and signature s as inputs, and returns whether it verifies that s was created from S and M. If the signing key S is indeed known by only one party, the signature s must have been produced by that party. This is because it is infeasible for a computationally limited party to compute the signature *s* without *S*. Asymmetric signature schemes are often called *public/ private key schemes* because *S* is maintained as a secret, never shared with another party, whereas the verification key is published to everyone.

Signature schemes were invented to facilitate authentication. To accomplish this goal, the verification key must be public, and it is usually published in a certificate, which we will denote as $cert(ID_A, V)$, where ID_A is the identity of the key holder of *S*, and *V* is the verification key corresponding to A. The certificate is issued by a well-known party called a *certificate authority*. The sole job of the certificate authority is to introduce one party to another. A certificate $cert(ID_A, V)$ issued by a certificate authority is an assertion that entity A has a public verification key *V* that is used to prove A's identity.

As with symmetric authentication, hundreds of different authentication protocols can be based on signature schemes. The following is one example among legion:

$$A \rightarrow B\colon cert(ID_A, V), R_A \tag{4}$$

Here $cert(ID_A, V)$ is A's certificate, conveying its identity ID_A and verification key V; R_A is a random number generated by A. If B is willing to begin a new session with A, it responds with the message:

$$B \rightarrow A\colon cert(ID_B, V'), R_B, R_A, sig_B(ID_A, R_B, R_A) \tag{5}$$

R_B is a random number generated by B, and $sig_B(ID_A, R_B, R_A)$ is B's signature over the message with fields ID_A, R_B, and R_A. Including IDA under B's signature is essential because it is B's way of asserting that A is the target of message 2. Including *RB* and *RA* in the information signed is also necessary to defeat man-in-the-middle attacks. A responds with a third message:

$$A \rightarrow B\colon cert(ID_A, V), R_b, sig_B(ID_B, R_B) \tag{6}$$

A Caveat

Mutual authentication is necessary to establish identities. Identities are needed to decide on the access control policies to apply to a particular conversation, that is, to answer the question, Which information that the party knows is suitable for sharing in the context of this communications instance? Authentication—mutual or otherwise—has very limited utility if the communications channel is not protected against eavesdropping and forgeries.

One of the most common mistakes made by Wi-Fi hotspot operators, for instance, is to require authentication but disable eavesdropping and forgery protection for the subsequent Internet access via the hotspot. This is because anyone with a Wi-Fi radio transmitter can access the medium and hijack the session from a paying customer. Another way of saying this is that authentication is useful only when it's used in conjunction with a secure channel. This leads to the topic of session key establishment. The most common use of mutual authentication is to establish ephemeral session keys using the long-lived authentication keys. We will discuss session key establishment next.

Key Establishment

Since it is generally infeasible for authentication to be meaningful without a subsequent secure channel, and since we know how to establish a secure channel across the Internet if we have a key, the next goal is to add key establishment to mutual authentication protocols. In this model, a mutual authentication protocol establishes an ephemeral session key as a side effect of its successful operation; this session key can then be used to construct all the encryption and authentication keys needed to establish a secure channel. All the session states, such as sequence number, replay windows, and key identifiers, can be initialized in conjunction with the completion of the mutual authentication protocol.

It is usually feasible to add key establishment to an authentication protocol. Let's illustrate this with the symmetric key authentication protocol, based on a message authentication code, discussed previously. To extend the protocol to establish a key, we suppose instead that A and B share two long-lived keys K and K'. The first key K is a message authentication key as before. The second key K' is a derivation key, the only function of which is to construct other keys within the context of the authentication protocol. This is accomplished as follows: After verifying message 2 (from line 2 previously), A computes a session key SK as:

$$SK \leftarrow prf(K', R_A, R_B, ID_A, ID_B, length) \tag{7}$$

Here *prf* is another cryptographic primitive called a *pseudo random function*. A pseudo random function is characterized by the properties that (a) its output is indistinguishable from random by any computationally limited adversary and (b) it is hard to invert, that is, given a fixed output O, it is infeasible for any computationally limited adversary to find an input I so that $O \leftarrow prf(I)$. The output SK of (7) is *length* bits long and can be split into two pieces to become encryption and message authentication keys. B generates the same SK when it receives message 3. An example of a pseudo-random function is any block cipher, such as AES, in cipher-block chaining MAC mode. Cipher-block chaining MAC mode is just like Cipher-block chaining mode, except all but the last block of encrypted data is discarded.

This construction meets the goal of creating an independent, ephemeral set of encryptions of message authentication keys for each session. The construction creates independent keys because any two outputs of a *prf* appear to be independently selected at random to any adversary that is computationally limited. A knows that all the outputs are statistically distinct, because A picks the parameter to the prf R_A randomly for each instance of the protocol; similarly for B. And using the communications instances identifiers *RA, RB* along with A and B's identities ID_A and ID_B are interpreted as a "contract" to use *SK* only for this session instance and only between A and B.

Public key versions of key establishment based on signatures and asymmetric encryption also exist, but we will close with one last public key variant based on a completely different asymmetric key principle called the *Diffie–Hellman algorithm.*

The Diffie–Hellman algorithm is based on the discrete logarithm problem in finite groups. A group G is a mathematical object that is closed under an associative multiplication and has inverses for each element in G. The prototypical example of a finite group is the integers under addition modulo a prime number p.

The idea is to begin with an element g of a finite group G that has a long period. This means to $g^1 = g$, $g^2 = g \times g$, $g^3 = g^2 \times g$, ... Since G is finite, this sequence must eventually repeat. It turns out that $g = g^{n+1}$ for some integer $n > 1$, and $g^n = e$ is the group's neutral element. The element e has the property that $h \times e = e \times h = h$ for every element h in G, and n is called the *period* of g. With such an element it is easy to compute powers of g, but it is hard to compute the logarithm of g^k. If g is chosen carefully, no polynomial time algorithm is known that can compute k from g^k. This property leads to a very elegant key agreement scheme:

$$\begin{aligned} A \rightarrow B&: cert(ID_A, V), g^a \\ B \rightarrow A&: g^b,\ cert(ID_B, V'), sig_B(g^a, g^b,\ ID_A) \\ A \rightarrow B&: sig_A(g^b, g^a, ID_B) \end{aligned}$$

The session key is then computed as $SK \leftarrow prf(K, g^a\ g^b, ID_A, ID_B)$, where $K \leftarrow prf(0, g^{ab})$. In this protocol, a is a random number chosen by A, b is a random number chosen by B, and 0 denotes the all zeros key. Note that A sends g^a unprotected across the channel to B.

The quantity g^{ab} is called the Diffie–Hellman key. Since B knows the random secret b, it can compute $g^{ab} = (g^a)^b$ from A's public value g^a, and similarly A can compute g^{ab} from B's public value g^b. This construction poses no risk, because the discrete logarithm problem is intractable, so it is computationally infeasible for an attacker to determine a from g^a. Similarly, B may send g^b across the channel in the clear, because a third party cannot extract b from g^b. B's signature on message 2 prevents forgeries and assures that the response is from B. Since no method is known to compute g^{ab} from g^a and g^b, only A and B will know the Diffie–Hellman key at the end of the protocol. The step $K \leftarrow prf(0, g^{ab})$ extracts all the computational entropy from the Diffie–Hellman key. The construction $SK \leftarrow prf(K, g^a\ g^b, ID_A, ID_B)$ computes a session key, which can be split into encryption and message authentication keys as before.

The major drawback of Diffie–Hellman is that it is subject to man-in-the-middle attacks. The preceding protocol uses signatures to remove this threat. B's signature authenticates B to *a* and also binds g^a and g^b together, preventing man-in-the-middle attacks. Similarly, A's signature on message 3 assures B that the session is with A.

These examples illustrate that is practical to construct session keys that meet the requirements for cryptography, if a preexisting long-lived relationship already exists.

State Consistency

We have already observed that the protocol specified in Equations (1) through (3) achieves state consistency when the protocol succeeds. Both parties agree on the identities and on the session instance. When a session key *SK* is derived, as in Equation (7), both parties also agree on the key. Determining which parties know which pieces of information after each protocol message is the essential tool for a security analysis of this kind of protocol.
The analysis of this protocol is typical for authentication and key establishment protocols.

4. Conclusion

This chapter examined how cryptography is used on the Internet to secure protocols. It reviewed the architecture of the Internet protocol suite, as even what security means is a function of the underlying system architecture. Next it reviewed the Dolev–Yao model, which describes the threats to which network communications are exposed. In particular, all levels of network protocols are completely exposed to eavesdropping and manipulation by an attacker, so using cryptography properly is a first-class requirement to derive any benefit from its use. We learned that effective security mechanisms to protect session-oriented and session establishment protocols are different, although they can share many cryptographic primitives. Cryptography can be very successful at protecting messages on the Internet, but doing so requires preexisting, long-lived relationships. How to build secure open communities is still an open problem; it is probably intractable because a solution would imply the elimination of conflict between human beings who do not know each other.

CHAPTER 8

The Botnet Problem

Daniel Ramsbrock,[*] Xinyuan Wang,[*] Xuxian Jiang[†]
[*]George Mason University, [†]North Carolina State University

A *botnet* is a collection of compromised Internet computers being controlled remotely by attackers for malicious and illegal purposes. The term comes from these programs being called *robots*, or *bots* for short, due to their automated behavior.

Bot software is highly evolved Internet malware, incorporating components of viruses, worms, spyware, and other malicious software. The person controlling a botnet is known as the *botmaster* or *bot-herder*, and he seeks to preserve his anonymity at all costs. Unlike previous malware such as viruses and worms, the motivation for operating a botnet is financial. Botnets are extremely profitable, earning their operators hundreds of dollars per day. Botmasters can either rent botnet processing time to others or make direct profits by sending spam, distributing spyware to aid in identity theft, and even extorting money from companies via the threat of a distributed denial-of-service (DDoS) attack [1]. It is no surprise that many network security researchers believe that botnets are one of the most pressing security threats on the Internet today.

Bots are at the center of the undernet economy. Almost every major crime problem on the Net can be traced to them.

—Jeremy Linden, formerly of Arbor Networks [2]

1. Introduction

You sit down at your computer in the morning, still squinting from sleep. Your computer seems a little slower than usual, but you don't think much of it. After checking the news, you try to sign into eBay to check on your auctions. Oddly enough, your password doesn't seem to work. You try a few more times, thinking maybe you changed it recently—but without success.

Figuring you'll look into it later, you sign into online banking to pay some of those bills that have been piling up. Luckily, your favorite password still works there—so it must be a temporary problem with eBay. Unfortunately, you are in for more bad news: The $0.00 balance on your checking and savings accounts isn't just a "temporary problem." Frantically clicking through the pages, you see that your accounts have been completely cleaned out with wire transfers to several foreign countries.

You check your email, hoping to find some explanation of what is happening. Instead of answers, you have dozens of messages from "network operations centers" around the world, informing you in no uncertain terms that your computer has been scanning, spamming, and sending out massive amounts of traffic over the past 12 hours or so. Shortly afterward, your Internet connection stops working altogether, and you receive a phone call from your service provider. They are very sorry, they explain, but due to something called "botnet activity" on your computer, they have temporarily disabled your account. Near panic now, you demand an explanation from the network technician on the other end. "What exactly is a botnet? How could it cause so much damage overnight?"

Though this scenario might sound far-fetched, it is entirely possible; similar things have happened to thousands of people over the last few years. Once a single bot program is installed on a victim computer, the possibilities are nearly endless. For example, the attacker can get your online passwords, drain your bank accounts, and use your computer as a remote-controlled "zombie" to scan for other victims, send out spam emails, and even launch DDoS attacks.

This chapter describes the botnet threat and the countermeasures available to network security professionals. First, it provides an overview of botnets, including their origins, structure, and underlying motivation. Next, the chapter describes existing methods for defending computers and networks against botnets. Finally, it addresses the most important aspect of the botnet problem: how to identify and track the botmaster in order to eliminate the root cause of the botnet problem.

2. Botnet Overview

Bots and botnets are the latest trend in the evolution of Internet malware. Their black-hat developers have built on the experience gathered from decades of viruses, worms, Trojan horses, and other malware to create highly sophisticated software that is difficult to detect and remove. Typical botnets have several hundred to several thousand members, though some botnets have been detected with over 1.5 million members [3]. As of January 2007, Google's Vinton Cerf estimated that up to 150 million computers (about 25% of all Internet hosts) could be infected with bot software [4].

Origins of Botnets

Before botnets, the main motivation for Internet attacks was fame and notoriety. By design, these attacks were noisy and easily detected. High-profile examples are the Melissa email worm (1999), ILOVEYOU (2000), Code Red (2001), Slammer (2003), and Sasser (2004) [5, 6]. Though the impact of these viruses and worms was severe, the damage was relatively short-lived and consisted mainly of the cost of the outage plus man-hours required for cleanup. Once the infected files had been removed from the victim computers and the vulnerability patched, the attackers no longer had any control.

By contrast, botnets are built on the very premise of extending the attacker's control over his victims. To achieve long-term control, a bot must be stealthy during every part of its lifecycle, unlike its predecessors [2]. As a result, most bots have a relatively small network footprint and do not create much traffic during typical operation. Once a bot is in place, the only required traffic consists of incoming commands and outgoing responses, constituting the botnet's command and control (C&C) channel. Therefore, the scenario at the beginning of the chapter is not typical of all botnets. Such an obvious attack points to either a brazen or inexperienced botmaster, and there are plenty of them.

The concept of a remote-controlled computer bot originates from Internet Relay Chat (IRC), where benevolent bots were first introduced to help with repetitive administrative tasks such as channel and nickname management [1, 2]. One of the first implementations of such an IRC bot was Eggdrop, originally developed in 1993 and still one of the most popular IRC bots [6, 7]. Over time, attackers realized that IRC was in many ways a perfect medium for large-scale botnet C&C. It provides an instantaneous one-to-many communications channel and can support very large numbers of concurrent users [8].

Botnet Topologies and Protocols

In addition to the traditional IRC-based botnets, several other protocols and topologies have emerged recently. The two main botnet topologies are centralized and peer-to-peer (P2P). Among centralized botnets, IRC is still the predominant protocol, [9–11] but this trend is decreasing and several recent bots have used HTTP for their C&C channels [9, 11]. Among P2P botnets, many different protocols exist, but the general idea is to use a decentralized collection of peers and thus eliminate the single point of failure found in centralized botnets. P2P is becoming the most popular botnet topology because it has many advantages over centralized botnets [12].

Centralized

Centralized botnets use a single entity (a host or a small collection of hosts) to manage all bot members. The advantage of a centralized topology is that it is fairly easy to implement and produces little overhead. A major disadvantage is that the entire botnet becomes useless if the central entity is removed, since bots will attempt to connect to nonexistent servers. To provide redundancy against this problem, many modern botnets rely on dynamic DNS services and/or fast-flux DNS techniques. In a fast-flux configuration, hundreds or thousands of compromised hosts are used as proxies to hide the identities of the true C&C servers. These hosts constantly alternate in a round-robin DNS configuration to resolve one hostname to many different IP addresses (none of which are the true IPs of C&C servers). Only the proxies know the true C&C servers, forwarding all traffic from the bots to these servers [13].

As we've described, the IRC protocol is an ideal candidate for centralized botnet control, and it remains the most popular among in-the-wild botmasters, [9–11] although it appears that will not be true much longer. Popular examples of IRC bots are Agobot, Spybot, and Sdbot [13]. Variants of these three families make up most active botnets today. By its nature, IRC is centralized and allows nearly instant communication among large botnets. One of the major disadvantages is that IRC traffic is not very common on the Internet, especially in an enterprise setting. As a result, standard IRC traffic can be easily detected, filtered, or blocked. For this reason, some botmasters run their IRC servers on nonstandard ports. Some even use customized IRC implementations, replacing easily recognized commands such as JOIN and PRIVMSG with other text. Despite these countermeasures, IRC still tends to stick out from the regular Web and email traffic due to uncommon port numbers.

Recently, botmasters have started using HTTP to manage their centralized botnets. The advantage of using regular Web traffic for C&C is that it must be allowed to pass through virtually all firewalls, since HTTP comprises a majority of Internet traffic. Even closed firewalls that only provide Web access (via a proxy service, for example) will allow HTTP traffic to pass. It is possible to inspect the content and attempt to filter out malicious C&C traffic, but this is not feasible due to the large number of existing bots and variants. If botmasters use HTTPS (HTTP encrypted using SSL/TLS), then even content inspection becomes useless and all traffic must be allowed to pass through the firewall. However, a disadvantage of HTTP is that it does not provide the instant communication and built-in, scale-up properties of IRC: Bots must manually poll the central server at specific intervals. With large botnets, these intervals must be large enough and distributed well to avoid overloading the server with simultaneous requests. Examples of HTTP bots are Bobax [11, 14] and Rustock, with Rustock using a custom encryption scheme on top of HTTP to conceal its C&C traffic [15].

Peer-to-Peer

As defenses against centralized botnets have become more effective, more and more botmasters are exploring ways to avoid the pitfalls of relying on a centralized architecture and therefore a single point of failure. Symantec reports a "steady decrease" in centralized

IRC botnets and predicts that botmasters are now "accelerating their shift. . .to newer, stealthier control methods, using protocols such as. . .peer-to-peer" [12]. In the P2P model, no centralized server exists, and all member nodes are equally responsible for passing on traffic. "If done properly, [P2P] makes it near impossible to shut down the botnet as a whole. It also provides anonymity to the [botmaster], because they can appear as just another node in the network," says security researcher Joe Stewart of Lurhq [16]. There are many protocols available for P2P networks, each differing in the way nodes first join the network and the role they later play in passing traffic along. Some popular protocols are BitTorrent, WASTE, and Kademlia [13]. Many of these protocols were first developed for benign uses, such as P2P file sharing.

One of the first malicious P2P bots was Sinit, released in September 2003. It uses random scanning to find peers rather than relying on one of the established P2P bootstrap protocols [13]. As a result, Sinit often has trouble finding peers, which results in overall poor connectivity [17]. Due to the large amount of scanning traffic, this bot is easily detected by intrusion detection systems (IDSs) [18].

Another advanced bot using the P2P approach is Nugache, released in April 2006 [13]. It initially connects to a list of 22 predefined peers to join the P2P network and then downloads a list of active peer nodes from there. This implies that if the 22 "seed" hosts can be shut down, no new bots will be able to join the network, but existing nodes can still function [19]. Nugache encrypts all communications, making it harder for IDSs to detect and increasing the difficulty of manual analysis by researchers [16]. Nugache is seen as one of the first more sophisticated P2P bots, paving the way for future enhancements by botnet designers.

The most famous P2P bot so far is Peacomm, more commonly known as the Storm Worm. It started spreading in January 2007 and continues to have a strong presence [20]. To communicate with peers, it uses the Overnet protocol, based on the Kademlia P2P protocol. For bootstrapping, it uses a fixed list of peers (146 in one observed instance) distributed along with the bot. Once the bot has joined Overnet, the botmaster can easily update the binary and add components to extend its functionality. Often the bot is configured to automatically retrieve updates and additional components, such as an SMTP server for spamming, an email address harvesting tool, and a DoS module. Like Nugache, all of Peacomm's communications are encrypted, making it extremely hard to observe C&C traffic or inject commands appearing to come from the botmaster. Unlike centralized botnets relying on a dynamic DNS provider, Peacomm uses its own P2P network as a distributed DNS system that has no single point of failure. The fixed list of peers is a potential weakness, although it would be challenging to take all these nodes offline. Additionally, the attackers can always set up new nodes and include an updated peer list with the bot, resulting in an "arms race" to shut down malicious nodes [13].

3. Typical Bot Life Cycle

Regardless of the topology being used, the typical life cycle of a bot is similar:

1. *Creation*. First, the botmaster develops his bot software, often reusing existing code and adding custom features. He might use a test network to perform dry runs before deploying the bot in the wild.
2. *Infection*. There are many possibilities for infecting victim computers, including the following four. Once a victim machine becomes infected with a bot, it is known as a *zombie*.
   - *Software vulnerabilities*. The attacker exploits a vulnerability in a running service to automatically gain access and install his software without any user interaction. This was the method used by most worms, including the infamous Code Red and Sasser worms [5].
   - *Drive-by download*. The attacker hosts his file on a Web server and entices people to visit the site. When the user loads a certain page, the software is automatically installed without user interaction, usually by exploiting browser bugs, misconfigurations, or unsecured ActiveX controls.
   - *Trojan horse*. The attacker bundles his malicious software with seemingly benign and useful software, such as screen savers, antivirus scanners, or games. The user is fully aware of the installation process, but he does not know about the hidden bot functionality.
   - *Email attachment*: Although this method has become less popular lately due to rising user awareness, it is still around. The attacker sends an attachment that will automatically install the bot software when the user opens it, usually without any interaction. This was the primary infection vector of the ILOVEYOU email worm from 2000 [5]. The recent Storm Worm successfully used enticing email messages with executable attachments to lure its victims [20].
3. *Rallying*. After infection, the bot starts up for the first time and attempts to contact its C&C server(s) in a process known as *rallying*. In a centralized botnet, this could be an IRC or HTTP server, for example. In a P2P botnet, the bots perform the bootstrapping protocol required to locate other peers and join the network. Most bots are very fault-tolerant, having multiple lists of backup servers to attempt if the primary ones become unavailable. Some C&C servers are configured to immediately send some initial commands to the bot (without botmaster intervention). In an IRC botnet, this is typically done by including the commands in the C&C channel's topic.
4. *Waiting*. Having joined the C&C network, the bot waits for commands from the botmaster. During this time, very little (if any) traffic passes between the victim and the C&C servers. In an IRC botnet, this traffic would mainly consist of periodic keep-alive messages from the server.

5. *Executing*. Once the bot receives a command from the botmaster, it executes it and returns any results to the botmaster via the C&C network. The supported commands are only limited by the botmaster's imagination and technical skills. Common commands are in line with the major uses of botnets: scanning for new victims, sending spam, sending DoS floods, setting up traffic redirection, and many more.

Following execution of a command, the bot returns to the waiting state to await further instructions. If the victim computer is rebooted or loses its connection to the C&C network, the bot resumes in the rallying state. Assuming it can reach its C&C network, it will then continue in the waiting state until further commands arrive.

Figure 8.1 shows the detailed infection sequence in a typical IRC-based botnet.

1. An existing botnet member computer launches a scan, then discovers and exploits a vulnerable host.
2. Following the exploit, the vulnerable host is made to download and install a copy of the bot software, constituting an infection.
3. When the bot starts up on the vulnerable host, it enters the rallying state: It performs a DNS lookup to determine the current IP of its C&C server.
4. The new bot joins the botnet's IRC channel on the C&C server for the first time, now in the waiting state.
5. The botmaster sends his commands to the C&C server on the botnet's IRC channel.
6. The C&C server forwards the commands to all bots, which now enter the executing state.

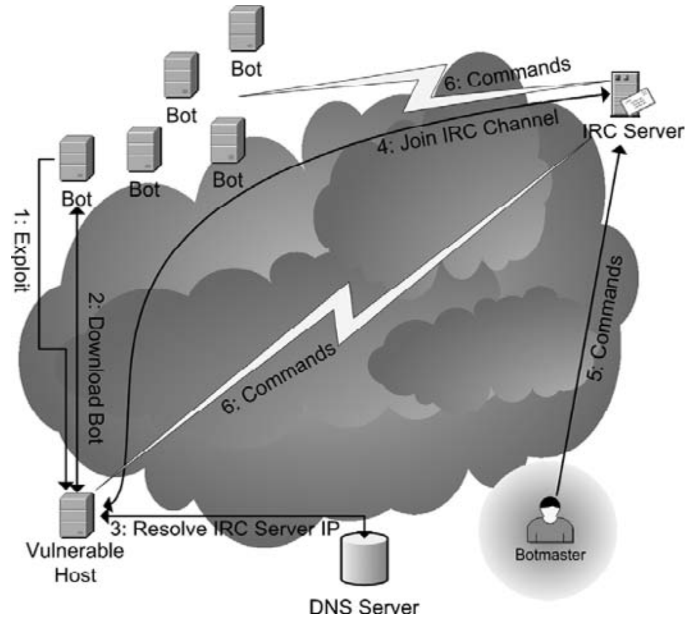


Figure 8.1: Infection sequence of a typical centralized IRC-based botnet.

4. The Botnet Business Model

Unlike the viruses and worms of the past, botnets are motivated by financial profit. Organized crime groups often use them as a source of income, either by hiring "freelance" botmasters or by having their own members create botnets. As a result, network security professionals are up against motivated, well-financed organizations that can often hire some of the best minds in computers and network security. This is especially true in countries such as Russia, Romania, and other Eastern European nations where there is an abundance of IT talent at the high school and university level but legitimate IT job prospects are very limited. In such an environment, criminal organizations easily recruit recent graduates by offering far better opportunities than the legitimate job market [21–24]. One infamous example of such a crime organization is the Russian Business Network (RBN), a Russian Internet service provider (ISP) that openly supports criminal activity [21, 25]. They are responsible for the Storm Worm (Peacomm), [25] the March 2007 DDoS attacks on Estonia, [25] and a high-profile attack on the Bank of India in August 2007, [26] along with many other attacks.

It might not be immediately obvious how a collection of computers can be used to cause havoc and produce large profits. The main point is that botnets provide *anonymous* and *distributed* access to the Internet. The anonymity makes the attackers untraceable, and a botnet's distributed nature makes it extremely hard to shut down. As a result, botnets are perfect vehicles for criminal activities on the Internet. Some of the main profit-producing methods are explained here, [27] but criminals are always devising new and creative ways to profit from botnets:

- *Spam.* Spammers send millions of emails advertising phony or overpriced products, phishing for financial data and login information, or running advance-fee schemes such as the Nigerian 419 scam [28]. Even if only a small percentage of recipients respond to this spam, the payoff is considerable for the spammer. It is estimated that up to 90% of all spam originates from botnets [2].
- *DDoS and extortion.* Having amassed a large number of bots, the attacker contacts an organization and threatens to launch a massive DDoS attack, shutting down its Web site for several hours or even days. Another variation on this method is to find vulnerabilities, use them steal financial or confidential data, and then demand money for the "safe return" of the data and to keep it from being circulated in the underground economy [23]. Often, companies would rather pay off the attacker to avoid costly downtime, lost sales, and the lasting damage to its reputation that would result from a DDoS attack or data breach.
- *Identity theft.* Once a bot has a foothold on a victim's machine, it usually has complete control. For example, the attacker can install keyloggers to record login and password information, search the hard drive for valuable data, or alter the DNS configuration to

redirect victims to look-alike Web sites and collect personal information, known as *pharming [29]*. Using the harvested personal information, the attacker can make fraudulent credit card charges, clean out the victim's bank account, and apply for credit in the victim's name, among many other things.
- *Click fraud.* In this scenario, bots are used to repeatedly click Web advertising links, generating per-click revenue for the attacker [2]. This represents fraud because only the clicks of human users with a legitimate interest are valuable to advertisers. The bots will not buy the product or service as a result of clicking the advertisement.

These illegal activities are extremely profitable. For example, a 2006 study by the Germany Honeynet Project estimated that a botmaster can make about $430 per day just from per-install advertising software [30]. A 20-year-old California botmaster indicted in February 2006 earned $100,000 in advertising revenue from his botnet operations [31]. However, both of these cases pale in comparison to the estimated $20 million worth of damage caused by an international ring of computer criminals known as the A-Team [32].

Due to these very profitable uses of botnets, many botmasters make money simply by creating botnets and then renting out processing power and bandwidth to spammers, extortionists, and identity thieves. Despite a recent string of high-profile botnet arrests, these are merely a drop in the bucket [4]. Overall, botmasters still have a fairly low chance of getting caught due to a lack of effective traceback techniques. The relatively low risk combined with high yield makes the botnet business very appealing as a fundraising method for criminal enterprises, especially in countries with weak computer crime enforcement.

5. Botnet Defense

When botnets emerged, the response was similar to previous Internet malware: Antivirus vendors created signatures and removal techniques for each new instance of the bot. This approach initially worked well at the host level, but researchers soon started exploring more advanced methods for eliminating more than one bot at a time. After all, a botnet with tens of thousands of members would be very tedious to combat one bot at a time.

This section describes the current defenses against centralized botnets, moving from the host level to the network level, then to the C&C server, and finally to the botmaster himself.

Detecting and Removing Individual Bots

Removing individual bots does not usually have a noticeable impact on the overall botnet, but it is a crucial first step in botnet defense. The basic antivirus approach using signature-based detection is still effective with many bots, but some are starting to use polymorphism, which creates unique instances of the bot code and evades signature-based detection. For example, Agobot is known to have thousands of variants, and it includes built-in support for polymorphism to change its signature at will [33].

To deal with these more sophisticated bots and all other polymorphic malware, detection must be done using behavioral analysis and heuristics. Researchers Stinson and Mitchell have developed a taint-based approach called BotSwat that marks all data originating from the network. If this data is used as input for a system call, there is a high probability that it is bot-related behavior, since user input typically comes from the keyboard or mouse on most end-user systems [34].

Detecting C&C Traffic

To mitigate the botnet problem on a larger scale, researchers turned their attention to network-based detection of the botnet's C&C traffic. This method allows organizations or even ISPs to detect the presence of bots on their entire network, rather than having to check each machine individually.

One approach is to examine network traffic for certain known patterns that occur in botnet C&C traffic. This is, in effect, a network-deployed version of signature-based detection, where signatures have to be collected for each bot before detection is possible. Researchers Goebel and Holz implemented this method in their Rishi tool, which evaluates IRC nicknames for likely botnet membership based on a list of known botnet naming schemes. As with all signature-based approaches, it often leads to an "arms race" where the attackers frequently change their malware and the network security community tries to keep up by creating signatures for each new instance [35].

Rather than relying on a limited set of signatures, it is also possible to use the IDS technique of anomaly detection to identify unencrypted IRC botnet traffic. This method was successfully implemented by researchers Binkley and Singh at Portland State University, and as a result they reported a significant increase in bot detection on the university network [36].

Another IDS-based detection technique called BotHunter was proposed by Gu et al. in 2007. Their approach is based on IDS dialog correlation techniques: It deploys three separate network monitors at the network perimeter, each detecting a specific stage of bot infection. By correlating these events, BotHunter can reconstruct the traffic dialog between the infected machine and the outside Internet. From this dialog, the engine determines whether a bot infection has taken place with a high accuracy rate [37].

Moving beyond the scope of a single network/organization, traffic from centralized botnets can be detected at the ISP level based only on transport layer flow statistics. This approach was developed by Karasaridis et al. and solves many of the problems of packet-level inspection. It is passive, highly scalable, and only uses flow summary data (limiting privacy issues). Additionally, it can determine the size of a botnet without joining and can even detect botnets using encrypted C&C. The approach exploits the underlying principle of centralized botnets: Each bot has to contact the C&C server, producing detectable patterns in network traffic flows [38].

Beyond the ISP level, a heuristic method for Internet-wide bot detection was proposed by Ramachandran et al. in 2006. In this scheme, query patterns of DNS black-hole lists (DNSBLs) are used to create a list of possible bot-infected IP addresses. It relies on the fact that botmasters need to periodically check whether their spam-sending bots have been added to a DNSBL and have therefore become useless. The query patterns of botmasters to a DNSBL are very different from those of legitimate mail servers, allowing detection [39]. One major limitation is that this approach focuses mainly on the sending of spam. It would most likely not detect bots engaged in other illegal activities, such as DDoS attacks or click fraud, since these do not require DNSBL lookups.

Detecting and Neutralizing the C&C Servers

Though detecting C&C traffic and eliminating all bots on a given local network is a step in the right direction, it still doesn't allow the takedown of an entire botnet at once. To achieve this goal in a centralized botnet, access to the C&C servers must be removed. This approach assumes that the C&C servers consist of only a few hosts that are accessed directly. If hundreds or thousands of hosts are used in a fast-flux proxy configuration, it becomes extremely challenging to locate and neutralize the true C&C servers.

In work similar to BotHunter, researchers Gu et al. developed BotSniffer in 2008. This approach represents several improvements, notably that BotSniffer can handle encrypted traffic, since it no longer relies only on content inspection to correlate messages. A major advantage of this approach is that it requires no advance knowledge of the bot's signature or the identity of C&C servers. By analyzing network traces, BotSniffer detects the spatial-temporal correlation among C&C traffic belonging to the same botnet. It can therefore detect both the bot members and the C&C server(s) with a low false positive rate [40].

Most of the approaches mentioned under "Detecting C&C Traffic" can also be used to detect the C&C servers, with the exception of the DNSBL approach [39]. However, their focus is mainly on detection and removal of individual bots. None of these approaches mentions targeting the C&C servers to eliminate an entire botnet.

One of the few projects that has explored the feasibility of C&C server takedown is the work of Freiling et al. in 2005 [41]. Although their focus is on DDoS prevention, they describe the method that is generally used in the wild to remove C&C servers when they are detected. First, the bot binary is either reverse-engineered or run in a sandbox to observe its behavior, specifically the hostnames of the C&C servers. Using this information, the proper dynamic DNS providers can be notified to remove the DNS entries for the C&C servers, preventing any bots from contacting them and thus severing contact between the botmaster and his botnet. Dagon et al. used a similar approach in 2006 to obtain experiment data for modeling botnet propagation, redirecting the victim's connections from the true C&C server to their sinkhole host [42]. Even though effective, the manual analysis and contact with the DNS operator is a slow process. It can take up to several days until all C&C servers are located and neutralized. However, this process is essentially the best available approach for shutting down entire botnets in the wild. As we mentioned, this technique becomes much harder when fast-flux proxies are used to conceal the real C&C servers or a P2P topology is in place.

Attacking Encrypted C&C Channels

Though some of the approaches can detect encrypted C&C traffic, the presence of encryption makes botnet research and analysis much harder. The first step in dealing with these advanced botnets is to penetrate the encryption that protects the C&C channels.

A popular approach for adding encryption to an existing protocol is to run it on top of SSL/TLS; to secure HTTP traffic, ecommerce Web sites run HTTP over SSL/TLS, known as HTTPS. Many encryption schemes that support key exchange (including SSL/TLS) are susceptible to man-in-the-middle (MITM) attacks, whereby a third party can impersonate the other two parties to each other. Such an attack is possible only when no authentication takes place prior to the key exchange, but this is a surprisingly common occurrence due to poor configuration.

The premise of an MITM attack is that the client does not verify that it's talking to the real server, and vice versa. When the MITM receives a connection from the client, it immediately creates a separate connection to the server (under a different encryption key) and passes on the client's request. When the server responds, the MITM decrypts the response, logs and possibly alters the content, then passes it on to the client reencrypted with the proper key. Neither the client nor the server notice that anything is wrong, because they are communicating with each other over an encrypted connection, as expected.
The important difference is that unknown to either party, the traffic is being decrypted and reencrypted by the MITM in transit, allowing him to observe and alter the traffic.

In the context of bots, two main attacks on encrypted C&C channels are possible:
(1) "gray-box" analysis, whereby the bot communicates with a local machine impersonating the C&C server, and (2) a full MITM attack, in which the bot communicates with the true C&C server. Figure 8.2 shows a possible setup for both attacks, using the DeleGate proxy [43] for the conversion to and from SSL/TLS.

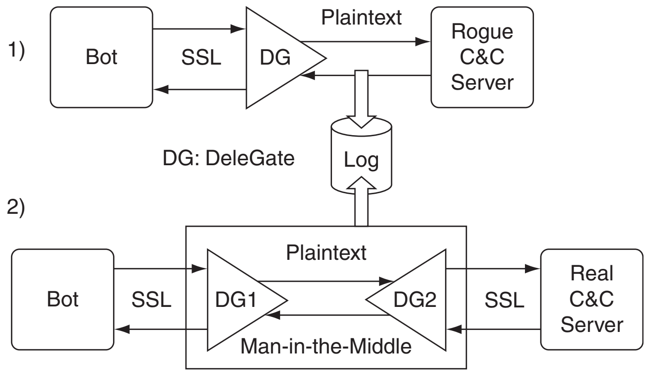


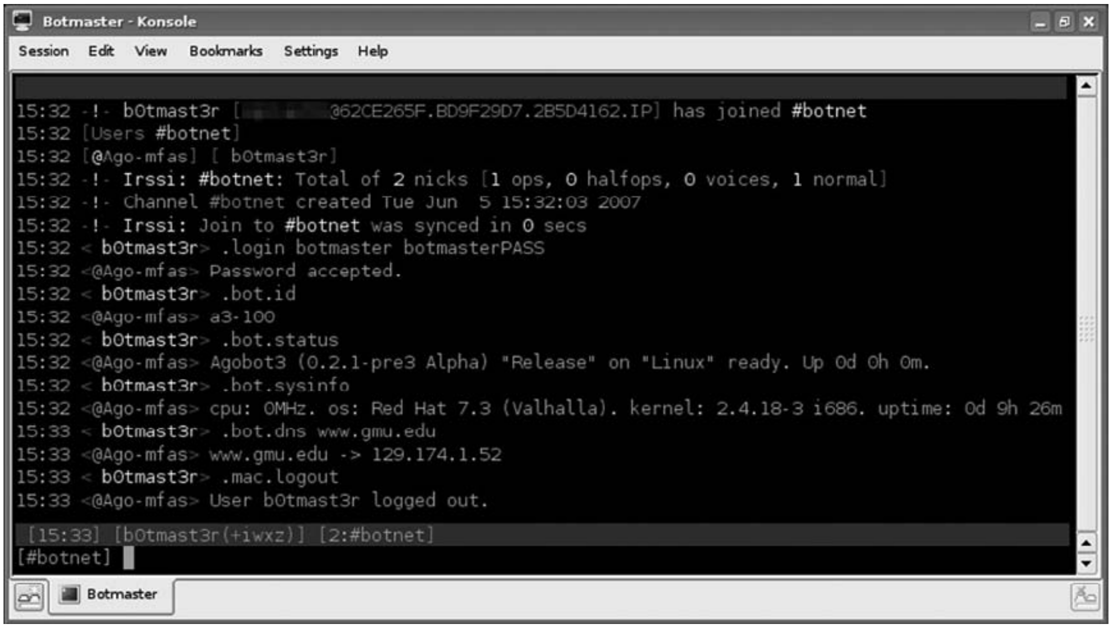
Figure 8.2: Setups for man-in-the-middle attacks on encrypted C&C channels.

The first attack is valuable to determine the authentication information required to join the live botnet: the address of the C&C server, the IRC channel name (if applicable), plus any required passwords. However, it does not allow the observer to see the interaction with the larger botnet, specifically the botmaster. The second attack reveals the full interaction with the botnet, including all botmaster commands, the botmaster password used to control the bots, and possibly the IP addresses of other bot members (depending on the configuration of the C&C server). Figures 8.3–8.5 show the screenshots of the full MITM attack on a copy of Agobot configured to connect to its C&C server via SSL/TLS. Specifically, Figure 8.3 shows the botmaster's IRC window, with his commands and the bot's responses. Figure 8.4 shows the encrypted SSL/TLS trace, and Figure 8.5 shows the decrypted plaintext that was observed at the DeleGate proxy. The botmaster password *botmasterPASS* is clearly visible, along with the required username, *botmaster*.

Armed with the botmaster username and password, the observer could literally take over the botnet. He could log in as the botmaster and then issue a command such as Agobot's .bot. remove, causing all bots to disconnect from the botnet and permanently remove themselves from the infected computers. Unfortunately, there are legal issues with this approach because it constitutes unauthorized access to all the botnet computers, despite the fact that it is in fact a benign command to remove the bot software.

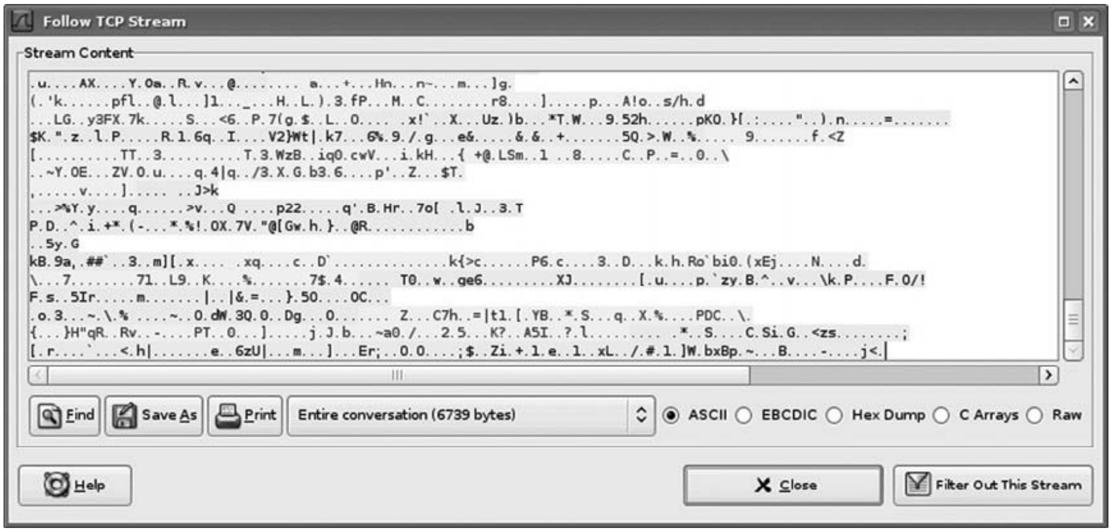
Locating and Identifying the Botmaster

Shutting down an entire botnet at once is a significant achievement, especially when the botnet numbers in the tens of thousands of members. However, there is nothing stopping the botmaster from simply deploying new bots to infect the millions of vulnerable hosts on the Internet, creating a new botnet in a matter of hours. In fact, most of the machines belonging to the shut-down botnet are likely to become infected again because the vulnerabilities and any attacker-installed backdoors often remain active, despite the elimination of the C&C servers. Botnet-hunting expert Gadi Evron agrees: "When we



```
15:32 -!- b0tmast3r [[redacted]@62CE265F.BD9F29D7.2B5D4162.IP] has joined #botnet
15:32 [Users #botnet]
15:32 [@Ago-mfas] [ b0tmast3r]
15:32 -!- Irssi: #botnet: Total of 2 nicks [1 ops, 0 halfops, 0 voices, 1 normal]
15:32 -!- Channel #botnet created Tue Jun  5 15:32:03 2007
15:32 -!- Irssi: Join to #botnet was synced in 0 secs
15:32 < b0tmast3r> .login botmaster botmasterPASS
15:32 <@Ago-mfas> Password accepted.
15:32 < b0tmast3r> .bot.id
15:32 <@Ago-mfas> a3-100
15:32 < b0tmast3r> .bot.status
15:32 <@Ago-mfas> Agobot3 (0.2.1-pre3 Alpha) "Release" on "Linux" ready. Up 0d 0h 0m.
15:32 < b0tmast3r> .bot.sysinfo
15:32 <@Ago-mfas> cpu: 0MHz. os: Red Hat 7.3 (Valhalla). kernel: 2.4.18-3 i686. uptime: 0d 9h 26m
15:33 < b0tmast3r> .bot.dns www.gmu.edu
15:33 <@Ago-mfas> www.gmu.edu -> 129.174.1.52
15:33 < b0tmast3r> .mac.logout
15:33 <@Ago-mfas> User b0tmast3r logged out.
 [15:33] [b0tmast3r(+iwxz)] [2:#botnet]
[#botnet]
```

Figure 8.3: Screenshot showing the botmaster's IRC window.



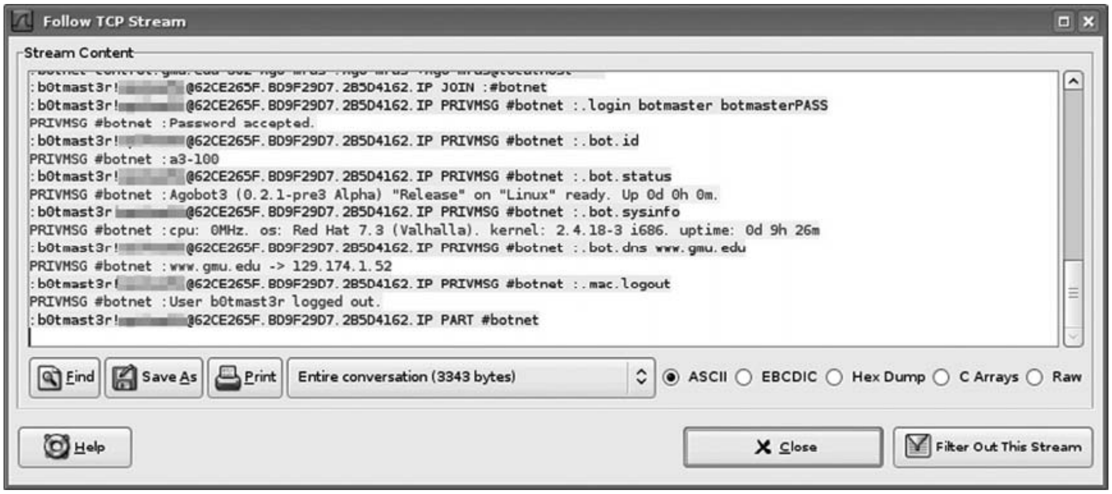
```
.u....AX....Y.Oa..R.v...@........ a...+...Hn...n~...m...]g.
(.'k......pfl..@.l...]1..._...H..L.).3.fP...M..C........r8....].....p...A!o..s/h.d
...LG..y3FX.7k.....S...<6..P.7(g.$..L..O.... .x!`..X...Uz.)b...*T.W...9.52h......pKO.}{.:....."..).n.....=.......
$K.".z..l.P.....R.1.6q..I....V2}Wt|.k7...6%.9./.g...e&.....&.&..+.......5Q.>.W..%..... 9.......f.<Z
[..........TT..3..........T.3.WzB..iq0.cwV...i.kH...{ +@.LSm..1 ..8.....C..P..=..0..\
..~Y.OE...ZV.O.u....q.4|q../3.X.G.b3.6....p'..Z...$T.
,.....v....]..... ..J>k
...>%Y.y....q......>v...Q ....p22.....q'.B.Hr..7o[ .l.J..3.T
P.D..^.i.+*.(-...*.%!.OX.7V."@[Gw.h.}..@R...........b
..5y.G
kB.9a,.##`..3..m][.x.... .xq....c..D`..............k{>c......P6.c....3..D...k.h.Ro`bi0.(xEj....N....d.
\...7.........71..L9..K....%.......7$.4...... T0..w..ge6.........XJ........[.u....p.`zy.B.^..v...\k.P....F.0/!
F.s..5Ir.....m.......|..|&.=...}.50....OC...
.o.3...~.\.% ....~..0.dW.3Q.0..Dg...0....... Z...C7h..=|t1.[.YB..*.S...q..X.%....PDC..\.
{...}H"qR..Rv..-....PT..0...].....j.J.b...~a0./...2.5...K?..A5I..?.l......... .*..S....C.Si.G..<zs........;
[.r....`...<.h|.......e..6zU|...m...]...Er;..0.0....;$..Zi.+.1.e..1..xL../.#.1.]W.bxBp.~...B....-....j<.
```

Entire conversation (6739 bytes) [✓ ASCII] EBCDIC Hex Dump C Arrays Raw

Figure 8.4: Screenshot showing the SSL/TLS-encrypted network traffic.

disable a command-and-control server, the botnet is immediately recreated on another host. We're not hurting them anymore," he said in a 2006 interview [44].

The only permanent solution of the botnet problem is to go after the root cause: the botmasters. Unfortunately, most botmasters are very good at concealing their identities and



```
:b0tmast3r![redacted]@62CE265F.BD9F29D7.2B5D4162.IP JOIN :#botnet
:b0tmast3r![redacted]@62CE265F.BD9F29D7.2B5D4162.IP PRIVMSG #botnet :.login botmaster botmasterPASS
PRIVMSG #botnet :Password accepted.
:b0tmast3r![redacted]@62CE265F.BD9F29D7.2B5D4162.IP PRIVMSG #botnet :.bot.id
PRIVMSG #botnet :a3-100
:b0tmast3r![redacted]@62CE265F.BD9F29D7.2B5D4162.IP PRIVMSG #botnet :.bot.status
PRIVMSG #botnet :Agobot3 (0.2.1-pre3 Alpha) "Release" on "Linux" ready. Up 0d 0h 0m.
:b0tmast3r[redacted]@62CE265F.BD9F29D7.2B5D4162.IP PRIVMSG #botnet :.bot.sysinfo
PRIVMSG #botnet :cpu: 0MHz. os: Red Hat 7.3 (Valhalla). kernel: 2.4.18-3 i686. uptime: 0d 9h 26m
:b0tmast3r![redacted]@62CE265F.BD9F29D7.2B5D4162.IP PRIVMSG #botnet :.bot.dns www.gmu.edu
PRIVMSG #botnet :www.gmu.edu -> 129.174.1.52
:b0tmast3r![redacted]@62CE265F.BD9F29D7.2B5D4162.IP PRIVMSG #botnet :.mac.logout
PRIVMSG #botnet :User b0tmast3r logged out.
:b0tmast3r![redacted]@62CE265F.BD9F29D7.2B5D4162.IP PART #botnet
```

Find
Save As
Print
Entire conversation (3343 bytes)
(●) ASCII ( ) EBCDIC ( ) Hex Dump ( ) C Arrays ( ) Raw
Help
Close
Filter Out This Stream

Figure 8.5: Screenshot showing decrypted plaintext from the DeleGate proxy.

locations, since their livelihood depends on it. Tracking the botmaster to her true physical location is a complex problem that is described in detail in the next section. So far, there is no published work that would allow automated botmaster traceback on the Internet, and it remains an open problem.

6. Botmaster Traceback

The botnet field is full of challenging problems: obfuscated binaries, encrypted C&C channels, fast-flux proxies protecting central C&C servers, customized communication protocols, and many more (see Figure 8.6). Arguably the most challenging task is locating the botmaster. Most botmasters take precautions on multiple levels to ensure that their connections cannot be traced to their true locations.

The reason for the botmaster's extreme caution is that a successful trace would have disastrous consequences. He could be arrested, his computer equipment could be seized and scrutinized in detail, and he could be sentenced to an extended prison term. Additionally, authorities would likely learn the identities of his associates, either from questioning him or by searching his computers. As a result, he would never again be able to operate in the Internet underground and could even face violent revenge from his former associates when he is released.

In the United States, authorities have recently started to actively pursue botmasters, resulting in several arrests and convictions. In November 2005, 20-year-old Jeanson James Ancheta of California was charged with botnet-related computer offenses [45]. He pleaded guilty in January 2006 and could face up to 25 years in prison [46]. In a similar case, 20-year-old

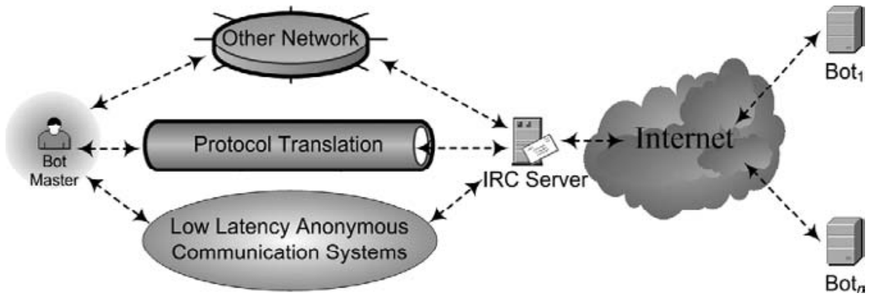


Figure 8.6: Botnet C&C traffic laundering.

Christopher Maxwell was indicted on federal computer charges. He is accused of using his botnet to attack computers at several universities and a Seattle hospital, where bot infections severely disrupted operations [31].

In particular, the FBI's Operation Bot Roast has resulted in several high-profile arrests, both in the United States and abroad [47]. The biggest success was the arrest of 18-year-old New Zealand native Owen Thor Walker, who was a member of a large international computer crime ring known as the A-Team. This group is reported to have infected up to 1.3 million computers with bot software and caused about $20 million in economic damage. Despite this success, Walker was only a minor player, and the criminals in control of the A-Team are still at large [32].

Unfortunately, botmaster arrests are not very common. The cases described here represent only several individuals; thousands of botmasters around the world are still operating with impunity. They use sophisticated techniques to hide their true identities and locations, and they often operate in countries with weak computer crime enforcement. The lack of international coordination, both on the Internet and in law enforcement, makes it hard to trace botmasters and even harder to hold them accountable to the law [22].

Traceback Challenges

One defining characteristic of the botmaster is that he originates the botnet C&C traffic. Therefore, one way to find the botmaster is to track the botnet C&C traffic. To hide himself, the botmaster wants to disguise his link to the C&C traffic via various traffic-laundering techniques that make tracking C&C traffic more difficult. For example, a botmaster can route his C&C traffic through a number of intermediate hosts, various protocols, and low-latency anonymous networks to make it extremely difficult to trace. To further conceal his activities, a botmaster can also encrypt his traffic to and from the C&C servers. Finally, a botmaster only needs to be online briefly and send small amounts of traffic to interact with his botnet, reducing the chances of live traceback. Figure 8.6 illustrates some of the C&C traffic-laundering techniques a botmaster can use.

Stepping Stones

The intermediate hosts used for traffic laundering are known as *stepping stones*. The attacker sets them up in a chain, leading from the botmaster's true location to the C&C server. Stepping stones can be SSH servers, proxies (such as SOCKS), IRC bouncers (BNCs), virtual private network (VPN) servers, or any number of network redirection services. They usually run on compromised hosts, which are under the attacker's control and lack audit/logging mechanisms to trace traffic. As a result, manual traceback is tedious and time-consuming, requiring the cooperation of dozens of organizations whose networks might be involved in the trace.

The major challenge posed by stepping stones is that all routing information from the previous hop (IP headers, TCP headers, and the like) is stripped from the data before it is sent out on a new, separate connection. Only the content of the packet (the application layer data) is preserved, which renders many existing tracing schemes useless. An example of a technique that relies on routing header information is *probabilistic packet marking*. This approach was introduced by Savage et al. in 2000, embedding tracing information in an unused IP header field [48]. Two years later, Goodrich expanded this approach, introducing "randomize-and-link" for better scalability [49]. Another technique for IP-level traceback is the log/hash-based scheme introduced by Snoeren et al. [50] and enhanced by Li et al. [51] These techniques were very useful in combating the fast-spreading worms of the early 2000s, which did not use stepping stones. However, these approaches do not work when stepping stones are present, since IP header information is lost.

Multiple Protocols

Another effective and efficient method to disguise the botmaster is to launder the botnet C&C traffic across other protocols. Such protocol laundering can be achieved by either *protocol tunneling* or *protocol translation*. For example, a sophisticated botmaster could route its command and control traffic through SSH (or even HTTP) tunnels to reach the command and control center. The botmaster could also use some intermediate host *X* as a stepping stone, use some real-time communication protocols other than IRC between the botmaster host and host *X*, and use IRC between the host *X* and the IRC server. In this case, host *X* performs the protocol translation at the application layer and serves as a conduit of the botnet C&C channel. One protocol that is particularly suitable for laundering the botnet command and control is instant messaging (IM), which supports real-time text-based communication between two or more people.

Low-Latency Anonymous Network

Besides laundering the botnet C&C across stepping stones and different protocols, a sophisticated botmaster could anonymize its C&C traffic by routing it through some low-latency anonymous communication systems. For example, Tor—the second generation of onion routing—uses an overlay network of onion routers to provide anonymous outgoing connections and anonymous hidden services. The botmaster could use Tor as a virtual tunnel to anonymize his TCP-based C&C traffic to the IRC server of the botnet. At the same time, the IRC server of the botnet could utilize Tor's hidden services to anonymize the IRC server of the botnet in such a way that its network location is unknown to the bots and yet it could communicate with all the bots.

Encryption

All or part of the stepping stone chain can be encrypted to protect it against content inspection, which could reveal information about the botnet and botmaster. This can be done using a number of methods, including SSH tunneling, SSL/TLS-enabled BNCs, and IPsec tunneling. Using encryption defeats all content-based tracing approaches, so the tracer must rely on other network flow characteristics, such as packet size or timing, to correlate flows to each other.

Low-Traffic Volume

Since the botmaster only has to connect briefly to issue commands and retrieve results from his botnet, a low volume of traffic flows from any given bot to the botmaster. During a typical session, only a few dozen packets from each bot can be sent to the botmaster. Tracing approaches that rely on analysis of packet size or timing will most likely be ineffective because they typically require a large amount of traffic (several hundred packets) to correlate flows with high statistical confidence. Examples of such tracing approaches [52–54] all use timing information to embed a traceable watermark. These approaches can handle stepping stones, encryption, and even low-latency anonymizing network, but they cannot be directly used for botmaster traceback due to the low traffic volume.

Traceback Beyond the Internet

Even if all three technical challenges can be solved and even if all Internet-connected organizations worldwide cooperate to monitor traffic, there are additional traceback challenges beyond the reach of the Internet (see Figure 8.7). Any IP-based traceback method assumes that the true source IP belongs to the computer the attacker is using and that this machine can be physically located. However, in many scenarios this is not true—for example, (1) Internet-connected mobile phone networks, (2) open wireless (Wi-Fi) networks, and (3) public computers, such as those at libraries and Internet cafés.

Most modern cell phones support text-messaging services such as Short Message Service (SMS), and many smart phones also have full-featured IM software. As a result, the botmaster can use a mobile device to control her botnet from any location with cell phone reception. To enable her cell phone to communicate with the C&C server, a botmaster needs

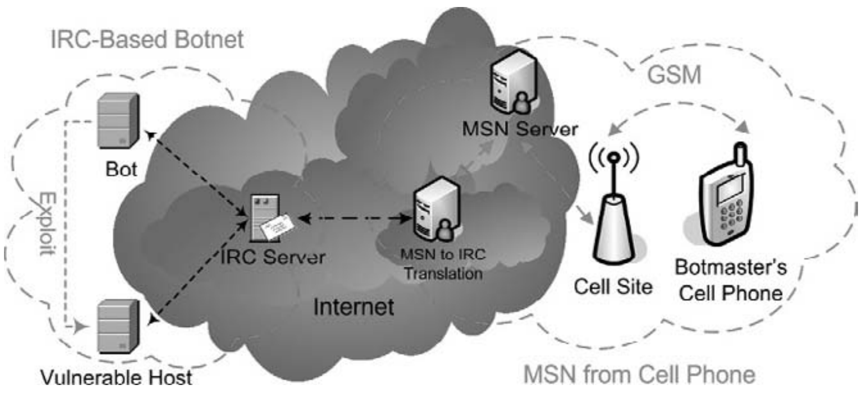
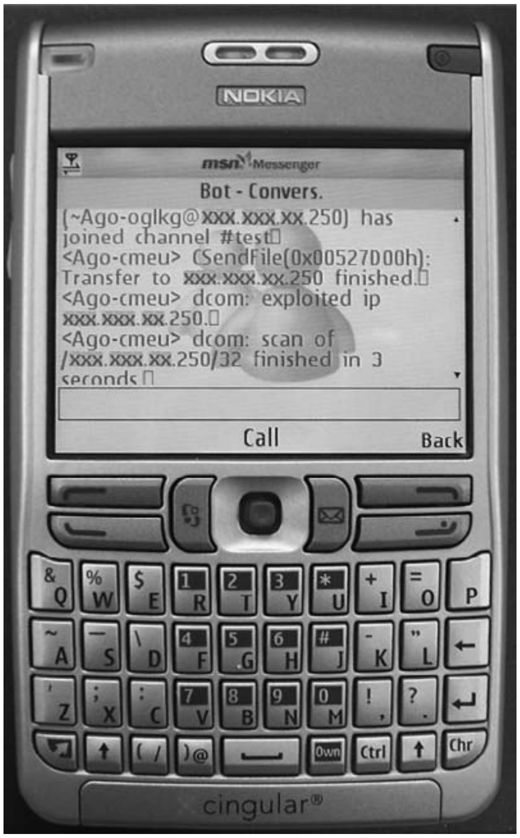


Figure 8.7: Using a cell phone to evade Internet-based traceback.

to use a protocol translation service or a special IRC client for mobile phones. She can run the translation service on a compromised host, an additional stepping stone. For an IRC botnet, such a service would receive the incoming SMS or IM message, then repackage it as an IRC message and send it on to the C&C server (possibly via more stepping stones), as shown in Figure 8.7. To eliminate the need for protocol translation, the botmaster can run a native IRC client on a smart phone with Internet access. Examples of such clients are the Java-based WLIrc [55] and jmIrc [56] open source projects. In Figure 8.8, a Nokia smartphone is shown running MSN Messenger, controlling an Agobot zombie via MSN-IRC protocol translation. On the screen, a new bot has just been infected and has joined the IRC channel following the botmaster's .scan.dcom command.

When a botnet is being controlled from a mobile device, even a perfect IP traceback solution would only reach as far as the gateway host that bridges the Internet and the carrier's mobile network. From there, the tracer can ask the carrier to complete the trace and disclose the name and even the current location of the cell phone's owner. However, there are several problems with this approach. First, this part of the trace again requires lots of manual work and cooperation of yet another organization, introducing further delays and making a real-time trace unlikely. Second, the carrier won't be able to determine the name of the subscriber if he is using a prepaid cell phone. Third, the tracer could obtain an approximate physical location based on cell site triangulation. Even if he can do this in real time, it might not be very useful if the botmaster is in a crowded public place. Short of detaining all people in the area and checking their cell phones, police won't be able to pinpoint the botmaster.

A similar situation arises when the botmaster uses an unsecured Wi-Fi connection. This could either be a public access point or a poorly configured one that is intended to be private. With a strong antenna, the botmaster can be located up to several thousand feet away. In a typical downtown area, such a radius can contain thousands of people and just as many computers. Again, short of searching everyone in the vicinity, the police will be unable to find the botmaster.



```
(~Ago-oglkg@xxx.xxx.xx.250) has
joined channel #test
<Ago-cmeu> (SendFile(0x00527D00h):
Transfer to xxx.xxx.xx.250 finished.
<Ago-cmeu> dcom: exploited ip
xxx.xxx.xx.250.
<Ago-cmeu> dcom: scan of
/xxx.xxx.xx.250/32 finished in 3
seconds
```

Figure 8.8: Using a Nokia smartphone to control an Agobot-based botnet.
(Photo courtesy of Ruishan Zhang.)

Finally, many places provide public Internet access without any logging of the users' identities. Prime examples are public libraries, Internet cafés, and even the business centers at most hotels. In this scenario, a real-time trace would actually find the botmaster, since he would be sitting at the machine in question. However, even if the police are late by only several minutes, there might no longer be any record of who last used the computer. Physical evidence such as fingerprints, hair, and skin cells would be of little use, since many people use these computers each day. Unless a camera system is in place and it captured a clear picture of the suspect on his way to/from the computer, the police again will have no leads.

This section illustrates a few common scenarios where even a perfect IP traceback solution would fail to locate the botmaster. Clearly, much work remains on developing automated, integrated traceback solutions that work across various types of networks and protocols.

7. Summary

Botnets are one of the biggest threats to the Internet today, and they are linked to most forms of Internet crime. Most spam, DDoS attacks, spyware, click fraud, and other attacks originate from botnets and the shadowy organizations behind them. Running a botnet is immensely profitable, as several recent high-profile arrests have shown. Currently, many botnets still rely on a centralized IRC C&C structure, but more and more botmasters are using P2P protocols to provide resilience and avoid a single point of failure. A recent large-scale example of a P2P botnet is the Storm Worm, widely covered in the media.

A number of botnet countermeasures exist, but most are focused on bot detection and removal at the host and network level. Some approaches exist for Internet-wide detection and disruption of entire botnets, but we still lack effective techniques for combating the root of the problem: the botmasters who conceal their identities and locations behind chains of stepping-stone proxies.

The three biggest challenges in botmaster traceback are stepping stones, encryption, and the low traffic volume. Even if these problems can be solved with a technical solution, the trace must be able to continue beyond the reach of the Internet. Mobile phone networks, open wireless access points, and public computers all provide an additional layer of anonymity for the botmasters.

Short of a perfect solution, even a partial traceback technique could serve as a very effective deterrent for botmasters. With each botmaster that is located and arrested, many botnets will be eliminated at once. Additionally, other botmasters could decide that the risks outweigh the benefits when they see more and more of their colleagues getting caught. Currently, the economic equation is very simple: Botnets can generate large profits with relatively low risk of getting caught. A botmaster traceback solution, even if imperfect, would drastically change this equation and convince more botmasters that it simply is not worth the risk of spending the next 10–20 years in prison.

References

[1] Holz T. A short visit to the bot zoo. IEEE Security and Privacy 2005;3(3):76–9.
[2] Berinato S. Attack of the bots, WIRED. Issue 14.11, November 2006, www.wired.com/wired/archive/14.11/botnet.html.
[3] Evers J. 'Bot herders' may have controlled 1.5 million PCs. http://news.cnet.com/Bot-herders-may-have-controlled-1.5-million-PCs/2100-7350_3-5906896.html.
[4] Greenberg A. Spam crackdown 'a drop in the bucket'. Forbes June 14, 2007, www.forbes.com/security/2007/06/14/spam-arrest-fbi-tech-security-cx_ag_0614spam.html.
[5] Wikipedia contributors. Timeline of notable computer viruses and worms. http://en.wikipedia.org/w/index.php?title=Timeline_of_notable_computer_viruses_and_worms&oldid=207972502 (accessed May 3, 2008).

[6] Barford P, Yegneswaran V. “An inside look at botnets,” Special Workshop on Malware Detection, Advances in Information Security. Springer Verlag, 2006.
[7] Wikipedia contributors. Eggdrop. http://en.wikipedia.org/w/index.php?title=Eggdrop&oldid=207430332 (accessed May 3, 2008).
[8] Cooke E, Jahanian F, McPherson D. The zombie roundup: Understanding, detecting, and disturbing botnets, In: Proc. 1st Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI), Cambridge; July 7, 2005. p. 39–44.
[9] Ianelli N, Hackworth A. Botnets as a vehicle for online crime. In: Proc. 18th Annual Forum of Incident Response and Security Teams (FIRST), Baltimore; June 25–30, 2006.
[10] Rajab M, Zarfoss J, Monrose F, Terzis A. A multifaceted approach to understanding the botnet phenomenon. In: Proc. of the 6th ACM SIGCOM Internet Measurement Conference, Brazil: Rio de Janeiro; October 2006.
[11] Trend Micro. Taxonomy of botnet threats. Trend Micro Enterprise Security Library; November 2006.
[12] Symantec. Symantec internet security threat report, trends for July–December 2007. Volume XIII, April 2008.
[13] Grizzard J, Sharma V, Nunnery C, Kang B, Dagon D. Peer-to-peer botnets: Overview and case study. In: Proc. First Workshop on Hot Topics in Understanding Botnets (HotBots), Cambridge, April 2007.
[14] Stewart J. Bobax Trojan analysis, SecureWorks May 17, 2004, http://secureworks.com/research/threats/bobax.
[15] Chiang K, Lloyd L. A case study of the Rustock Rootkit and Spam Bot. In: Proc. First Workshop on Hot Topics in Understanding Botnets (HotBots), Cambridge, April 10, 2007.
[16] Lemos R. Bot software looks to improve peerage, SecurityFocus. May 2, 2006, www.securityfocus.com/news/11390/.
[17] Wang P, Sparks S, Zou C. An advanced hybrid peer-to-peer botnet. In: Proc. First Workshop on Hot Topics in Understanding Botnets (HotBots), Cambridge, April 10, 2007.
[18] Stewart J. Sinit P2P Trojan analysis. SecureWorks. December 8, 2004, www.secureworks.com/research/threats/sinit/.
[19] Schoof R, Koning R. Detecting peer-to-peer botnets. unpublished paper, University of Amsterdam, February 4, 2007, http://staff.science.uva.nl/~delaat/sne-2006-2007/p17/report.pdf.
[20] Wikipedia contributors. Storm worm. http://en.wikipedia.org/w/index.php?title=Storm_Worm&oldid=207916428 (accessed May 4, 2008).
[21] Bizeul D. Russian business network study. unpublished paper, November 20, 2007, www.bizeul.org/files/RBN_study.pdf.
[22] Cha AE. Internet dreams turn to crime, Washington Post May 18, 2003, www.washingtonpost.com/ac2/wp-dyn/A2619-2003May17.
[23] Koerner BI. From Russia with løpht, Legal Affairs May–June 2002, http://legalaffairs.org/issues/May-June-2002/feature_koerner_mayjun2002.msp.
[24] Delio M. Inside Russia’s hacking culture. WIRED. March 12, 2001, www.wired.com/culture/lifestyle/news/2001/03/42346.
[25] Wikipedia contributors. Russian business network. http://en.wikipedia.org/w/index.php?title=Russian_Business_Network&oldid=209665215 (accessed May 3, 2008).
[26] Tung L. Infamous Russian ISP behind Bank of India hack. ZDNet. September 4, 2007, http://news.zdnet.co.uk/security/0,1000000189,39289057,00.htm?r=2.
[27] Bächer P, Holz T, Kötter M, Wicherski G. Know your enemy: Tracking botnets. March 13, 2005, see www.honeynet.org/papers/bots/.
[28] Wikipedia contributors. E-mail spam, http://en.wikipedia.org/w/index.php?title=E-mail_spam&oldid=209902571 (accessed May 3, 2008).

[29] Wikipedia contributors. Pharming. http://en.wikipedia.org/w/index.php?title=Pharming&oldid=196469141 accessed May 3, 2008.

[30] Naraine R. Money bots: Hackers cash in on hijacked PCs. eWeek. September 8, 2006, www.eweek.com/article2/0,1759,2013924,00.asp.

[31] Roberts PF. DOJ indicts hacker for hospital botnet attack. eWeek. February 10, 2006, www.eweek.com/article2/0,1759,1925456,00.asp.

[32] Claburn T. New Zealander 'AKILL' pleads guilty to botnet charges. Information Week April 3, 2008, www.informationweek.com/news/security/cybercrime/showArticle.jhtml?articleID=207001573.

[33] Wikipedia contributors. Agobot (computer worm). http://en.wikipedia.org/w/index.php?title=Agobot_%28computer_worm%29&oldid=201957526 (accessed May 3, 2008).

[34] Stinson E, Mitchell J. Characterizing bots' remote control behavior. In: Proc. 4th International Conference on Detection of Intrusions & Malware and Vulnerability Assessment (DIMVA), Lucerne, Switzerland, July 12–13, 2007.

[35] Goebel J, Holz T. Rishi: Identify bot contaminated hosts by IRC nickname evaluation. In: Proc. First Workshop on Hot Topics in Understanding Botnets (HotBots), Cambridge, April 10, 2007.

[36] Binkley J, Singh S. An algorithm for anomaly-based botnet detection, In: Proc. 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI), San Jose, July 7, 2006. p. 43–8.

[37] Gu G, Porras P, Yegneswaran V, Fong M, Lee W. BotHunter: Detecting malware infection through IDS-driven dialog correlation. In: Proc. 16th USENIX Security Symposium, Boston; August, 2007.

[38] Karasaridis A, Rexroad B, Hoeflin D. Wide-scale botnet detection and characterization, In: Proc. First Workshop on Hot Topics in Understanding Botnets (HotBots), Cambridge, MA; April 10, 2007.

[39] Ramachandran A, Feamster N, Dagon D. Revealing botnet membership using DNSBL counter-intelligence, In: Proc. 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI), San Jose, CA; July 7, 2006. p. 49–54.

[40] Gu G, Zhang J, Lee W. BotSniffer: Detecting botnet command and control channels in network traffic, In: Proc. 15th Network and Distributed System Security Symposium (NDSS), San Diego, February 2008.

[41] Freiling F, Holz T, Wicherski G. Botnet tracking: Exploring a root-cause methodology to prevent denial-of-service attacks. In: Proc. 10th European Symposium on Research in Computer Security (ESORICS), Milan, Italy, September 12–14, 2005.

[42] Dagon D, Zou C, Lee W. Modeling botnet propagation using time zones, In: Proc. 13th Network and Distributed System Security Symposium (NDSS), February 2006.

[43] DeleGate multi-purpose application gateway. www.delegate.org/delegate/ (accessed May 4, 2008).

[44] Naraine R. Is the botnet battle already lost? eWeek. October 16, 2006, www.eweek.com/article2/0,1895,2029720,00.asp.

[45] Roberts PF. California man charged with botnet offenses. eWeek. November 3, 2005, www.eweek.com/article2/0,1759,1881621,00.asp.

[46] Roberts PF. Botnet operator pleads guilty. eWeek. January 24, 2006, www.eweek.com/article2/0,1759,1914833,00.asp.

[47] Nichols S. FBI 'bot roast' scores string of arrests. vnunet.com. December 3, 2007, www.vnunet.com/vnunet/news/2204829/bot-roast-scores-string-arrests.

[48] Savage S, Wetherall D, Karlin A, Anderson T. Practical network support for IP traceback, In: Proc. ACM SIGCOMM 2000, Sept. 2000. p. 295–306.

[49] Goodrich MT. Efficient packet marking for large-scale IP traceback, In: Proc. 9th ACM Conference on Computer and Communications Security (CCS 2002), October 2002. p. 117–26.

[50] Snoeren A, Patridge C, Sanchez LA, Jones CE, Tchakountio F, Kent ST, et al. Hash-based IP traceback. In: Proc. ACM SIGCOMM 2001, September 2001. p. 3–14.

[51] Li J, Sung M, Xu J, Li L. Large-scale IP traceback in high-speed internet: Practical techniques and theoretical foundation, In: Proc. 2004 IEEE Symposium on Security and Privacy, IEEE, 2004.
[52] Wang X, Chen S, Jajodia S. Network flow watermarking attack on low-latency anonymous communication systems. In: Proc. 2007 May IEEE Symposium on Security and Privacy; 2007.
[53] Wang X, Chen S, Jajodia S. Tracking anonymous, peer-to-peer VoIP calls on the internet. In: Proc. 12th ACM Conference on Computer and Communications Security (CCS 2005), October 2005.
[54] Wang X, Reeves D. Robust correlation of encrypted attack traffic through stepping stones by manipulation of interpacket delays. Proc. 10th ACM Conference on Computer and Communications Security (CCS 2003), October 2003. p. 20–9.
[55] WLIrc wireless IRC client for mobile phones. http://wirelessirc.sourceforge.net/ (accessed May 3, 2008).
[56] jmIrc: Java mobile IRC-client (J2ME). http://jmirc.sourceforge.net/ (accessed May 3, 2008).

CHAPTER 9

Intranet Security

Bill Mansoor
Information Systems Audit and Control Association (ISACA)

Intranet Security as News in the Media

- "State Department Contract Employees Fired, Another Disciplined for Looking at Passport File" [1]
- "Laptop stolen with a million customer data records" [2]
- "eBayed VPN kit hands over access to council network" [3]
- "(Employee) caught selling personal and medical information about . . . FBI agent to a confidential source . . . for $500" [4].
- "Data thieves gain access to TJX through unsecured wireless access point" [5]

Headline dramas like these in the mainstream media are embarrassing nightmares to top brass in any large corporation. These events have a lasting impact on a company's bottom line because the company reputation and customer trust take a direct hit. Once events like these transpire, customers and current and potential investors never look at the company in the same trusting light again, regardless of remediation measures. The smart thing, then, is to avoid this kind of limelight. The onus of preventing such embarrassing security gaffes falls squarely on the shoulders of IT security chiefs (CISOs and security officers), who are sometimes hobbled by unclear mandates from government regulators and lack of sufficient budgeting to tackle the mandates.

However, federal governments across the world are not taking breaches of personal data lightly (see side bar, "TJX: Data Breach with 45 Million Data Records Stolen"). In view of a massive plague of publicized data thefts in the past decade, recent mandates such as the Health Insurance Portability and Accountability Act (HIPAA), Sarbanes–Oxley, and the Payment Card Industry-Data Security Standard (PCI-DSS) Act within the United States now have teeth. These go so far as to spell out stiff fines and personal jail sentences for CEOs who neglect data breach issues.

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00009-9

TJX: Data Breach with 45 Million Data Records Stolen

The largest-scale data breach in history occurred in early 2007 at TJX, the parent company for the TJ Maxx, Marshalls, and HomeGoods retail chains.

In the largest identity-theft case ever investigated by the U.S. Department of Justice, 11 people were convicted of wire fraud in the case. The primary suspect was found to perpetrate the intrusion by wardriving and taking advantage of an unsecured Wi-Fi access point to get in and set up a "sniffer" software instance to capture credit-card information from a database [12].

Though the intrusion was earlier believed to have taken place from May 2006 to January 2007, TJX later found that it took place as early as July 2005. The data compromised included portions of the credit- and debit-card transactions for approximately 45 million customers [6].

As seen in the TJX case, intranet data breaches can be a serious issue, impacting a company's goodwill in the open marketplace as well as spawning class-action lawsuits [7].

Gone are the days when intranet security was a superficial exercise; security inside the firewall was all but nonexistent. There was a feeling of implicit trust in the internal user. After all, if you hired that person, trained him for years, how could you not trust him?

In the new millennium, the Internet has come of age, and so have its users. The last largely computer-agnostic generation has exited the user scene; their occupational shoes have been filled with the "X and Y" generations. Many of these young people have grown up with the Internet, often familiar with it since elementary school. It is not uncommon today to find young college students who started their programming interests in the fifth or sixth grade.

With such a level of computer-savvy in users, the game of intranet security has changed (see side bar, "Network Breach Readiness: Many Are Still Complacent"). Resourceful as ever, these new users have gotten used to the idea of being hyperconnected to the Internet using mobile technology such as personal digital assistants (PDAs) and smart phones and firewalled barriers. For a corporate intranet that uses older ideas of using access control as the cornerstone of data security, such mobile access to the Internet at work needs careful analysis and control. The idea of building a virtual moat around your well-constructed castle (investing in a firewall and hoping to call it an intranet) is gone. Hyperconnected "knowledge workers" with laptops, PDAs and USB keys that have whole operating systems built in have made sure of it.

Network Breach Readiness: Many Are Still Complacent

The level of readiness for breaches among IT shops across the country is still far from optimal. The Ponemon Institute, a security think tank, surveyed some industry personnel and came up with some startling revelations. It is hoped that these will change in the future:

- Eighty-five percent of industry respondents reported that they had experienced a data breach.
- Of those responding, 43% had no incident response plan in place, and 82% did not consult legal counsel before responding to the incident.
- Following a breach, 46% of respondents still had not implemented encryption on portable devices (laptops, PDAs) with company data stored on them [8].

If we could reuse the familiar vehicle ad tagline of the 1980s, we would say that the new intranet is not "your father's intranet anymore." The intranet as just a simple place to share files and to list a few policies and procedures has ceased to be. The types of changes can be summed up in the following list of features, which shows that the intranet has become a combined portal as well as a public dashboard. Some of the features can include:

- A searchable corporate personnel directory of phone numbers by department. Often the list is searchable only if the exact name is known.
- Expanded activity guides and a corporate calendar with links for various company divisions.
- Several RSS feeds for news according to divisions such as IT, HR, Finance, Accounting, and Purchasing.
- Company blogs (weblogs) by top brass that talk about the current direction for the company in reaction to recent events, a sort of "mission statement of the month."
- Intranets frequently feature a search engine for searching company information, often helped by a search appliance from Google. Microsoft also has its own search software on offer that targets corporate intranets.
- One or several "wiki" repositories for company intellectual property, some of it of a mission-critical nature. Usually granular permissions are applied for access here. One example could be court documents for a legal firm with rigorous security access applied.
- A section describing company financials and other mission-critical indicators. This is often a separate Web page linked to the main intranet page.
- A "live" section with IT alerts regarding specific downtimes, outages, and other critical time-sensitive company notifications. Often embedded within the portal, this is displayed in a "ticker-tape" fashion or like an RSS-type dynamic display.

Of course, this list is not exhaustive; some intranets have other unique features not listed here. But in any case, intranets these days do a lot more than simply list corporate phone numbers.

Recently, knowledge management systems have presented another challenge to intranet security postures. Companies that count knowledge as a prime protected asset (virtually all companies these days) have started deploying “mashable” applications that combine social networking (such as Facebook and LinkedIn), texting, and microblogging (such as Twitter) features to encourage employees to “wikify” their knowledge and information within intranets. One of the bigger vendors in this space, Socialtext, has introduced a mashable wiki app that operates like a corporate dashboard for intranets [9, 10].

Socialtext has individual widgets, one of which, “Socialtext signals,” is a microblogging engine. In the corporate context, microblogging entails sending short SMS messages to apprise colleagues of recent developments in the daily routine. Examples could be short messages on progress on any major project milestone—for example, joining up major airplane assemblies or getting Food and Drug Administration testing approval for a special experimental drug.

These emerging scenarios present special challenges to security personnel guarding the borders of an intranet. The border as it once existed has ceased to be. One cannot block stored knowledge from leaving the intranet when a majority of corporate mobile users are accessing intranet wikis from anywhere using inexpensive mini-notebooks that are given away with cell phone contracts [11].

If we consider the impact of national and international privacy mandates on these situations, the situation is compounded further for C-level executives in multinational companies who have to come up with responses to privacy mandates in each country in which the company does business. The privacy mandates regarding private customer data have always been more stringent in Europe than in North America, which is a consideration for doing business in Europe.

It is hard enough to block entertainment-related Flash video traffic from time-wasting Internet abuse without blocking a video of last week’s corporate meeting at headquarters. Only letting in traffic on an exception basis becomes untenable or impractical because of a high level of personnel involvement needed for every ongoing security change. Simply blocking YouTube.com or Vimeo.com is not sufficient. Video, which has myriad legitimate work uses nowadays, is hosted on all sorts of content-serving (caching and streaming) sites worldwide, which makes it well near impossible to block using Web filters.
The evolution of the Internet Content Adaptation Protocol (ICAP), which standardizes Web site categories for content-filtering purposes, is under way. However, ICAP still does not solve the problem of the dissolving networking “periphery [12].”

Guarding movable and dynamic data—which may be moving in and out of the perimeter without notice, flouting every possible mandate—is a key feature of today’s intranet.
The dynamic nature of data has rendered the traditional confidentiality, integrity, and availability (CIA) architecture somewhat less relevant. The changing nature of data security necessitates some specialized security considerations.

- Intranet security policies and procedures (P&Ps) are the first step toward a legal regulatory framework. The P&Ps needed on any of the security controls listed below should be compliant with federal and state mandates (such as HIPAA, Sarbanes-Oxley, the European Directive 95/46/EC on the protection of personal data, and PCI-DSS, among others). These P&Ps have to be signed off by top management and placed on the intranet for review by employees. There should be sufficient teeth in all procedural sections to enforce the policy, explicitly spelling out sanctions and other consequences of noncompliance, leading up to discharge.
- To be factual, none of these government mandates spell out details on implementing any security controls. That is the vague nature of federal and international mandates. Interpretation of the security controls is better left after the fact to an entity such as the National Institute of Standards and Technology (NIST) in the United States or the Geneva-based International Organization for Standardization (ISO). These organizations have extensive research and publication guidance for any specific security initiative. Most of NIST's documents are offered as free downloads from its Web site [13]. ISO security standards such as 27002~27005 are also available for a nominal fee from the ISO site.

Policies and procedures, once finalized, need to be automated as much as possible (one example is mandatory password changes every three months). Automating policy compliance takes the error-prone human factor out of the equation (see side bar, "Access Control in the Era of Social Networking"). There are numerous software tools available to help accomplish security policy automation.

Access Control in the Era of Social Networking

In an age in which younger users have grown up with social networking sites as part of their digital lives, corporate intranet sites are finding it increasingly difficult to block them from using these sites at work. Depending on the company, some are embracing social networking as part of their corporate culture; others, especially government entities, are actively blocking these sites. Detractors mention as concerns wasted bandwidth, lost productivity, and the possibility of infections with spyware and worms.

However, blocking these sites can be difficult because most social networking and video sites such as Vimeo and YouTube can use port 80 to vector Flash videos into an intranet—which is wide open for HTTP access. Flash videos have the potential to provide a convenient Trojan horse for malware to get into the intranet.

To block social networking sites, one needs to block either the Social Networking category or block the specific URLs (such as YouTube.com) for these sites in the Web-filtering proxy appliance. Flash videos are rarely downloaded from YouTube itself. More often a redirected caching site is used to send in the video. The caching sites also need to be blocked; this is categorized under Content Servers.

1. Plugging the Gaps: Network Access Control and Access Control

The first priority of an information security officer in most organizations is to ensure that there is a relevant corporate policy on access controls. Simple on the surface, the subject of access control is often complicated by the variety of ways the intranet is connected to the external world.

Remote users coming in through traditional or SSL (browser-based) virtual private networks (VPNs), control over use of USB keys, printouts, and CD-ROMs all require that a comprehensive endpoint security solution be implemented.

The past couple of years have seen large-scale adoption of network access control (NAC) products in the mid-level and larger IT shops to manage endpoint security. Endpoint security ensures that whomever is plugging into or accessing any hardware anywhere within the intranet has to comply with the minimum baseline corporate security policy standards. This can include add-on access credentials but goes far beyond access. Often these solutions ensure that traveling corporate laptops are compliant with a minimum patching level, scans, and antivirus definition levels before being allowed to connect to the intranet.

The NAC appliances that enforce these policies often require that a NAC fat client is installed on every PC and laptop. This rule can be enforced during logon using a logon script. The client can also be a part of the standard OS image for deploying new PCs and laptops.

Microsoft has built a NAC-type framework into some versions of its client OSs (Vista and XP SP3) to ease compliance with its NAC server product called MS Network Policy Server, which closely works with its Windows 2008 Server product (see side bar, "The Cost of a Data Breach"). The company has been able to convince quite a few industry networking heavyweights (notably Cisco and Juniper) to adopt its NAP standard [14].

The Cost of a Data Breach

- As of July 2007, the average breach cost per incident was $4.8 million.
- This works out to $182 per exposed record.
- It represents an increase of more than 30% from 2005.
- Thirty-five percent of these breaches involved the loss or theft of a laptop or other portable device.
- Seventy percent were due to a mistake or malicious intent by an organization's own staff.
- Since 2005 almost 150 million individuals' identifiable information has been compromised due to a data security breach.
- Nineteen percent of consumers notified of a data breach discontinued their relationship with the business, and a further 40% considered doing so [15].

Essentially the technology has three parts: a policy-enforceable client, a decision point, and an enforcement point. The client could be an XP SP3 or Vista client (either a roaming user or guest user) trying to connect to the company intranet. The decision point in this case would be the Network Policy Server product, checking to see whether the client requesting access meets the minimum baseline to allow it to connect. If it does not, the decision point product would pass this data on to the enforcement point, a network access product such as a router or switch, which would then be able to cut off access.

The scenario would repeat itself at every connection attempt, allowing the network's health to be maintained on an ongoing basis. Microsoft's NAP page has more details and animation to explain this process [16].

Access control in general terms is a relationship triad among internal users, intranet resources, and the actions internal users can take on those resources. The idea is to give users only the least amount of access they require to perform their job. The tools used to ensure this in Windows shops utilize Active Directory for Windows logon scripting and Windows user profiles. Granular classification is needed for users, actions, and resources to form a logical and comprehensive access control policy that addresses who gets to connect to what, yet keeping the intranet safe from unauthorized access or data-security breaches. Quite a few off-the-shelf solutions geared toward this market often combine inventory control and access control under a "desktop life-cycle" planning umbrella.

Typically, security administrators start with a "Deny-All" policy as a baseline before slowly building in the access permissions. As users migrate from one department to another, are promoted, or leave the company, in large organizations this job can involve one person by herself. This person often has a very close working relationship with Purchasing, Helpdesk, and HR, getting coordination and information from these departments on users who have separated from the organization and computers that have been surplused, deleting and modifying user accounts and assignments of PCs and laptops.

Helpdesk software usually has an inventory control component that is readily available to Helpdesk personnel to update and/or pull up to access details on computer assignments and user status. Optimal use of form automation can ensure that these details occur (such as deleting a user on the day of separation) to avoid any possibility of an unwelcome data breach.

2. Measuring Risk: Audits

Audits are another cornerstone of a comprehensive intranet security policy. To start an audit, an administrator should know and list what he is protecting as well as knowing the relevant threats and vulnerabilities to those resources.

Assets that need protection can be classified as either tangible or intangible. *Tangible assets* are, of course, removable media (USB keys), PCs, laptops, PDAs, Web servers, networking equipment, DVR security cameras, and employees' physical access cards. *Intangible assets* can include company intellectual property such as corporate email and wikis, user passwords, and, especially for HIPAA and Sarbanes-Oxley mandates, personally identifiable health and financial information, which the company could be legally liable to protect.

Threats can include theft of USB keys, laptops, PDAs, and PCs from company premises, resulting in a data breach (for *tangible assets*) and weak passwords and unhardened operating systems in servers (for *intangible assets*).

Once a correlated listing of assets and associated threats and vulnerabilities has been made we have to measure the impact of a breach, which is known as *risk*. The common rule of thumb to measure risk is:

$$Risk = Value\ of\ asset \times Threat \times Vulnerability$$

It is obvious that an Internet-facing Web server faces greater risk and requires priority patching and virus scanning because the vulnerability and threat components are high in that case (these servers routinely get sniffed and scanned over the Internet by hackers looking to find holes in their armor). However, this formula can standardize the priority list so that the actual audit procedure (typically carried out weekly or monthly by a vulnerability-scanning device) is standardized by risk level. Vulnerability-scanning appliances usually scan server farms and networking appliances only because these are high-value targets within the network for hackers who are looking for either unhardened server configurations or network switches with default factory passwords left on by mistake. To illustrate the situation, look at Figure 9.1, which illustrates an SQL injection attack on a corporate database [17].

The value of an asset is subjective and can be assessed only by the IT personnel in that organization (see side bar, "Questions for a Nontechnical Audit of Intranet Security"). If the IT staff has an ITIL (Information Technology Infrastructure Library) process under way, the value of an asset will often already have been classified and can be used. Otherwise, a small spreadsheet can be created with classes of various tangible and intangible assets (as part of a hardware/software cataloguing exercise) and values assigned that way.

Questions for a Nontechnical Audit of Intranet Security

- Is all access (especially to high-value assets) logged?
- In case of laptop theft, is encryption enabled so that the records will be useless to the thief?
- Are passwords verifiably strong enough to comply with the security policy? Are they changed frequently and held to strong encryption standards?
- Are all tangible assets (PCs, laptops, PDAs, Web servers, networking equipment) tagged with asset tags?
- Is the process for surplusing obsolete IT assets secure (meaning, are disks wiped for personally identifiable data before surplusing happens)?
- Is email and Web usage logged?
- Are peer-to-peer (P2P) and instant messaging (IM) usage controlled?

Based on the answers you get (or don't), you can start the security audit procedure by finding answers to these questions.

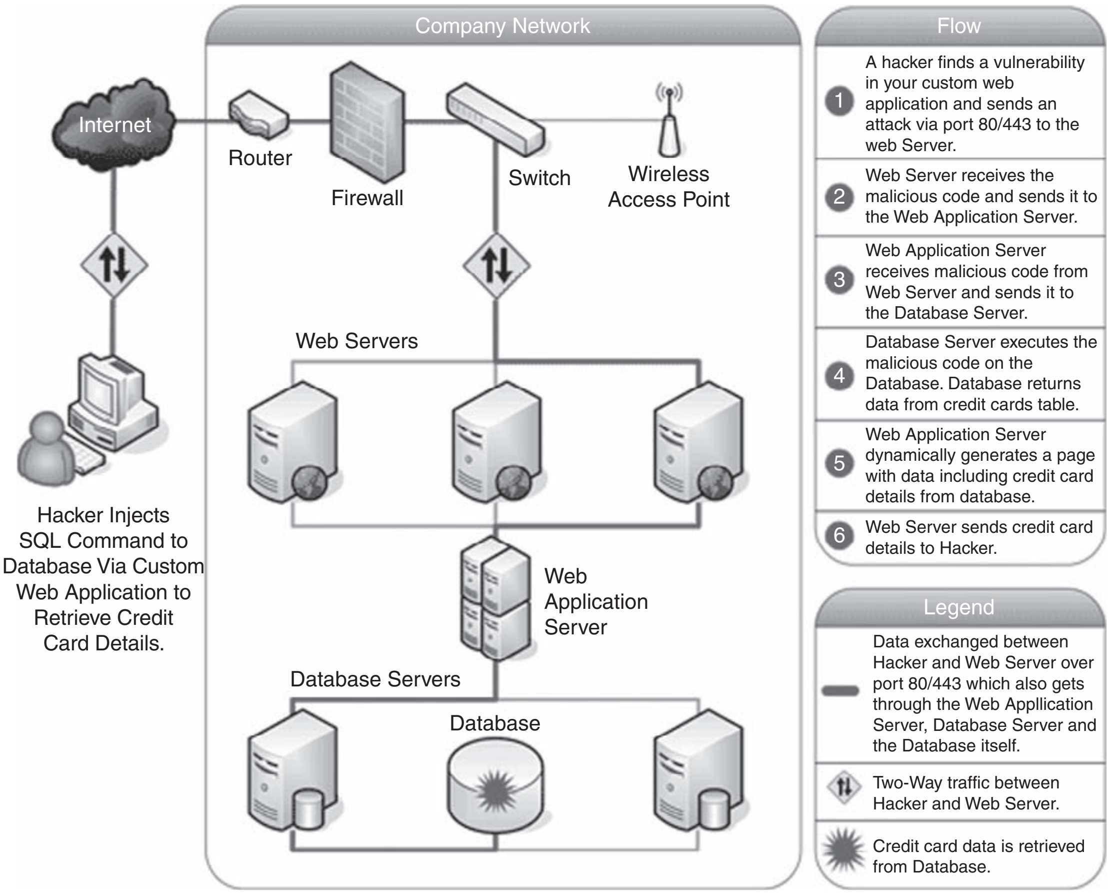

Company Network
Internet
Router
Firewall
Switch
Wireless Access Point
Web Servers
Web Application Server
Database Servers
Database
Hacker Injects SQL Command to Database Via Custom Web Application to Retrieve Credit Card Details.
Flow
1 A hacker finds a vulnerability in your custom web application and sends an attack via port 80/443 to the web Server.
2 Web Server receives the malicious code and sends it to the Web Application Server.
3 Web Application Server receives malicious code from Web Server and sends it to the Database Server.
4 Database Server executes the malicious code on the Database. Database returns data from credit cards table.
5 Web Application Server dynamically generates a page with data including credit card details from database.
6 Web Server sends credit card details to Hacker.
Legend
Data exchanged between Hacker and Web Server over port 80/443 which also gets through the Web Appllication Server, Database Server and the Database itself.
Two-Way traffic between Hacker and Web Server.
Credit card data is retrieved from Database.

Figure 9.1: SQL injection attack.
Source: © acunetix.com.

3. Guardian at the Gate: Authentication and Encryption

To most lay users, authentication in its most basic form is two-factor authentication—meaning a username and a password. Although adding further factors (such as additional autogenerated personal identification numbers [PINs] and/or biometrics) makes authentication stronger by magnitudes, one can do a lot with just the password within a two-factor situation. Password strength is determined by how hard the password is to crack using a password-cracker application that uses repetitive tries using common words (sometimes from a stored dictionary) to match the password. Some factors will prevent the password from being cracked easily and make it a stronger password:

- Password length (more than eight characters)
- Use of mixed case (both uppercase and lowercase)

- Use of alphanumeric characters (letters as well as numbers)
- Use of special characters (such as !, ?, %, and #)

The ACL in a Windows AD environment can be customized to demand up to all four factors in the setting or renewal of a password, which will render the password strong.

Prior to a few years ago, the complexity of a password (the last three items in the preceding list) was favored as a measure of strength in passwords. However, the latest preference as of this writing is to use uncommon passwords—joined-together sentences to form passphrases that are quite long but don't have much in the way of complexity. Password authentication ("what you know") as two-factor authentication is not as secure as adding a third factor to the equation (a dynamic token password). Common types of third-factor authentication include biometrics (fingerprint scan, palm scan, or retina scan—in other words, "what you are") and token-type authentication (software or hardware PIN–generating tokens—that is, "what you have").

Proximity or magnetic swipe cards and tokens have seen common use for physical premises-access authentication in high-security buildings (such as financial and R&D companies) but not for network or hardware access within IT.

When remote or teleworker employees connect to the intranet via VPN tunnels or Web-based SSL VPNs (the outward extension of the intranet once called an *extranet*), the connection needs to be encrypted with strong 3DES or AES type encryption to comply with patient data and financial data privacy mandates. The standard authentication setup is usually a username and a password, with an additional hardware token-generated random PIN entered into a third box. Until lately, RSA as a company was one of the bigger players in the hardware-token field; it incidentally also invented the RSA algorithm for public-key encryption.

As of this writing, hardware tokens cost under $30 per user in quantities of greater than a couple hundred pieces, compared to about a $100 only a decade ago. Most vendors offer free lifetime replacements for hardware tokens. Instead of a separate hardware token, some inexpensive software token generators can be installed within PC clients, smart phones, and BlackBerry devices. Tokens are probably the most cost-effective enhancement to security today.

4. Wireless Network Security

Employees using the convenience of wireless to log into the corporate network (usually via laptop) need to have their laptops configured with strong encryption to prevent data breaches. The first-generation encryption type known as Wireless Equivalent Privacy (WEP) was easily

deciphered (cracked) using common hacking tools and is no longer widely used.
The latest standard in wireless authentication is WPA or WPA2 (802.11i), which offer stronger encryption compared to WEP. Though wireless cards in laptops can offer all the previously noted choices, they should be configured with WPA or WPA2 if possible.

There are quite a few hobbyists roaming corporate areas looking for open wireless access points (transmitters) equipped with powerful Wi-Fi antennas and wardriving software, a common package being Netstumbler. Wardriving was originally meant to log the presence of open Wi-Fi access points on Web sites (see side bar, "Basic Ways to Prevent Wi-Fi Intrusions in Corporate Intranets"), but there is no guarantee that actual access and use (*piggybacking*, in hacker terms) won't occur, curiosity being human nature. If there is a profit motive, as in the TJX example, access to corporate networks will take place, although the risk of getting caught and resulting risk of criminal prosecution will be high. Furthermore, installing a RADIUS server is a must to check access authentication for roaming laptops.

Basic Ways to Prevent Wi-Fi Intrusions in Corporate Intranets

1. Reset and customize the default Service Set Identifier (SSID) or Extended Service Set Identifier (ESSID) for the access point device before installation.
2. Change the default admin password.
3. Install a RADIUS server, which checks for laptop user credentials from an Active Directory database (ACL) from the same network before giving access to the wireless laptop. See Figures 9.2 and 9.3 for illustrated explanations of the process.

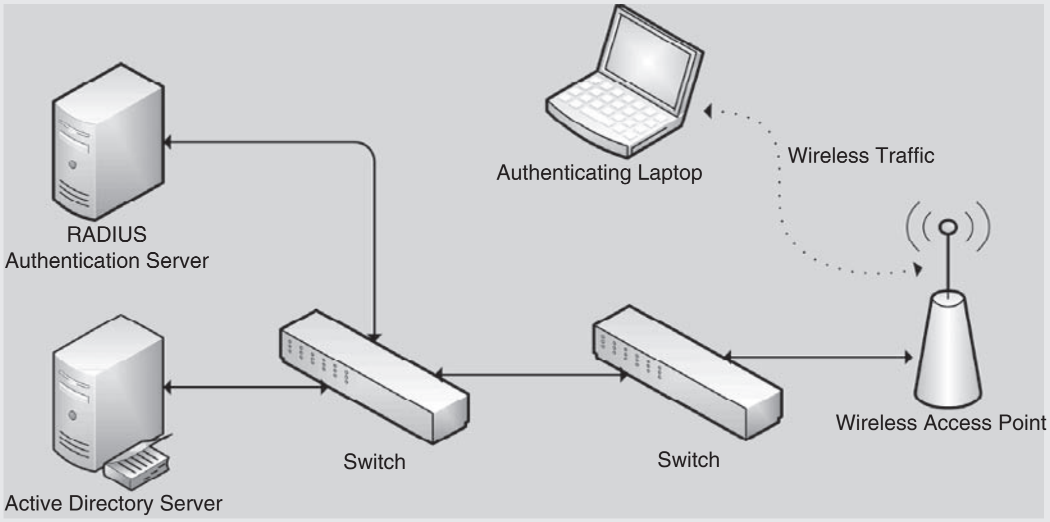


Figure 9.2: Wireless EAP authentication using Active Directory and authentication servers.

Continued

Basic Ways to Prevent Wi-Fi Intrusions in Corporate Intranets—Cont'd

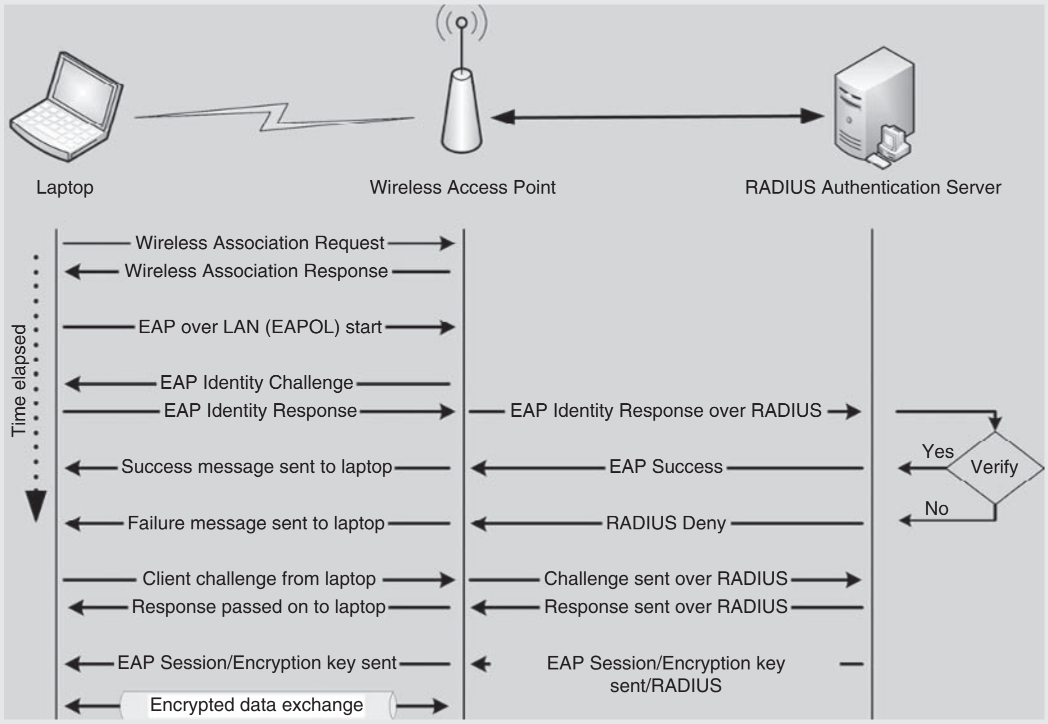

Laptop
Wireless Access Point
RADIUS Authentication Server
Time elapsed
Wireless Association Request
Wireless Association Response
EAP over LAN (EAPOL) start
EAP Identity Challenge
EAP Identity Response
EAP Identity Response over RADIUS
Yes
Verify
No
Success message sent to laptop
EAP Success
Failure message sent to laptop
RADIUS Deny
Client challenge from laptop
Challenge sent over RADIUS
Response passed on to laptop
Response sent over RADIUS
EAP Session/Encryption key sent
EAP Session/Encryption key sent/RADIUS
Encrypted data exchange

Figure 9.3: High-level wireless Extensible Authentication Protocol (EAP) workflow.

4. Enable WPA or WPA2 encryption, not WEP, which is easily cracked.
5. Periodically try to wardrive around your campus and try to sniff (and disable) nonsecured network-connected rogue access points set up by naïve users.
6. Document the wireless network by using one of the leading wireless network management software packages made for that purpose.

Note: Contrary to common belief, turning off SSID broadcast won't help unless you're talking about a home access point situation. Hackers have an extensive suite of tools with which to sniff SSIDs for lucrative corporate targets, which will be broadcast anyway when connecting in clear text (unlike the real traffic, which will be encrypted).

5. Shielding the Wire: Network Protection

Firewalls are, of course, the primary barrier to a network. Typically rule based, firewalls prevent unwarranted traffic from getting into the intranet from the Internet. These days, firewalls also do some stateful inspections within packets to peer a little into the header contents of an incoming packet, to check validity—that is, to check whether a streaming video packet is really what it says it is, and not malware masquerading as streaming video.

Intrusion prevention systems (IPSs) are a newer type of inline network appliance that uses heuristic analysis (based on a weekly updated signature engine) to find patterns of malware identity and behavior and to block malware from entering the periphery of the intranet. The IPS and the intrusion detection system (IDS), however, operate differently.

IDSs are typically *not* sitting inline; they sniff traffic occurring anywhere in the network, cache extensively, and can correlate events to find malware. The downside of IDSs is that unless their filters are extensively modified, they generate copious amounts of false positives—so much so that "real" threats become impossible to sift out of all the noise.

IPSs, in contrast, work *inline* and inspect packets rapidly to match packet signatures. The packets pass through many hundreds of parallel filters, each containing matching rules for a different type of malware threat. Most vendors publish new sets of malware signatures for their appliances every week. However, signatures for common worms and injection exploits such as SQL-slammer, Code-red, and NIMDA are sometimes hardcoded into the application-specific integrated chip (ASIC) that controls the processing for the filters. Hardware-enhancing a filter helps avert massive-scale attacks more efficiently because it is performed in hardware, which is more rapid and efficient compared to software signature matching. Incredible numbers of malicious packets can be dropped from the wire using the former method.

The buffers in an enterprise-class IPS are smaller compared to those in IDSs and are quite fast—akin to a high-speed switch to preclude latency (often as low as 200 microseconds during the highest load). A top-of-the-line midsize IPS box's total processing threshold for all input and output segments can exceed 5 gigabits per second using parallel processing [18].

However, to avoid overtaxing CPUs and for efficiency's sake, IPSs usually block only a very limited number of important threats out of the thousands of malware signatures listed. Tuning IPSs can be tricky—just enough blocking to silence the false positive noise but making sure all critical filters are activated to block important threats.

The most important factors in designing a critical data infrastructure are resiliency, robustness, and redundancy regarding the operation of inline appliances. Whether one is talking about firewalls or inline IPSs, redundancy is paramount (see side bar, "Types of Redundancy for Inline Security Appliances"). Intranet robustness is a primary concern where data has to available on a 24/7 basis.

Types of Redundancy for Inline Security Appliances

1. Security appliances usually have dual power supplies (often hot-swappable) and are designed to be connected to two separate UPS devices, thereby minimizing chances of a failure within the appliance itself. The hot-swap capability minimizes replacement time for power supplies.
2. We can configure most of these appliances to either shut down the connection or fall back to a level-two switch (in case of hardware failure). If reverting to a fallback state, most IPSs become basically a bump in the wire and, depending on the type of traffic, can be configured to fail open so that traffic remains uninterrupted. Also, inexpensive, small third-party switchboxes are available to enable this failsafe high-availability option for a single IPS box. The idea is to keep traffic flow active regardless of attacks.
3. IPS or firewall devices can be placed in dual-redundant failover mode, either in active-active (load-sharing) or active-passive (primary-secondary) mode. The devices commonly use a protocol called Virtual Router Redundancy Protocol (VRRP) where the secondary pings the primary every second to check live status and assumes leadership to start processing traffic in case pings are not returned from the primary. The switchover is instantaneous and transparent to most network users. Prior to the switchover, all data and connection settings are fully synchronized at identical states between both boxes to ensure failsafe switchover.
4. Inline IPS appliances are relatively immune to attacks because they have highly hardened Linus/Unix operating systems and are designed from the ground up to be robust and low-maintenance appliances (logs usually clear themselves by default).

Note: Contrary to common belief, turning off SSID broadcast won't help unless you're talking about a home access point situation. Hackers have an extensive suite of tools with which to sniff SSIDs for lucrative corporate targets, which will be broadcast anyway when connecting in clear text (unlike the real traffic, which will be encrypted).

Most security appliances come with syslog reporting (event and alert logs sent usually via port 514 UDP) and email notification (set to alert beyond a customizable threshold) as standard. The syslog reporting can be forwarded to a security events management (SEM) appliance, which consolidates syslogs into a central threat console for benefit of event correlation and forwards warning emails to administrators based on preset threshold criteria. Moreover, most firewalls and IPSs can be configured to forward their own notification email to administrators in case of an impending threat scenario.

For those special circumstances where a wireless-type LAN connection is the primary one (whether microwave beam, laser beam, or satellite-type connection), redundancy can be ensured by a secondary connection of equal or smaller capacity. For example, in certain northern Alaska towns where digging trenches into the hardened icy permafrost is expensive and rigging wire across the tundra is impractical due to the extreme cold, the primary network connections between towns are always via microwave link, often operating in dual redundant mode.

6. Weakest Link in Security: User Training

Intranet security awareness is best communicated to users in two primary ways—during new employee orientation and by ongoing targeted training for users in various departments, with specific user audiences in mind.

A formal security training policy should be drafted and signed off by management, with well-defined scopes, roles, and responsibilities of various individuals, such as the CIO and the information security officer, and posted on the intranet. New recruits should be given a copy of all security policies to sign off on before they are granted user access. The training policy should also spell out the HR, Compliance, and PR departments' roles in the training program.

Training can be given using the PowerPoint Seminar method in large gatherings before monthly "all-hands" departmental meetings and also via an emailed Web link to a Flash video format presentation. The latter can also be configured to have an interactive quiz at the end, which should pique audience interest on the subject and help them remember relevant issues.

As far as topics to be included in the training, any applicable federal or industry mandate such as HIPAA, SOX, PCI-DSS, or ISO 27002 should be discussed extensively first, followed by discussions on tackling social engineering, spyware, viruses, and so on.

The topics of data theft and corporate data breaches are frequently in the news. This subject can be extensively discussed with emphasis on how to protect personally identifiable information in a corporate setting. Password policy and access control topics are always good things to discuss; users at a minimum need to be reminded to sign off their workstations before going on break.

7. Documenting the Network: Change Management

Controlling the IT infrastructure configuration of a large organization is more about change control than other things. Often the change control guidance comes from documents such as the ITIL series of guidebooks.

After a baseline configuration is documented, change control—a deliberate and methodical process that ensures that any changes made to the baseline IT configuration of the organization (such as changes to network design, AD design, and so on)—is extensively documented and authorized only after prior approval. This is done to ensure that unannounced or unplanned changes are not allowed to hamper the day-to-day efficiency and business functions of the overall intranet infrastructure.

In most government entities, even very small changes are made to go through change management (CM); however, management can provide leeway to managers to approve a certain minimal level of ad hoc change that has no potential to disrupt operations. In most organizations where mandates are a day-to-day affair, no ad hoc change is allowed unless it goes through supervisory-level change management meetings.

The goal of change management is largely to comply with mandates—but for some organizations, waiting for a weekly meeting can slow things significantly. If justified, an emergency CM meeting can be called to approve a time-sensitive change.

Practically speaking, the change management process works like this: A formal change management document is filled out (usually a multitab online Excel spreadsheet) and forwarded to the change management ombudsman (maybe a project management person). See the side bar "Change Management Spreadsheet Details to Submit to a CM Meeting" for some CM form details.

The document must have supervisory approval from the requestor's supervisor before proceeding to the ombudsman. The ombudsman posts this change document on a section of the intranet for all other supervisors and managers within the CM committee to review in advance. Done this way, the change management committee, meeting in its weekly or biweekly change approval meetings, can voice reservations or ask clarification questions of the change-initiating person, who is usually present to explain the change. At the end of the deliberations the decision is then voted on to approve, deny, modify, or delay the change (sometimes with preconditions).

Change Management Spreadsheet Details to Submit to a CM Meeting

- Name and organizational details of the change-requestor
- Actual change details, such as the time and duration of the change
- Any possible impacts (high, low, medium) to significant user groups or critical functions
- The amount of advance notice needed for impacted users via email (typically two working days)
- Evidence that the change has been tested in advance
- Signature and approval of the supervisor and her supervisor (manager)
- Whether and how rollback is possible
- Postchange, a "postmortem tab" has to confirm whether the change process was successful and any revealing comments or notes for the conclusion
- One of the tabs can be an "attachment tab" containing embedded Visio diagrams or word documentation embedded within the Excel sheet to aid discussion

If approved, the configuration change is then made (usually within the following week). The postmortem section of the change can then be updated to note any issues that occurred during the change (such as a rollback after change reversal and the causes).

In recent years, some organizations have started to operate the change management collaborative process using social networking tools at work. This allows disparate flows of information, such as emails, departmental wikis, and file-share documents, to belong to a unified thread for future reference.

8. Rehearse the Inevitable: Disaster Recovery

Possible disaster scenarios can range from the mundane to the biblical in proportion. In intranet or general IT terms, recovering from a disaster successfully can mean resuming critical IT support functions for mission-critical business functions. Whether such recovery is smooth and hassle-free depends on how prior disaster-recovery planning occurs and how this plan is tested to address all relevant shortcomings adequately.

The first task when planning for disaster recovery (DR) is to assess the business impact of a certain type of disaster on the functioning of an intranet using business impact analysis (BIA). BIA involves certain metrics; again, off-the shelf software tools are available to assist with this effort. The scenario could be a natural hurricane-induced power outage or a human-induced critical application crash. In any one of these scenarios, one needs to assess the type of impact in time, productivity, and financial terms.

BIAs can take into consideration the breadth of impact. For example, if the power outage is caused by a hurricane or an earthquake, support from generator vendors or the electricity utility could be hard to get because of the large demands for their services. BIAs also need to take into account historical and local weather priorities. Though there could be possibilities of hurricanes occurring in California or earthquakes occurring along the Gulf Coast of Florida, for most practical purposes the chances of those disasters occurring in those locales are pretty remote. Historical data can be helpful for prioritizing contingencies.

Once the business impacts are assessed to categorize critical systems, a disaster recovery (DR) plan can be organized and tested. The criteria for recovery have two types of metrics: a recovery point objective (RPO) and a recovery time objective (RTO).

In the DR plan, the RPO refers to how far back or "back to what point in time" that backup data has to be recovered. This timeframe generally dictates how often tape backups are taken, which can again depend on the criticality of the data. The most common scenario for medium-sized IT shops is daily incremental backups and a weekly full backup on tape. Tapes are sometimes changed automatically by the tape backup appliances.

One important thing to remember is to rotate tapes (that is, put them on a life-cycle plan by marking them for expiry) to make sure that tapes have complete data integrity during a restore. Most tape manufacturers have marking schemes for this task. Although tapes are still relatively expensive, the extra amount spent on always having fresh tapes ensures that there are no nasty surprises at the time of a crucial data recovery.

RTO refers to how long it takes to restore backed up or recovered data to its original state for resuming normal business processes. The critical factor here is cost. It will cost much more to restore data within an hour using an online backup process or to resume operations using a hotsite rather than a five-hour restore using stored tape backups. If business process resumption is critical, cost becomes less a factor.

DR also has to take into account resumption of communication channels. If network and telephone links aren't up, having a timely tape restore does little good to resume business functions. Extended campus network links are often dependent on leased lines from major vendors such as Verizon and AT&T, so having a trusted vendor relationship with agreed-on service-level agreement (SLA) standards is a requirement.

Depending on budgets, one can configure DR to happen almost instantly, if so desired, but that is a far more costly option. Most shops with "normal" data flows are okay with business being resumed within the span of about three to fours hours or even a full working day after a major disaster. Balancing costs with business expectations is the primary factor in the DR game. Spending inordinately for a rare disaster that might never happen is a waste of resources. It is fiscally imprudent (not to mention futile) to try to prepare for every contingency possible.

Once the DR plan is more or less finalized, a DR committee can be set up under an experienced DR professional to orchestrate the routine training of users and managers to simulate disasters on a frequent basis. In most shops this means management meeting every two months to simulate a DR "war room" (command center) situation and employees going through a mandatory interactive six-month disaster recovery training, listing the DR personnel to contact.

Within the command center, roles are preassigned, and each member of the team carries out his or her role as though it were a real emergency or disaster. DR coordination is frequently modeled after the U.S. Federal Emergency Management Agency (FEMA) guidelines, an active entity that has training and certification tracks for DR management professionals.

There are scheduled simulated "generator shutdowns" in most shops on a biweekly or monthly basis to see how the systems actually function. The systems can include uninterruptible power supplies (UPSs), emergency lighting, email and cell phone notification methods, and alarm enunciators and sirens. Since electronics items in a server room are sensitive to moisture damage, gas-based Halon fire-extinguishing systems are used. These Halon systems also have a provision to test them (often twice a year) to determine their readiness. The vendor will be happy to be on retainer for these tests, which can be made part of the purchasing agreement as a SLA. If equipment is tested on a regular basis, shortcomings and major hardware maintenance issues with major DR systems can be easily identified, documented, and redressed.

In a severe disaster situation, priorities need to be exercised on what to salvage first. Clearly, trying to recover employee records, payroll records, and critical business mission data such as customer databases will take precedence. Anything irreplaceable or not easily replaceable needs priority attention.

We can divide the levels of redundancies and backups to a few progressive segments. The level of backup sophistication would of course be dependent on (1) criticality and (2) time-to-recovery criteria of the data involved.

At the very basic level, we can opt not to back up any data or not even have procedures to recover data, which means that data recovery would be a failure. Understandably, this is not a common scenario.

More typical is contracting with an archival company of a local warehouse within a 20-mile periphery. Tapes are backed up onsite and stored offsite, with the archival company picking up the tapes from your facility on a daily basis. The time to recover is dependent on retrieving the tapes from archival storage, getting them onsite, and starting a restore. The advantages here are lower cost. However, the time needed to transport tapes and recover them might not be acceptable, depending on the type of data and the recovery scenario.

Often a "coldsite" or "hotsite" is added to the intranet backup scenario. A coldsite is a smaller and scaled-down copy of the existing intranet data center that has only the most essential pared-down equipment supplied and tested for recovery but not in a perpetually ready state (powered down as in "cold," with no live connection). These coldsites can house the basics, such as a Web server, domain name servers, and SQL databases, to get an informational site started up in very short order.

A hotsite is the same thing as a coldsite except that in this case the servers are always running and the Internet and intranet connections are "live" and ready to be switched over much more quickly than on a coldsite. These are just two examples of how the business resumption and recovery times can be shortened.

Recovery can be made very rapidly if the hotsite is linked to the regular data center using fast leased-line links (such as a DS3 connection). Backups synched in real time with identical RAID disks at the hotsite over redundant high-speed data links afford the shortest recovery time.

In larger intranet shops based in defense-contractor companies, there are sometimes requirements for even faster data recovery with far more rigid standards for data integrity. To-the-second real-time data synchronization in addition to hardware synchronization ensures that duplicate sites thousands of miles away can be up and running within a matter of seconds—even faster than a hotsite. Such extreme redundancy is typically needed for critical national databases (i.e., air traffic control or customs databases that are accessed 24/7, for example).

At the highest level of recovery performance, most large database vendors offer "zero data loss" solutions, with a variety of cloned databases synchronized across the country that automatically failover and recover in an instantaneous fashion to preserve a consistent status—often free from human intervention. Oracle's version is called Data Guard; most mainframe vendors offer a similar product varying in tiers and features offered.

The philosophy here is simple: The more dollars you spend, the more readiness you can buy. However, the expense has to be justified by the level of criticality for the availability of the data.

9. Controlling Hazards: Physical and Environmental Protection

Physical access and environmental hazards are very relevant to security within the intranet. People are the primary weak link in security (as previously discussed), and controlling the activity and movement of authorized personnel and preventing access to unauthorized personnel fall within the purview of these security controls.

This important area of intranet security must first be formalized within a management-sanctioned and published P&P.

Physical access to data center facilities (as well as IT working facilities) is typically controlled using card readers. These were scanning types in the last two decades but are increasingly being converted to near-field or proximity-type access card systems. Some high-security facilities (such as bank data centers) use smartcards, which use encryption keys stored within the cards for matching keys.

Some important and common-sense topics should be discussed within the subject of physical access. First, disbursal of cards needs to be a deliberate and high-security affair requiring the signatures of at least two supervisory-level people who can be responsible for the authenticity and actual need for access credentials for a person to specific areas.

Access-card permissions need to be highly granular. An administrative person will probably never need to be in server room, so that person's access to the server room should be blocked. Areas should be categorized and catalogued by sensitivity and access permissions granted accordingly.

Physical data transmission access points to the intranet have to be monitored via digital video recording (DVR) and closed-circuit cameras if possible. Physical electronic eavesdropping can occur to unmonitored network access points in both wireline and wireless ways. There have been known instances of thieves intercepting LAN communication from unshielded Ethernet cable (usually hidden above the plenum or false ceiling for longer runs). All a data thief needs is to place a TAP box and a miniature (Wi-Fi) wireless transmitter at entry or exit points to the intranet to copy and transmit all communications. At the time of this writing, these transmitters are the size of a USB key. The miniaturization of electronics has made data theft possible for part-time thieves. Spy-store sites give determined data thieves plenty of workable options at relatively little cost.

Using a DVR solution to monitor and store access logs to sensitive areas and correlating them to the timestamps on the physical access logs can help forensic investigations in case of a physical data breach, malfeasance, or theft. It is important to remember that DVR records typically rotate and are erased every week. One person has to be in charge of the DVR so records are saved to optical disks weekly before they are erased. DVR tools need some tending to because their sophistication level often does not come up to par with other network tools.

Written or PC-based sign-in logs must be kept at the front reception desk, with timestamps. Visitor cards should have limited access to private and/or secured areas. Visitors must provide official identification, log times coming in and going out, and names of persons to be visited and the reason for their visit. If possible, visitors should be escorted to and from the specific person to be visited, to minimize the chances of subversion or sabotage.

Entries to courthouses and other special facilities have metal detectors but these may not be needed for every facility. The same goes for bollards and concrete entry barriers to prevent car bombings. In most government facilities where security is paramount, even physical entry points to parking garages have special personnel (usually deputed from the local sheriff's department) to check under cars for hidden explosive devices.

Contractor laptops must be registered and physically checked in by field support personnel, and if these laptops are going to be plugged into the local network, the laptops need to be virus-scanned by data-security personnel and checked for unauthorized utilities or suspicious software (such as hacking utilities, Napster, or other P2P threats).

Supply of emergency power to the data center and the servers has to be robust to protect the intranet from corruption due to power failures. Redundancy has to be exercised all the way from the utility connection to the servers themselves. This means there has to be more than one power connection to the data center (from more than one substation/transformer, if it is a larger data center). There has to be provision of alternate power supply (a ready generator to supply some, if not all, power requirements) in case of failure of utility power to the facility.

Power supplied to the servers has to come from more than one single UPS because most servers have two removable power inputs. Data center racks typically have two UPSs on the bottom supplying power to two separate power strips on both sides of the rack for this redundancy purpose (for seamless switchover). In case of a power failure, the UPSs instantly take over the supply of power and start beeping, alerting personnel to gracefully shut down servers. UPSs usually have reserve power for brief periods (less than 10 minutes) until the generator kicks in, relieving the UPS of the large burden of the server power loads. Generators come on trailers or are skid-mounted and are designed to run as long as there is fuel available in the tank, which can be about three to five days, depending on the model and capacity to generate (in thousands of kilowatts).

Increasingly, expensive polluting batteries have made UPSs in larger data centers fall out of favor compared to flywheel power supplies, which is a cleaner, batteryless technology to supply interim power. Maintenance of this technology is half as costly as UPS and offers the same functionality [19].

There has to be provision for rechargeable emergency luminaries within the server room as well as all areas occupied by administrators so that entry and exit are not hampered during a power failure.

Provision for fire detection and firefighting must also be made. As mentioned previously, Halon gas fire-suppression systems are appropriate for server rooms because sprinklers will inevitably damage expensive servers if the servers are still turned on during sprinkler activation.

Sensors have to be placed close to the ground to detect moisture from plumbing disasters and resultant flooding. Master shutoff valve locations for water have to be marked and identified and personnel trained on performing shutoffs periodically. Complete environmental control packages with cameras geared toward detecting any type of temperature, moisture, and sound abnormality are offered by many vendors. These sensors are connected to monitoring workstations using Ethernet LAN cabling. Reporting can occur through emails if customizable thresholds are met or exceeded.

10. Know Your Users: Personnel Security

Users working within intranet-related infrastructures have to be known and trusted. Often data contained within the intranet is highly sensitive, such as new product designs and financial or market-intelligence data gathered after much research and at great expense.

Assigning personnel to sensitive areas in IT entails attaching security categories and parameters to the positions, especially within IT. Attaching security parameters to a position is akin to attaching tags to a photograph or blog. Some parameters will be more important than others, but all describe the item to some extent. The categories and parameters listed on the personnel access form should correlate to access permissions to sensitive installations such as server rooms. Access permissions should be compliant to the organizational security policy in force at the time.

Personnel, especially those who will be handling sensitive customer data or individually identifiable health records, should be screened before hiring to ensure that they do not have felonies or misdemeanors on their records.

During transfers and terminations, all sensitive access tools should be reassessed and reassigned (or deassigned, in case of termination) for logical and physical access. Access tools can include such items as encryption tokens, company cell phones, laptops or PDAs, card keys, metal keys, entry passes, and any other company identification provided for employment. For people who are leaving the organization, an exit interview should be taken. System access should be terminated on the hour after former personnel have ceased to be employees of the company.

11. Protecting Data Flow: Information and System Integrity

Information integrity protects information and data flows while they are in movement to and from the users' desktops to the intranet. System integrity measures protect the systems that process the information (usually servers such as email or file servers). The processes to protect information can include antivirus tools, IPS and IDS tools, Web-filtering tools, and email encryption tools.

Antivirus tools are the most common security tools available to protect servers and users' desktops. Typically, enterprise-level antivirus software from larger vendors such as Symantec or McAfee will contain a console listing all machines on the network and will enable the administrators to see graphically (color or icon differentiation) which machines need virus remediation or updates. All machines will have a software client installed that does some scanning and reporting of the individual machines to the console. To save bandwidth, the management server that contains the console will be updated with the latest virus (and spyware) definition from the vendor. Then it is the management console's job to slowly update the software client in each computer with the latest definitions. Sometimes the client itself will need an update, and the console allows this to be done remotely.

IDS used to detect malware within the network from the traffic and communication malware used. There are certain patterns of behavior attached to each type of malware, and those signatures are what IDSs are used to match. IDSs are mostly defunct nowadays. The major problems with IDSs were that (1) IDSs used to produce too many false positives, which made sifting out actual threats a huge, frustrating exercise, and (2) IDSs had no teeth, that is, their functionality was limited to reporting and raising alarms. IDS devices could not stop malware from spreading because they could not block it.

Compared to IDSs, IPSs have seen much wider adoption across corporate intranets because IPS devices sit inline processing traffic at the periphery and they can block traffic or malware, depending on a much more sophisticated heuristic algorithm than IDS devices. Although IPSs are all mostly signature based, there are already experimental IPS devices that can stop threats, not on signature, but based only on suspicious or anomalous behavior. This is good news because the numbers of "zero-day" threats are on the increase, and their signatures are mostly unknown to the security vendors at the time of infection.

Web-filtering tools have gotten more sophisticated as well. Ten years ago Web filters could only block traffic to specific sites if the URL matched. Nowadays most Web filter vendors have large research arms that try to categorize specific Web sites under certain categories. Some vendors have realized the enormity of this task and have allowed the general public to contribute to this effort. The Web site www.trustedsource.org is an example; a person can go in and submit a single or multiple URLs for categorization. If they're examined and approved, the site category will then be added to the vendor's next signature update for their Web filter solution.

Web filters not only match URLs, they do a fair bit of packet-examining too these days—just to make sure that a JPEG frame is indeed a JPEG frame and not a worm in disguise. The categories of Web sites blocked by a typical midsized intranet vary, but some surefire blocked categories would be pornography, erotic sites, discrimination/hate, weapons/illegal activities, and dating/relationships.

Web filters are not just there to enforce the moral values of management. These categories—if not blocked at work—openly enable an employee to offend another employee (especially pornography or discriminatory sites) and are fertile grounds for a liability lawsuit against the employer.

Finally, email encryption has been in the news because of the various mandates such as Sarbanes-Oxley and HIPAA. Both mandates specifically mention email or communication encryption to encrypt personally identifiable financial or patient medical data while in transit. Lately the state of California (among other states) has adopted a resolution to discontinue fund disbursements to any California health organization that does not use email encryption as a matter of practice. This has caught quite a few California companies and local government entities unaware because email encryption software is relatively hard to implement. The toughest challenge yet is to train users to get used to the tool.

Email encryption works by entering a set of credentials to access the email rather than just getting email pushed to the user, as within the email client Outlook.

12. Security Assessments

A security assessment (usually done on a yearly basis for most midsized shops) not only uncovers various misconfigured items on the network and server-side sections of IT operations, it serves as a convenient blueprint for IT to activate necessary changes and get credibility for budgetary assistance from the accounting folks.

Typically most consultants take two to four weeks to conduct a security assessment (depending on the size of the intranet) and they primarily use open-source vulnerability scanners such as Nessus. GFI LANguard, Retina, and Core Impact are other examples of commercial vulnerability-testing tools. Sometimes testers also use other proprietary suites of tools (special open-source tools like the Metasploit Framework or Fragrouter) to conduct "payload-bearing attack exploits," thereby evading the firewall and the IPS to gain entry. In the case of intranet Web servers, cross-site scripting attacks can occur (see sidebar, "Types of Scans Conducted on Servers and Network Appliances during a Security Assessment").

Types of Scans Conducted on Servers and Network Appliances during a Security Assessment

- Firewalls and IPS devices configuration
- Regular and SSL VPN configuration
- Web server hardening (most critical; available as guides from vendors such as Microsoft)
- DMZ configuration
- Email vulnerabilities
- DNS server anomalies
- Database servers (hardening levels)
- Network design and access control vulnerabilities
- Internal PC health such as patching levels and incidence of spyware, malware, and so on

The results of these penetration tests are usually compiled as two separate items: (1) as a full-fledged technical report for IT and (2) as a high-level executive summary meant for and delivered to top management to discuss strategy with IT after the engagement.

13. Risk Assessments

Risk is defined as the probability of loss. In IT terms we're talking about compromising data CIA (confidentiality, integrity, or availability). Risk management is a way to manage the probability of threats causing an impact. Measuring risks using a risk assessment exercise is the first step toward managing or mitigating a risk. Risk assessments can identify network threats, their probabilities, and their impacts. The reduction of risk can be achieved by reducing any of these three factors.

Regarding intranet risks and threats, we're talking about anything from threats such as unpatched PCs getting viruses and spyware (with hidden keylogging software) to network-borne denial-of-service attacks and even large, publicly embarrassing Web vandalism threats, such as someone being able to deface the main page of the company Web site. The last is a very high-impact threat but mostly perceived to be a remote probability—unless, of course, the company has experienced this before. The awareness among vendors as well as users regarding security is at an all-time high due to security being a high-profile news item.

Any security threat assessment needs to explore and list exploitable vulnerabilities and gaps. Many midsized IT shops run specific vulnerability assessment (VA) tools in-house on a monthly basis. eEye's Retina Network Security Scanner and Foundstone's scanning tools appliance are two examples of VA tools that can be found in use at larger IT shops. These tools are consolidated on ready-to-run appliances that are usually managed through remote browser-based consoles. Once the gaps are identified and quantified, steps can be taken to gradually mitigate these vulnerabilities, minimizing the impact of threats.

In intranet risk assessments, we identify primarily Web server and database threats residing within the intranet, but we should also be mindful about the periphery to guard against breaches through the firewall or IPS.

14. Conclusion

It is true that the level of Internet hyperconnectivity among generation X and Y users has mushroomed lately, and the network periphery that we used to take for granted as a security shield has been diminished, to a large extent, because of the explosive growth of social networking and the resulting connectivity boom. However, with the various new types of incoming application traffic (VoIP, SIP, and XML traffic) to their networks, security administrators need to stay on their toes and deal with these new protocols by implementing newer tools and technology. One recent example of new technology is the application-level firewall for connecting outside vendors to intranets (also known as an XML firewall, placed within a DMZ) that protects the intranet from malformed XML and SOAP message exploits coming from outside sourced applications [20].

In conclusion, we can say that with the myriad security issues facing intranets today, most IT shops are still well equipped to defend themselves if they assess risks and, most important, train their employees regarding data security practices on an ongoing basis. The problems with threat mitigation remain largely a matter of meeting gaps in procedural controls rather than technical measures. Trained and security-aware employees are the biggest deterrent to data thefts and security breaches.

References

[1] Tapper J, Radia K. State Department contract employees fired, another disciplined for looking at passport file. ABCnews.com, March 21, 2008, http://abcnews.go.com/Politics/story?id=4492773&page=1.

[2] Laptop security blog, Absolute Software. http://blog.absolute.com/category/real-theft-reports/.

[3] Leyden J. eBayed VPN kit hands over access to council network. theregister.co.uk, September 29, 2008, www.theregister.co.uk/2008/09/29/second_hand_vpn_security_breach.

[4] Coffield B. Second criminal conviction under HIPAA. Health Care Law Blog, March 14, 2006, http://healthcarebloglaw.blogspot.com/2006/03/second-criminal-conviction-under-hipaa.html.

[5] TJX identity theft saga continues: 11 charged with pilfering millions of credit cards. Networkworld.com magazine, August 5, 2008, www.networkworld.com/community/node/30741?nwwpkg5breaches?ap15rcb.

[6] The TJX Companies, Inc. updates information on computer systems intrusion. February 21, 2007, www.tjx.com/Intrusion_Release_email.pdf.

[7] TJX class action lawsuit settlement site. The TJX Companies, Inc., and Fifth Third Bancorp, Case No. 07-10162, www.tjxsettlement.com/.
[8] Ponemon Institute announces result of survey assessing the business impact of a data security breach. May 15, 2007, www.ponemon.org/press/Ponemon_Survey_Results_Scott_and_Scott_FINAL1.pdf.
[9] Mowery J. Socialtext melds media and collaboration. cmswire.com, October 8, 2008, www.cmswire.com/cms/enterprise-20/socialtext-melds-media-and-collaboration-003270.php.
[10] Hof R. Socialtext 3.0: Will wikis finally find their place in business? Businessweek.com magazine, September 30, 2008, www.businessweek.com/the_thread/techbeat/archives/2008/09/socialtext_30_i.html.
[11] Hickey M. MSI's 3.5G Wind 120 coming in November, offer subsidized by Taiwanese Telecom. Crave.com, October 20, 2008, http://news.cnet.com/8301-17938_105-10070911-1.html?tag5mncol;title.
[12] Network Appliance, Inc. RFC Standards white paper for Internet Content Adaptation Protocol (ICAP). July 30, 2001, www.content-networking.com/references.html.
[13] National Institute of Standards and Technology, Computer Security Resource Center. http://csrc.nist.gov/.
[14] Juniper and Microsoft hook up for NAC work, May 22, 2007, PHYSORG.com, http://www.physorg.com/news99063542.html.
[15] Bocek K. What does a data breach cost? SCmagazine.com, July 2, 2007, www.scmagazineus.com/What-does-a-data-breach-cost/article/35131.
[16] NAP Program program details, Microsoft.com. www.microsoft.com/windowsserver2008/en/us/nap-features.aspx.
[17] Web application security—check your site for web application vulnerabilities. www.acunetix.com/websitesecurity/webapp-security.htm.
[18] IPS specification datasheet. TippingPoint® intrusion prevention system (IPS) technical specifications. www.tippingpoint.com/pdf/resources/datasheets/400918-007_IPStechspecs.pdf.
[19] Flywheel energy storage, Wikipedia.com. http://en.wikipedia.org/wiki/Flywheel_energy_storage.
[20] Latest standard (version 1.1) for SOAP message security standardfrom OASIS, a consortium for Web Services Security. www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf.

This page intentionally left blank

CHAPTER 10

Local Area Network Security

Dr. Pramod Pandya
California State University

Securing available resources on any corporate or academic data network is of paramount importance because most of these networks connect to the Internet for commercial or research activities. Therefore, the network is under attack from hackers on a continual basis, so network security technologies are ever evolving and playing catch-up with hackers. Around 20 years ago the number of potential users was small and the scope of any activity on the network was limited to local networks only. As the Internet expanded in its reach across national boundaries and as the number of users increased, potential risk to the network grew exponentially. Over the past 10 years ecommerce-related activities such as online shopping, banking, stock trading, and social networking have permeated extensively, creating a dilemma for both service providers and their potential clients, as to who is a trusted service provider and a trusted client on the network. Of course, this being a daunting task for security professionals, they have needed to design security policies appropriate for both the servers and their clients. The security policy must be a factor in clients' level of access to the resources. So, in whom do we place trust, and how much trust? Current network designs implement three levels of trust: most trusted, less trusted, and least trusted. Figure 10.1 reflects these levels of trust, as described here.

- The most trusted users belong to the *intranet*. These users have to authenticate to a centralize administrator to access the resources on the network.
- The less trusted users may originate from the intranet as well as the external users who are authenticated to access resources such as email and Web services.
- The least trusted users are the unauthenticated users; most of them are simply browsing the resources on the Internet with no malice intended. Of course, some are scanning the resources with the intent to break in and steal data.

These are the objectives of network security:

- *Confidentiality*. Only authorized users have access to the network.
- *Integrity*. Data cannot be modified by unauthorized users.
- *Access*. Security must be designed so that authorized users have uninterrupted access to data.

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00010-5

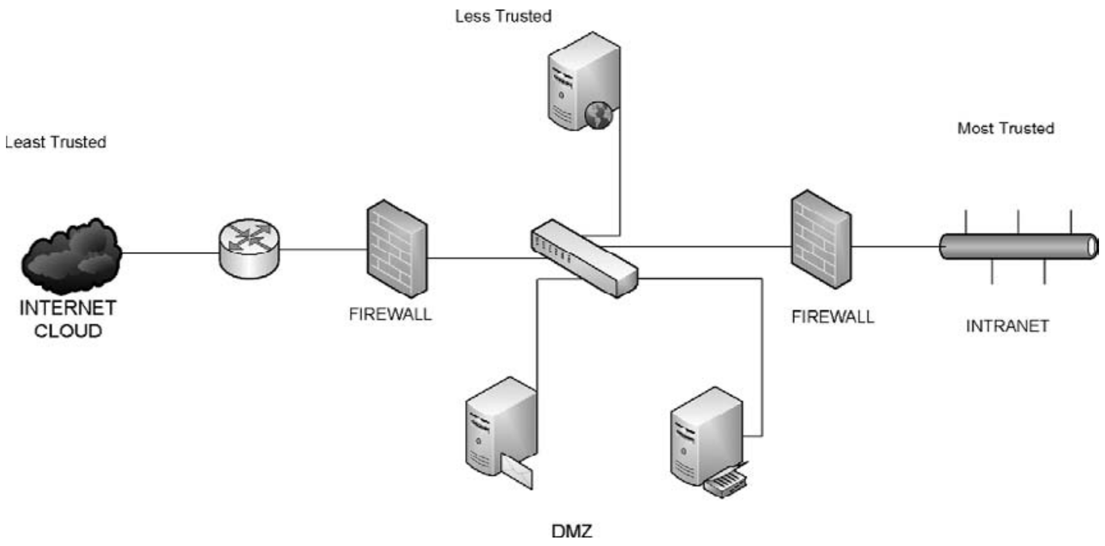


Figure 10.1: The DMZ.

Finally, the responsibility for the design and implementation of network security is headed by the chief information officer (CIO) of the enterprise network. The CIO has a pool of network administrators and legal advisers to help with this task. The network administrators define the placing of the network access controls, and the legal advisors underline the consequences and liabilities in the event of network security breaches. We have seen cases of customer records such as credit-card numbers, Social Security numbers, and personal information being stolen. The frequency of these reports has been on the increase in the past years, and consequently this has led to a discussion on the merits of encryption of stored data. One of the most quoted legal requirements on the part of any business, whether small or big, is the protection of consumer data under the Health Insurance Portability and Accountability Act (HIPAA), which restricts disclosure of health-related data and personal information.

1. Identify Network Threats

Network security threats can be in one of two categories: (1) disruptive type or (2) unauthorized access type.

Disruptive

Most LANs are designed as collapsed backbone networks using a layer-2 or layer-3 switch. If a switch or a router were to fail due to power failure, a segment or the entire network may cease to function until the power is restored. In some case, the network failure may be due to a virus attack on the secondary storage, thus leading to loss of data.

Unauthorized Access

This access type can be internal (employee) or external (intruder), a person who would attempt to break into resources such as database, file, and email or web servers that they have no permission to access. Banks, financial institutions, major corporations, and major retail businesses employ data networks to process customer transactions and store customer information and any other relevant data. Before the birth of the Internet Age, interinstitutional transactions were secured because the networks were not accessible to intruders or the general public. In the past 10 years, access to the Internet is almost universal; hence institutional data networks have become the target of frequent intruder attacks to steal customer records. One frequently reads in the news about data network security being compromised by hackers, leading to loss of credit card and debit card numbers, Social Security numbers, drivers' license numbers, and other sensitive information such as purchasing profiles. Over the years, although network security has increased, the frequency of attacks on the networks has also increased because the tools to breach the network security have become freely available on the Internet. In 1988 the U.S. Department of Defense established the Computer Emergency Response Team (CERT), whose mission is to work with the Internet community to prevent and respond to computer and network security breaches. Since the Internet is widely used for commercial activities by all kinds of businesses, the federal government has enacted stiffer penalties for hackers.

2. Establish Network Access Controls

In this section we outline steps necessary to secure networks through network controls. These network controls are either software or hardware based and are implemented in a hierarchical structure to reflect the network organization. This hierarchy is superimposed on the network from the network's perimeter to the access level per user of the network resources. The functions of the network control are to detect an unauthorized access, to prevent network security from being breached, and finally, to respond to a breach—thus the three categories of detect, prevent, and respond.

The role of prevention control is to stop unauthorized access to any resource available on the network. This could be implemented as simply as a password required to authenticate the user to access the resource on the network. For an authorized user this password can grant login to the network to access the services of a database, file, Web, print, or email server. The network administrator would need a password to access the switch or a router. The prevention control in this case is software based. An analog of hardware-based control would be, for example, if the resources such as server computers, switches, and routers are locked in a network access control room.

The role of the detection control is to monitor the activities on the network and identify an event or a set of events that could breach network security. Such an event may be a virus, spyware, or adware attack. The detection control software must, besides registering the attack, generate or trigger an alarm to notify of an unusual event so that a corrective action can be taken immediately, without compromising security.

The role of the response control is to take corrective action whenever network security is breached so that the same kind of breach is detected and any further damage is prevented.

3. Risk Assessment

During the initial design phase of a network, the network architects assess the types of risks to the network as well as the costs of recovering from attacks for all the resources that have been compromised. These cost factors can be realized using well-established accounting procedures such as cost/benefit analysis, return on investment, and total cost of ownership. These risks could range from natural disaster to an attack by a hacker. Therefore, you need to develop levels of risks to various threats to the network. You need to design some sort of spreadsheet that lists risks versus threats as well as responses to those identified threats. Of course, the spreadsheet would also mark the placing of the network access controls to secure the network.

4. Listing Network Resources

We need to identify the assets (resources) that are available on the corporate network. Of course, this list could be long, and it would make no sense to protect all the resources, except for those that are mission-critical to the business functions. Table 10.1 identifies mission-critical components of any enterprise network. You will observe that these mission-critical components need to be prioritized, since they do not all provide the same functions.

Table 10.1: Mission-Critical Components of Any Enterprise Network

Threats	Fire, Flood, Earthquake	Power Failure	Spam	Virus	Spyware, Adware	Hijacking
Resources						
Perimeter Router						
DNS Server						
WEB Server						
Email Server						
Core Switches						
Databases						

Some resources provide controlled access to a network; other resources carry sensitive corporate data. Hence the threats posed to these resources do not carry the same degree of vulnerabilities to the network. Therefore, the network access control has to be articulated and applied to each of the components listed, in varying degrees. For example, threats to DNS server pose a different set of problems from threats to the database servers. In the next section we itemize the threats to these resources and specific network access controls.

5. Threats

We need to identify the threats posed to the network from internal users as opposed to those from external users. The reason for such a distinction is that the internal users are easily traceable, compared to the external users. If a threat to the data network is successful, and it could lead to loss or theft of data or denial of access to the services offered by the network, it would lead to monetary loss for the corporation. Once we have identified these threats, we can rank them from most probable to least probable and design network security policy to reflect that ranking.

From Table 10.2, we observe that most frequent threats to the network are from viruses, and we have seen a rapid explosion in antivirus, antispamware, and spyware and adware software. Hijacking of resources such as domain name services, Web services, and perimeter routers would lead to what's most famously known as denial of service (DoS) or distributed denial of service (DDoS). Power failures can always be complemented by standby power supplies that could keep the essential resources from crashing. Natural disasters such as fire, floods, or earthquakes can be most difficult to plan for; therefore we see a tremendous growth in data protection and backup service provider businesses.

6. Security Policies

The fundamental goals of security policy are to allow uninterrupted access to the network resources for authenticated users and to deny access to unauthenticated users. Of course, this is always a balancing act between the users' demands on network resources and the

Table 10.2: Most Frequent Threats to the Network Are from Viruses

Rank	Threat
1	Virus
2	Spam
3	Spyware, Adware
4	Hijacking
5	Power Failure
6	Fire, Flood, Earthquake

evolutionary nature of information technology. The user community would prefer open access, whereas the network administrator insists on restricted and monitored access to the network.

The hacker is, in the final analysis, the arbitrator of the network security policy, since it is always the unauthorized user who discovers the potential flaw in the software. Hence, any network is as secure as the last attack that breached its security. It would be totally unrealistic to expect a secured network at all times, once it is built and secured. Therefore, network security design and its implementation represent the ultimate battle of the minds between the chief information security officer and the devil, the hacker. We can summarize that the network security policy can be as simple as to allow access to resources, or it can be several hundred pages long, detailing the levels of access and punishment if a breach is discovered. Most corporate network users now have to sign onto the usage policy of the network and are reminded that security breaches are a punishable offence.

The critical functions of a good security policy are:

- Appoint a security administrator who is conversant with users' demands and on a continual basis is prepared to accommodate the user community's needs.
- Set up a hierarchical security policy to reflect the corporate structure.
- Define ethical Internet access capabilities.
- Evolve the remote access policy.
- Provide a set of incident-handling procedures.

7. The Incident-Handling Process

The incident-handling process is the most important task of a security policy for the reason that you would not want to shut down the network in case of a network security breach. The purpose of the network is to share the resources; therefore an efficient procedure must be developed to respond to a breach. If news of the network security breach becomes public, the corporation's business practices could become compromised, thus resulting in compromise of its business operations. Therefore set procedures must be developed jointly with the business operations manager and the chief information officer. This calls for a modular design of the enterprise network so that its segments can be shut down in an orderly way, without causing panic.

Toward this end, we need a set of tools to monitor activities on the network—we need an intrusion detection and prevention system. These pieces of software will monitor network activities and log and report an activity that does not conform to usual or acceptable standards as defined by the software. Once an activity is detected and logged, response is activated. It is not merely sufficient to respond to an incident; the network administrator also has to activate tools to trace back to the source of this breach. This is critical so that the network administrator can update the security procedures to make sure that this particular incident does not take place.

8. Secure Design through Network Access Controls

A network is as secure as its weakest link in the overall design. To secure it, we need to identify the entry and exit points into the network. Since most data networks have computational nodes to process data and storage nodes to store data, stored data may need to be encrypted so that if network security is breached, stolen data may still remain confidential unless the encryption is broken. As we hear of cases of stolen data from either hacked networks or stolen nodes, encrypting data while it's being stored appears to be necessary to secure data.

The entry point to any network is a perimeter router, which sits between the external firewall and the Internet; this model is applicable to most enterprise networks that engage in some sort of ecommerce activities. Hence our first network access control is to define security policy on the perimeter router by configuring the appropriate parameters on the router. The perimeter router will filter traffic based on the range of IP addresses.

Next in the line of defense is the external firewall that filters traffic based on the state of the network connection. Additionally, the firewall could also check the contents of the traffic packet against the nature of the Transmission Control Protocol (TCP) connection requested. Following the firewall we have the so-called demilitarized zone, or DMZ, where we would place the following servers: Web, DNS, and email. We could harden these servers so that potential threatening traffic can be identified and appropriate incident response generated.

The DMZ is placed between two firewalls, so our last line of defense is the next firewall that would inspect the traffic and possibly filter out the potential threat. The nodes that are placed on the intranet can be protected by commercially available antivirus software. Last but not least, we could install on the network an intrusion detection and prevention system that will generate real-time response to a threat.

Next we address each of the network control access points. The traditional network design includes an access layer, a distribution layer, and the core layer. In the case of a local area network (LAN) we will use the access and distribution layers; the core layer would simply be our perimeter router that we discussed earlier in this section. Thus the LAN will consist of a number of segments reflecting the organizational structure. The segments could sit behind their firewall to protect one another as well, in case of network breach; segments under attack can be isolated, thus preventing a cascade-style attack on the network.

9. Intrusion Detection System Defined

An intrusion detection system, or IDS, can be both software and hardware based. IDSs listen to all the activities taking place on both the computer (node on a network) and the network itself. One could think of an IDS as like traffic police, whose function is to monitor the data packet traffic and detect and identify those data packets that match predefined unusual pattern of activities. An IDS can also be programmed to teach itself from its past activities to refine the rules for unusual activities. This should not come as a surprise, since the hackers also get smarter over time.

As we stated, the IDS collects information from a variety of system and network resources, but in actuality it captures packets of data as defined by the TCP/IP protocol stack. In this sense IDS is both a sniffer and analyzer software. IDS in its sniffer role would either capture all the data packets or select ones as specified by the configuration script. This configuration script is a set of rules that tell the analyzer what to look for in a captured data packet, then make an educated guess per rules and generate an alert. Of course, this could lead to four possible outcomes with regard to intrusion detection: false positive, false negative, true positive, or true negative. We address this topic in more detail later in the chapter.

IDSs perform a variety of functions:

- Monitor and analyze user and system activities
- Verify the integrity of data files
- Audit system configuration files
- Recognize activity of patterns, reflecting known attacks
- Statistical analysis of any undefined activity pattern

An IDS is capable of distinguishing different types of network traffic, such as an HTTP request over port 80 from some other application such as SMTP being run over port 80. We see here that an IDS understands which TCP/IP applications run over which preassigned port numbers, and therefore falsifying port numbers would be trivially detectable. This is a very easy illustration, but there are more complex attacks that are not that easy to identify, and we shall cover them later in this chapter.

The objective of intrusion detection software packages is to make possible the complex and sometimes virtually impossible task of managing system security. With this in mind, it might be worthwhile to bring to our attention two industrial-grade IDS software packages: Snort (NIDS), which runs on both Linux and Windows, and GFI LANguard S.E.L.M., a host intrusion detection system (HIDS), which runs on Windows only. Commercial-grade IDS software is designed with user-friendly interfaces that make it easy to configure scripts, which lay down the rules for intrusion detection.

Next let's examine some critical functions of an IDS:

- Can impose a greater degree of flexibility to the security infrastructure of the network
- Monitors the functionality of routers, including firewalls, key servers, and critical switches
- Can help resolve audit trails, thus often exposing problems before they lead to loss of data
- Can trace user activity from the network point of entry to the point of exit
- Can report on file integrity checks
- Can detect whether a system has been reconfigured by an attack
- Can recognize a potential attack and generate an alert
- Can make possible security management of a network by nonexpert staff

10. Network-Based IDS: Scope and Limitations

Network-based IDS (NIDS) sensors scan network packets at the router or host level, auditing data packets and logging any suspicious packets to a log file. Figure 10.2 is an example of a NIDS. The data packets are captured by a sniffer program, which is a part of the IDS

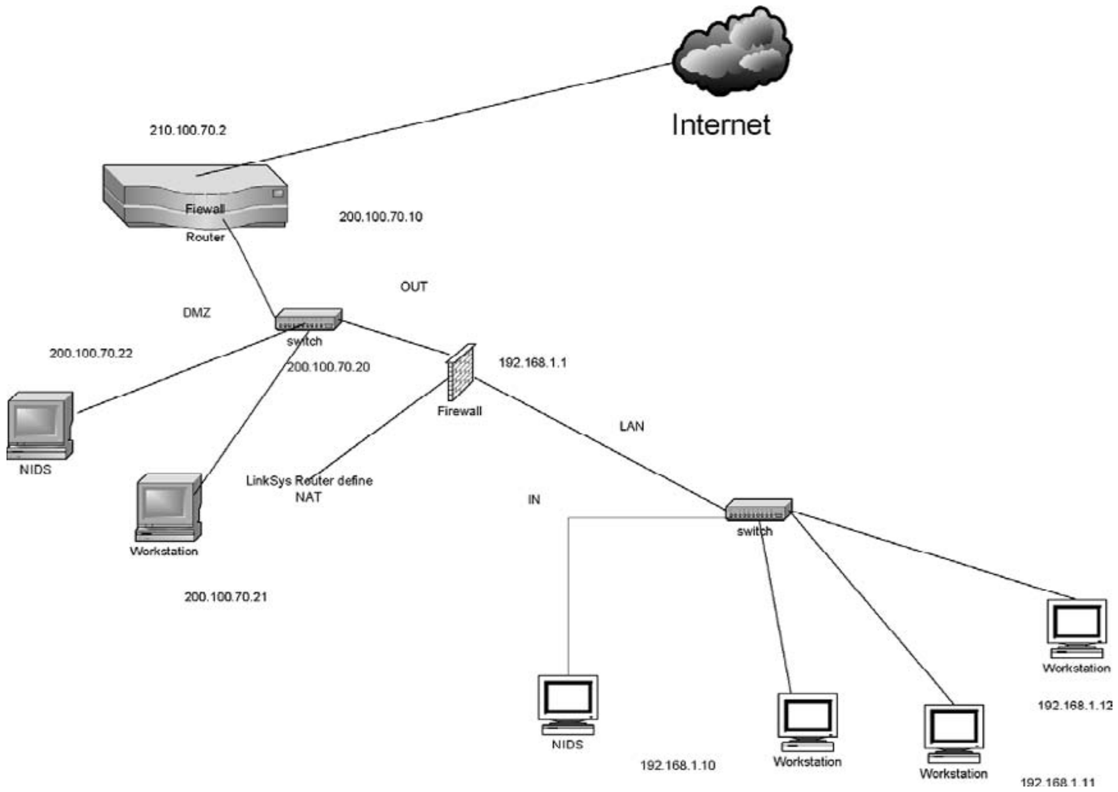


Figure 10.2: An example of a network-based intrusion detection system.

software package. The node on which the IDS software is enabled runs in promiscuous mode. In promiscuous mode, the NIDS node captures all the data packets on the network as defined by the configuration script. NIDSs have become a critical component of network security management as the number of nodes on the Internet has grown exponentially over last few years. Some of the common malicious attacks on networks are:

- IP address spoofing
- MAC address spoofing
- ARP cache poisoning
- DNS name corruption

11. A Practical Illustration of NIDS

This section illustrates the use of Snort as an example of a NIDS. The signature files are kept in the directory signatures under the directory .doc. Signature files are used to match defined signature against a pattern of bytes in the data packets, to identify a potential attack. Files marked as rules in the rules directory are used to trigger an alarm and write to the file alert .ids. Snort is installed on a node with IP address 192.168.1.22. The security auditing software Nmap is installed on a node with IP address 192.168.1.20. Nmap software is capable of generating ping sweeps, TCP SYN (half-open) scanning, TCP connect() scanning, and much more. Figure 10.2 has a node labeled *NIDS* (behind the Linksys router) on which Snort would be installed. One of the workstations would run Nmap software.

UDP Attacks

A UDP attack is generated from a node with IP address 192.168.1.20 to a node with IP address 192.168.1.22. Snort is used to detect a possible attack. Snort's detect engine uses one of the files in DOS under directory rules to generate the alert file alert.ids. We display a partial listing (see Listing 10.1) of the alert.ids file.

Listing 10.2 shows a partial listing of DOS rules file. The rules stated in the DOS rules file are used to generate the alert.ids file.

TCP SYN (Half-Open) Scanning

This technique is often referred to as *half-open* scanning because you don't open a full TCP connection. You send a SYN packet, as though you were going to open a real connection, and wait for a response. A SYN|ACK indicates that the port is listening. An RST is indicative of a nonlistener. If a SYN|ACK is received, you immediately send an RST to tear down the connection (actually, the kernel does this for you). The primary advantage of this scanning technique is that fewer sites will log it! SYN scanning is the -*s* option of Nmap.

```
[**] [1:0:0] DOS Teardrop attack [**]
[Priority: 0]
01/26-11:37:10.667833 192.168.1.20:1631 -> 192.168.1.22:21
UDP TTL:128 TOS:0x0 ID:60940 IpLen:20 DgmLen:69
Len: 41
[**] [1:0:0] DOS Teardrop attack [**]
[Priority: 0]
01/26-11:37:10.668460 192.168.1.20:1631 -> 192.168.1.22:21
UDP TTL:128 TOS:0x0 ID:60940 IpLen:20 DgmLen:69
Len: 41
[**] [1:0:0] DOS Teardrop attack [**]
[Priority: 0]
01/26-11:37:11.667926 192.168.1.20:1631 -> 192.168.1.22:21
UDP TTL:128 TOS:0x0 ID:60941 IpLen:20 DgmLen:69
Len: 41
[**] [1:0:0] DOS Teardrop attack [**]
[Priority: 0]
01/26-11:37:11.669424 192.168.1.20:1631 -> 192.168.1.22:21
UDP TTL:128 TOS:0x0 ID:60941 IpLen:20 DgmLen:69
Len: 41
[**] [1:0:0] DOS Teardrop attack [**]
[Priority: 0]
01/26-11:37:12.669316 192.168.1.20:1631 -> 192.168.1.22:21
UDP TTL:128 TOS:0x0 ID:60942 IpLen:20 DgmLen:69
Len: 41
```

Listing 10.1: An alert.ids file.

```
# (C) Copyright 2001, Martin Roesch, Brian Caswell, et al.  All rights reserved.
# $Id: dos.rules,v 1.30.2.1 2004/01/20 21:31:38 jh8 Exp $
#----------
# DOS RULES
#----------
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"DOS Jolt attack";
fragbits: M; dsize:408; reference:cve,CAN-1999-0345;
classtype:attempted-dos; sid:268; rev:2;)
alert udp $EXTERNAL_NET any -> $HOME_NET any (msg:"DOS Teardrop attack";
id:242; fragbits:M; reference:cve,CAN-1999-0015;
reference:url,www.cert.org/advisories/CA-1997-28.html;
reference:bugtraq,124; classtype:attempted-dos; sid:270; rev:2;)
alert udp any 19 <> any 7 (msg:"DOS UDP echo+chargen bomb";
reference:cve,CAN-1999-0635; reference:cve,CVE-1999-0103;
classtype:attempted-dos; sid:271; rev:3;)
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"DOS IGMP dos attack";
content:"|02 00|"; depth: 2; ip_proto: 2; fragbits: M+;
reference:cve,CVE-1999-0918; classtype:attempted-dos; sid:272; rev:2;)
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"DOS IGMP dos attack";
content:"|00 00|"; depth:2; ip_proto:2; fragbits:M+; reference:cve,CVE-
1999-0918; classtype:attempted-dos; sid:273; rev:2;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"DOS ath";
content:"+++ath"; nocase; itype: 8; reference:cve,CAN-1999-1228;
reference:arachnids,264; classtype:attempted-dos; sid:274; rev:2;)
```

Listing 10.2: The DOS rules file.

A SYN attack is generated using Nmap software from a node with IP address 192.168.1.20 to a node with IP address 192.168.1.22. Snort is used to detect for a possible attack. Snort's detect engine uses scan and ICMP rules files under directory rules to generate the alert file alert.ids. A partial listing of alert.ids file is shown in Listing 10.3.

A partial listing of the scan rules appears in Listing 10.4.

Listing 10.5 contains a partial listing of the ICMP rules.

The following points must be noted about NIDS:

- One NIDS is installed per LAN (Ethernet) segment.
- Place NIDS on the auxiliary port on the switch and then link all the ports on the switch to that auxiliary port.
- When the network is saturated with traffic, the NIDS might drop packets and thus create a potential "hole."
- If the data packets are encrypted, the usefulness of an IDS is questionable.

```
[**] [1:469:1] ICMP PING NMAP [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/24-19:28:24.774381 192.168.1.20 -> 192.168.1.22
ICMP TTL:44 TOS:0x0 ID:29746 IpLen:20 DgmLen:28
Type:8  Code:0  ID:35844   Seq:45940  ECHO
[Xref => http://www.whitehats.com/info/IDS162]
[**] [1:469:1] ICMP PING NMAP [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/24-19:28:24.775879 192.168.1.20 -> 192.168.1.22
ICMP TTL:44 TOS:0x0 ID:29746 IpLen:20 DgmLen:28
Type:8  Code:0  ID:35844   Seq:45940  ECHO
[Xref => http://www.whitehats.com/info/IDS162]
[**] [1:620:6] SCAN Proxy Port 8080 attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/24-19:28:42.023770 192.168.1.20:51530 -> 192.168.1.22:8080
TCP TTL:50 TOS:0x0 ID:53819 IpLen:20 DgmLen:40
******S* Seq: 0x94D68C2  Ack: 0x0  Win: 0xC00  TcpLen: 20
[**] [1:620:6] SCAN Proxy Port 8080 attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/24-19:28:42.083817 192.168.1.20:51530 -> 192.168.1.22:8080
TCP TTL:50 TOS:0x0 ID:53819 IpLen:20 DgmLen:40
******S* Seq: 0x94D68C2  Ack: 0x0  Win: 0xC00  TcpLen: 20
[**] [1:615:5] SCAN SOCKS Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/24-19:28:43.414083 192.168.1.20:51530 -> 192.168.1.22:1080
TCP TTL:59 TOS:0x0 ID:62752 IpLen:20 DgmLen:40
******S* Seq: 0x94D68C2  Ack: 0x0  Win: 0x1000  TcpLen: 20
[Xref => http://help.undernet.org/proxyscan/]
```

Listing 10.3: Alert.ids file.

```
# (C) Copyright 2001,2002, Martin Roesch, Brian Caswell, et al.
#    All rights reserved.
# $Id: scan.rules,v 1.21.2.1 2004/01/20 21:31:38 jh8 Exp $
#-----------
# SCAN RULES
#-----------
# These signatures are representitive of network scanners.  These include
# port scanning, ip mapping, and various application scanners.
#
# NOTE: This does NOT include web scanners such as whisker.  Those are
# in web*
#
alert tcp $EXTERNAL_NET 10101 -> $HOME_NET any (msg:"SCAN myscan";
stateless; ttl: >220; ack: 0; flags: S;reference:arachnids,439;
classtype:attempted-recon; sid:613; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 113 (msg:"SCAN ident version
request"; flow:to_server,established; content: "VERSION|0A|"; depth:
16;reference:arachnids,303; classtype:attempted-recon; sid:616; rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 80 (msg:"SCAN cybercop os
probe"; stateless; flags: SF12; dsize: 0; reference:arachnids,146;
classtype:attempted-recon; sid:619; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 3128 (msg:"SCAN Squid Proxy
attempt"; stateless; flags:S,12; classtype:attempted-recon; sid:618;
rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 1080 (msg:"SCAN SOCKS Proxy
attempt"; stateless; flags:S,12;
reference:url,help.undernet.org/proxyscan/; classtype:attempted-recon;
sid:615; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 8080 (msg:"SCAN Proxy Port 8080
attempt"; stateless; flags:S,12; classtype:attempted-recon; sid:620;
rev:6;)
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN FIN"; stateless;
flags:F,12; reference:arachnids,27; classtype:attempted-recon; sid:621;
rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN ipEye SYN scan";
flags:S; stateless; seq:1958810375; reference:arachnids,236;
classtype:attempted-recon; sid:622; rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN NULL";
stateless; flags:0; seq:0; ack:0; reference:arachnids,4;
classtype:attempted-recon; sid:623; rev:2;)
```

Listing 10.4: Scan rules.

```
# (C) Copyright 2001,2002, Martin Roesch, Brian Caswell, et al.
#    All rights reserved.
# $Id: icmp.rules,v 1.19 2003/10/20 15:03:09 chrisgreen Exp $
#-----------
# ICMP RULES
#-----------
#
# Description:
# These rules are potentially bad ICMP traffic.  They include most of the
# ICMP scanning tools and other "BAD" ICMP traffic (Such as redirect
host)
#
# Other ICMP rules are included in icmp-info.rules
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP ISS Pinger";
content:"|495353504e475251|";itype:8;depth:32; reference:arachnids,158;
classtype:attempted-recon; sid:465; rev:1;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP L3retriever
Ping"; content: "ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI"; itype: 8; icode: 0;
depth: 32; reference:arachnids,311; classtype:attempted-recon; sid:466;
rev:1;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP Nemesis v1.1
Echo"; dsize: 20; itype: 8; icmp_id: 0; icmp_seq: 0; content:
"|0000000000000000000000000000000000000000|"; reference:arachnids,449;
classtype:attempted-recon; sid:467; rev:1;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP PING NMAP";
dsize: 0; itype: 8; reference:arachnids,162; classtype:attempted-recon;
sid:469; rev:1;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP icmpenum
v1.1.1"; id: 666; dsize: 0; itype: 8; icmp_id: 666 ; icmp_seq: 0;
reference:arachnids,450; classtype:attempted-recon; sid:471; rev:1;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP redirect
host";itype:5;icode:1; reference:arachnids,135; reference:cve,CVE-1999-
0265; classtype:bad-unknown; sid:472; rev:1;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP redirect
net";itype:5;icode:0; reference:arachnids,199; reference:cve,CVE-1999-
0265; classtype:bad-unknown; sid:473; rev:1;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP superscan
echo"; content:"|0000000000000000|"; itype: 8; dsize:8;
classtype:attempted-recon; sid:474; rev:1;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP traceroute
ipopts"; ipopts: rr; itype: 0; reference:arachnids,238; classtype
```

Listing 10.5: ICMP rules.

Some Not-So-Robust Features of NIDS

Network security is a complex issue with myriad possibilities and difficulties. In networks, security is also a weakest link phenomenon, since it takes vulnerability on one node to allow a hacker to gain access to a network and thus create chaos on the network. Therefore IDS products are vulnerable.

An IDS cannot compensate for weak identification and authentication. Hence you must rely on other means of identification and authentication of users. This is best implemented by token-based or biometric schemes and one-time passwords.

An IDS cannot conduct investigations of attacks without human intervention. Therefore when an incident does occur, steps must be defined to handle the incident. The incident must be followed up to determine the responsible party, then the vulnerability that allowed the problem to occur should be diagnosed and corrected. You will observe that an IDS is not capable of identifying the attacker, only the IP address of the node that served as the hacker's point of entry.

An IDS cannot compensate for weaknesses in network protocols. IP and MAC address spoofing is a very common form of attack in which the source IP or MAC address does not correspond to the real source IP or MAC address of the hacker. Spoofed addresses can be mimicked to generate DDoS attacks.

An IDS cannot compensate for problems in the integrity of information the system provides. Many hacker tools target system logs, selectively erasing records corresponding to the time of the attack and thus covering the hacker's tracks. This calls for redundant information sources.

An IDS cannot analyze all the traffic on a busy network. A network-based IDS in promiscuous mode can capture all the data packets, and as the traffic level raises, NIDS can reach a saturation point and begin to lose data packets.

An IDS cannot always deal with problems involving packet-level attacks. The vulnerabilities lie in the difference between IDS interpretation of the outcome of a network transaction and the destination node for that network session's actual handling of the transaction. Therefore, a hacker can send a series of fragmented packets that elude detection and can also launch attacks on the destination node. Even worse, the hacker can lead to DoS on the IDS itself.

An IDS has problems dealing with fragmented data packets. Hackers would normally use fragmentation to confuse the IDS and thus launch an attack.

12. Firewalls

A firewall is either a single node or a set of nodes that enforce an access policy between two networks. Firewall technology evolved to protect the intranet from unauthorized users on the Internet. This was the case in the earlier years of corporate networks. Since then, the network administrators have realized that networks can also be attacked from trusted users as well as, for example, the employee of a company. The corporate network consists of hundreds of nodes per department and thus aggregates to over a thousand or more, and now there is a need to protect data in each department from other departments. Hence, a need for internal firewalls arose to protect data from unauthorized access, even if they are employees of the corporation. This need has, over the years, led to design of segmented IP networks, such that internal firewalls would form barriers within barriers, to restrict a potential break-in to an IP segment rather than expose the entire corporate network to a hacker. For this reason, network security has grown into a multibillion-dollar business.

Almost every intranet, whether of one node or many nodes, is always connected to the Internet, and thus a potential number of hackers wait to attack it. Thus every intranet is an IP network, with TCP- and UDP-based applications running over it. The design of TCP and UDP protocols require that every client/server application interacts with other client/server applications through TCP and UDP port numbers. As we stated earlier, these TCP and UDP port numbers are well known and hence give rise to a necessary weakness in the network. TCP and UDP port numbers open up "holes" in the networks by their very design. Every Internet and intranet point of entry has to be guarded, and you must monitor the traffic (data packets) that enter and leave the network.

A firewall is a combination of hardware and software technology, namely a sort of sentry, waiting at the points of entry and exit to look out for an unauthorized data packet trying to gain access to the network. The network administrator, with the help of other IT staff, must first identify the resources and the sensitive data that need to be protected from the hackers. Once this task has been accomplished, the next task is to identify who would have access to these identified resources and the data. We should pointed out that most of the networks in any corporation are never designed and built from scratch but are added to an existing network as the demand for networking grows with the growth of the business. So, the design of the network security policy has multilayered facets as well.

Once the network security policy is defined and understood, we can identify the proper placement of the firewalls in relation to the resources on the network. Hence, the next step would be to actually place the firewalls in the network as nodes. The network security policy now defines access to the network, as implemented in the firewall. These access rights to the network resources are based on the characteristics of TCP/IP protocols and the TCP/UDP port numbers.

Firewall Security Policy

The firewall enables the network administrator to centralize access control to the campuswide network. A firewall logs every packet that enters and leaves the network. The network security policy implemented in the firewall provides several types of protection, including the following:

- Block unwanted traffic
- Direct incoming traffic to more trustworthy internal nodes
- Hide vulnerable nodes that cannot easily be secured from external threats
- Log traffic to and from the network

A firewall is transparent to authorized users (both internal and external), whereas it is not transparent to unauthorized users. However, if the authorized user attempts to access a service that is not permitted to that user, a denial of that service will be echoed, and that attempt will be logged.

Firewalls can be configured in a number of architectures, providing various levels of security at different costs of installation and operations. Figure 10.2 is an example of a design termed a *screened Subnet*. In this design, the internal network is a private IP network, so the resources on that network are completely hidden from the users who are external to that network, such as users from the Internet. In an earlier chapter we talked about public versus private IP addresses. It is agreed to by the IP community that nodes with private IP addresses will not be accessible from outside that network. Any number of corporations may use the same private IP network address without creating packets with duplicated IP addresses. This feature of IP networks, namely private IP networks, adds to network security. In Figure 10.2, we used a Linksys router to support a private IP network (192.168.1.0) implementation. For the nodes on the 192.168.1.0 network to access the resources on the Internet, the Linksys router has to translate the private IP address of the data packet to a public IP address. In our scenario, the Linksys router would map the address of the node on the 192.168.1.0 network, to an address on the public network, 200.100.70.0. This feature is known as Network Address Translation (NAT), which is enabled on the Linksys router. You can see in Figure 10.2 that the Linksys router demarks the internal (IN) network from the external (OUT) network.

We illustrate an example of network address translation, as shown in Listing 10.6. The script configures a Cisco router that translates an internal private IP address to a public IP address.

```
ip nat pool net-sf 200.100.70.50 200.100.70.60 netmask 255.255.255.0
ip nat inside source list 1 pool net-sf
!
interface Ethernet0
 ip address 192.168.1.1 255.255.255.0
 ip nat inside
!
interface Ethernet1
 ip address 200.100.70.20 255.255.255.0
 ip nat outside
access-list 1 deny 192.168.1.0 0.0.0.255
```

Listing 10.6: Network Address Translation (NAT).

Of course, configuring a Linksys router is much simpler using a Web client. An explanation of the commands and their details follow the script.

Configuration Script for sf Router

The access-list command creates an entry in a standard traffic filter list:

- Access-list "access-list-number" permit|deny source [source-mask]
- Access-list number: identifies the list to which the entry belongs; a number from 1 to 99
- Permit|deny: this entry allows or blocks traffic from the specified address
- Source: identifies source IP address
- Source-mask: identifies the bits in the address field that are matched; it has a 1 in position indicating "don't care" bits, and a 0 in any position that is to be strictly followed

The IP access-group command links an existing access list to an outbound interface. Only one access list per port, per protocol, and per direction is allowed.

- Access-list-number: indicates the number of the access list to be linked to this interface
- In|out: selects whether the access list is applied to the incoming or outgoing interface; out is the default

NAT is a feature that operates on a border router between an inside private addressing scheme and an outside public addressing scheme. The inside private address is 192.168.1.0 and the outside public address is chosen to be 200.100.70.0. Equivalently, we have an intranet on the inside and the Internet on the outside.

13. Dynamic NAT Configuration

First a NAT pool is configured from which outside addresses are allocated to the requesting inside hosts: IP NAT pool "pool name" "start outside IP address" "finish outside IP address." Next the access-list is defined to determine which inside networks are translated by the NAT router: access-list "unique access-list number" permit|deny "inside IP network address." Finally the NAT pool and the access list are correlated:

- IP NAT inside source list "unique access list number" pool "pool name"
- Enable the NAT on each interface of the NAT router
- IP NAT inside ++++++++++++++++ IP NAT outside

You will note that only one interface may be configured as outside, yet multiple interfaces may be configured as inside, with regard to Static NAT configuration:

- IP NAT inside source static "inside IP address" "outside IP address"
- IP NAT inside source static 192.168.1.100 200.100.70.99

14. The Perimeter

In Figure 10.3, you will see yet another IP network labeled *demilitarized zone* (DMZ). You may ask, why yet another network? The rationale behind this design is as follows.

The users that belong to IN might want to access the resources on the Internet, such as read their email and send email to the users on the Internet. The corporation needs to advertise its products on the Internet.

The DMZ is the perimeter network, where resources have public IP addresses, so they are seen and heard on the Internet. The resources such as the Web (HTTP), email (SMTP), and domain name server (DNS) are placed in the DMZ, whereas the rest of the resources that

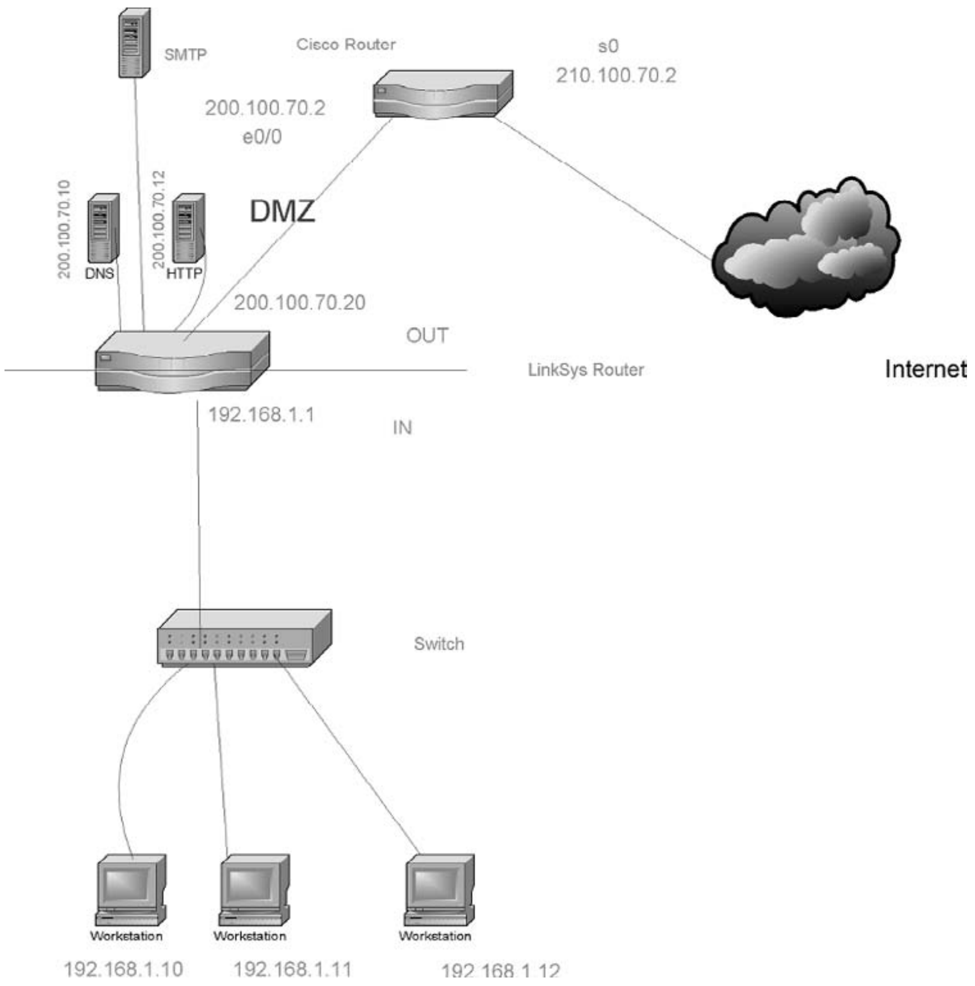


Figure 10.3: An illustrative firewall design.

belong to this corporation are completely hidden behind the Linksys router. The resources in the DMZ can be attacked by the hacker because they are open to users on the Internet. The relevant TCP and UDP port numbers on the servers in the DMZ have to be left open to the incoming and outgoing traffic. Does this create a potential "hole" in the corporate network? The answer to this is both yes and no. Someone can compromise the resources in the DMZ without the entire network being exposed to a potential attack.

The first firewall is the Cisco router, and it is the first line of defense, were network security policy implemented. On the Cisco router it is known as the Access Control List (ACL). This firewall will allow external traffic to inbound TCP port 80 on the Web server, TCP port 25 on the email server, and TCP and UDP port 53 on the DNS server. The external traffic to the rest of the ports will be denied and logged.

The second line of defense is the Linksys router that will have well-known ports closed to external traffic. It too will monitor and log the traffic. It is acceptable to place email and the Web server behind the Linksys router on the private IP network address. Then you will have to open up the TCP ports 80 and 25 on the Linksys router so that the external traffic can be mapped to ports 80 and 25, respectively. This would slow down the traffic because the Linksys router (or any commercial-grade router) would have to constantly map the port numbers back and forth. Finally, the DNS server would always need to be placed in the DMZ with a public IP address, since it will be used to resolve domain names by both internal and external users. This decision has to be left to the corporate IT staff.

15. Access List Details

The Cisco router in Figure 10.3 can be configured with the following access list to define network security policy. Building an access list in the configuration script of the router does not activate the list unless it is applied to an interface. "ip access-group 101 in" applies the access-list 101 to the serial interface of the router. Some of the access-list commands are explained here. For more information on Cisco access-list commands, visit the Cisco Web site (www.cisco.com):

- ip access-group group no. {in|out}: default is out
- What is the group number?
- The group number is the number that appears in the access-list command line
- What is {in|out}?
- In implies that the packet enters the router's interface from the network
- Out implies that the packet leaves the router's interface to the network

All TCP packets are IP packets, but all IP packets are not TCP packets. Therefore, entries matching on IP packets are more generic than matching on TCP, UDP, or ICMP packets. Each entry in the access list is interpreted (see Listing 10.7) from top to bottom for each

```
interface serial0
ip address 210.100.70.2
ip access-group 101 in
!
access-list 101 permit tcp any any established
```

Listing 10.7: Access-list configuration script.

packet on the specified interface. Once a match is reached, the remaining access-list entries are ignored. Hence, the order of entries in an access list is very critical, and therefore more specific entries should appear earlier on.

This permits TCP from any host to any host if the ACK or RST bit is set, which indicates that it is part of an established connection. You will note that in a TCP Full Connect, the first packet from the source node does not have the ACK bit set. The keyword *established* is meant to prevent an untrusted user from initiating a connection while allowing packets that are part of already established TCP connections to go through:

- Access-list 101 permit udp any gt 1023 host 200.100.70.10 eq 53
- Permit UDP protocol from any host with port greater than 1023 to the DNS server at port 53
- Access-list 101 permit ip any host 200.100.70.12
- Permit IP from any host to 200.100.70.12

or

- Access-list 101 permit TCP any 200.100.70.12 eq 80
- Permit any host to engage with our HTTP server on port 80 only
- Access-list 101 permit icmp any echo-reply
- Permit ICMP from any host to any host if the packet is in response to a ping request
- Access-list 101 deny ip any any

The last access-list command is implicit (that is, not explicitly stated). The action of this last access-list is to deny all other packets.

16. Types of Firewalls

Conceptually, there are three types of firewalls:

- *Packet filtering*—permit packets to enter or leave the network through the interface on the router on the basis of protocol, IP address, and port numbers.
- *Application-layer firewall*—a proxy server that acts as an intermediate host between the source and the destination nodes.
- *Stateful-inspection layer*—validates the packet on the basis of its content.

17. Packet Filtering: IP Filtering Routers

An IP packet-filtering router permits or denies the packet to either enter or leave the network through the interface (incoming and outgoing) on the basis of the protocol, IP address, and the port number. The protocol may be TCP, UDP, HTTP, SMTP, or FTP. The IP address under consideration would be both the source and the destination addresses of the nodes. The port numbers would correspond to the well-known port numbers. The packet-filtering firewall has to examine every packet and make a decision on the basis of defined ACL; additionally it will log the following guarded attacks on the network:

- A hacker will attempt to send IP spoofed packets using raw sockets (we will discuss more about usage of raw sockets in the next chapters)
- Log attempted network scanning for open TCP and UDP ports—NIDS will carry out this detective work in more detail
- SYN attacks using TCP connect(), and TCP half open
- Fragment attacks

18. Application-Layer Firewalls: Proxy Servers

These are proxy servers that act as an intermediary host between the source and the destination nodes. Each of the sources would have to set up a session with the proxy server and then the proxy server would set up a session with the destination node. The packets would have to flow through the proxy server. There are examples of Web and FTP proxy servers on the Internet. The proxy servers would also have to be used by the internal users, that is, the traffic from the internal users will have to run through the proxy server to the outside network. Of course, this slows the flow of packets, but you must pay the price for added network security.

19. Stateful Inspection Firewalls

In here the firewall will examine the contents of the packets before permitting them to either enter or leave the network. The contents of the packets must conform with the protocol declared with the packet. For example, if the protocol declared is HTTP, the contents of the packet must be consistent with the HTTP packet definition.

20. Network-Based IDS Complements Firewalls

A firewall acts as a barrier, if so designed, among various IP network segments. Firewalls may be defined among IP intranet segments to protect resources. In any corporate network, there will always be more than one firewall because an intruder could be one of the authorized network users. Hence the following points should be noted:

- Not all threats originate outside the firewall.
- The most trusted users are also the potential intruders.
- Firewalls themselves may be subject to attack.

Since the firewall sits at the boundary of the IP network segments, it can only monitor the traffic entering and leaving the interface on the firewall that connects to the network. If the intruder is internal to the firewall, the firewall will not be able to detect the security breach. Once an intruder has managed to transit through the interface of the firewall, the intruder would go undetected, which could possibly lead to stealing sensitive information, destroying information, leaving behind viruses, staging attacks on other networks, and most important, leaving spyware software to monitor the activities on the network for future attacks. Hence, a NIDS would play a critical role in monitoring activities on the network and continually looking for possible anomalous patterns of activities.

Firewall technology has been around for the past 20 years, so much has been documented about its weaknesses and strengths. Information about firewalls is freely available on the Internet. Hence a new breed of hackers has utilized *tunneling* as a means of bypassing firewall security policy. NIDS enhances security infrastructure by monitoring system activities for signs of attack and then, based on the system settings, responds to the attack as well as generates an alarm. Response to a potential attack is known as the *incident response* or *incident handling*, which combines investigation and diagnosis phases. Incident response has been an emerging technology in the past couple of years and is now an integral part of intrusion detection and prevention technology.

Finally, but not least, securing network systems is an ongoing process in which new threats arise all the time. Consequently, firewalls, NIDS, and intrusion prevention systems are continuously evolving technologies. In this chapter and subsequent chapters our focus has been and will be wired networks. However, as wireless data networks proliferate and seamlessly connect to the cellular voice networks, the risk of attacks on the wired networks is growing exponentially.

21. Monitor and Analyze System Activities

Figure 10.1 shows the placement of a NIDS, one in the DMZ and the other in the private network. This suggests at least two points on the network from which we capture data packets. The next question is the timing of the information collection, although this depends on the degree of threat perceived to the network.

If the level of perceived threat to the network is low, an immediate response to the attack is not very critical. In such a case, interval-oriented data capturing and analysis is most economical in terms of load placed on a NIDS and other resources on the network.

Additionally, there might not be full-time network security personnel to respond to an alarm triggered by the NIDS.

If the level of perceived threat is imminent and the time and the data are mission-critical to the organization, real-time data gathering and analysis are of extreme importance. Of course, the real-time data gathering would impact the CPU cycles on the NIDS and would lead to a massive amount of data storage. With real-time data capturing and analysis, real-time response to an attack can be automated with notification. In such a case, network activities can be interrupted, the incident could be isolated, and system and network recovery could be set in motion.

Analysis Levels

Capturing and storing data packets are among the manageable functions of any IDS. How do we analyze the data packets that represent potential or imminent threats to the network?

We need to examine the data packets and look for evidence that could point to a threat. Let's examine the makeup of data packets. Of course, any packet is almost encapsulated by successive protocols from the Internet model, with the data as its kernel. Potential attacks could be generated by IP or MAC spoofing, fragmented IP packets leading to some sort of DoS, saturating the resource with flooding, and much more. We should remind readers that since humans are not going to examine the data packets, this process of examination is relegated to an algorithm. This algorithm must compare the packets with a known format of the packet (signature) that suggests an attack is in progress, or it could be that there is some sort of unusual activity on the network. How does one distinguish abnormal from normal sets of activities? There must be some baseline (statistical) that indicates normal, and deviation from it would be an indicator of abnormal. We explore these concepts in the following paragraphs.

We can identify two levels of analysis: signature and statistical.

22. Signature Analysis

Signature analysis includes some sort of pattern matching of the contents of the data packets. There are patterns corresponding to known attacks. These known attacks are stored in a database, and a pattern is examined against the known pattern, which defines signature analysis. Most commercial NIDS products perform signature analysis against a database of known attacks, which is part of the NIDS software. Even though the databases of known attacks may be proprietary to the vendor, the client of this software should be able to increase the scope of the NIDS software by adding signatures to the database. Snort is open-source NIDS software, and the database of known attacks is maintained and updated by the user community. This database is an ASCII (human-readable) file.

23. Statistical Analysis

First we have to define what constitutes a normal traffic pattern on the network. Then we must identify deviations away from normal patterns as potential threats. These deviations must be arrived at by statistical analysis of the traffic patterns. A good example would be how many times records are written to a database over a given time interval, and deviations from normally accepted numbers would be an indication of an impending attack. Of course, a clever hacker could mislead the detector into accepting attack activity as normal by gradually varying behavior over time. This would be an example of a false negative.

24. Signature Algorithms

Signature analysis is based on these algorithms:

- Pattern matching
- Stateful pattern matching
- Protocol decode-based analysis
- Heuristic-based analysis
- Anomaly-based analysis

Pattern Matching

Pattern matching is based on searching for a fixed sequence of bytes in a single packet. In most cases the pattern is matched against only if the suspect packet is associated with a particular service or, more precisely, destined to and from a particular port. This helps to reduce the number of packets that must get examined and thus speed up the process of detection. However, it tends to make it more difficult for systems to deal with protocols that do not live on well-defined ports.

The structure of a signature based on the simple pattern-matching approach might be as follows: First, the packet is IPv4 and TCP, the destination port is 3333, and the payload contains the fictitious string *psuw*, trigger an alarm. In this example, the pattern *psuw* is what we were searching for, and one of the IDS rules implies to trigger an alarm. One could do a variation on this example to set up more convoluted data packets. The advantage of this simple algorithm is:

- This method allows for direct correlation of an exploit with the pattern; it is highly specific.
- This method is applicable across all protocols.
- This method reliably alerts on the pattern matched.

The disadvantages of this pattern-matching approach are as follows:

- Any modification to the attack can lead to missed events (false negatives).
- This method can lead to high false-positive rates if the pattern is not as unique as the signature writer assumed.
- This method is usually limited to inspection of a single packet and, therefore, does not apply well to the stream-based nature of network traffic such as HTTP traffic. This scenario leads to easily implemented evasion techniques.

Stateful Pattern Matching

This method of signature development adds to the pattern-matching concept because a network stream comprises more than a single atomic packet. Matches should be made in context within the state of the stream. This means that systems that perform this type of signature analysis must consider arrival order of packets in a TCP stream and should handle matching patterns across packet boundaries. This is somewhat similar to a stateful firewall.

Now, instead of looking for the pattern in every packet, the system has to begin to maintain state information on the TCP stream being monitored. To understand the difference, consider the following scenario. Suppose that the attack you are looking for is launched from a client connecting to a server and you have the pattern-match method deployed on the IDS. If the attack is launched so that in any given single TCP packet bound for the target on port 3333 the string is present, this event triggers the alarm. If, however, the attacker causes the offending string to be sent such that the fictitious *gp* is in the first packet sent to the server and *o* is in the second, the alarm does not get triggered. If the stateful pattern-matching algorithm is deployed instead, the sensor has stored the *gp* portion of the string and is able to complete the match when the client forwards the fictitious *p*. The advantages of this technique are as follows:

- This method allows for direct correlation of an exploit with the pattern.
- This method is applicable across all protocols.
- This method makes evasion slightly more difficult.
- This method reliably alerts on the pattern specified.

The disadvantages of the stateful pattern matching-based analysis are as follows:

- Any modification to the attack can lead to missed events (false negatives).
- This method can lead to high false-positive rates if the pattern is not as unique as the signature writer assumed.

Protocol Decode-Based Analysis

In many ways, intelligent extensions to stateful pattern matches are protocol decode-based signatures. This class of signature is implemented by decoding the various elements in the same manner as the client or server in the conversation would. When the elements of the protocol are identified, the IDS applies rules defined by the request for comments (RFCs) to look for violations. In some instances, these violations are found with pattern matches within a specific protocol field, and some require more advanced techniques that account for such variables as the length of a field or the number of arguments.

Consider the fictitious example of the *gwb* attack for illustration purposes. Suppose that the base protocol that the attack is being run over is the fictitious OBL protocol, and more specifically, assume that the attack requires that the illegal fictitious argument *gpp* must be passed in the OBL Type field. To further complicate the situation, assume that the Type field is preceded by a field of variable length called OBL Options. The valid list of fictitious options includes *gppi, nppi, upsnfs*, and *cvjmep*. Using the simple or the stateful pattern-matching algorithm in this case leads to false positives because the option *gppi* contains the pattern that is being searched for. In addition, because the field lengths are variable, it would be impossible to limit such false positives by specifying search start and stop locations. The only way to be certain that *gpp* is being passed in as the OBL type argument is to fully decode the protocol.

If the protocol allows for behavior that the pattern-matching algorithms have difficulty dealing with, not doing full protocol decodes can also lead to false negatives. For example, if the OBL protocol allows every other byte to be a NULL if a value is set in the OBL header, the pattern matchers would fail to see fx00ox00ox00. The protocol decode-enabled analysis engine would strip the NULLS and fire the alarm as expected, assuming that *gpp* was in the Type field. Thus, with the preceding in mind, the advantages of the protocol decode-based analysis are as follows:

- This method can allow for direct correlation of an exploit.
- This method can be more broad and general to allow catching variations on a theme.
- This method minimizes the chance for false positives if the protocol is well defined and enforced.
- This method reliably alerts on the violation of the protocol rules as defined in the rules script.

The disadvantages of this technique are as follows:

- This method can lead to high false-positive rates if the RFC is ambiguous and allows developers the discretion to interpret and implement as they see fit. These gray area protocol violations are very common.
- This method requires longer development times to properly implement the protocol parser.

Heuristic-Based Analysis

A good example of this type of signature is a signature that would be used to detect a port sweep. This signature looks for the presence of a threshold number of unique ports being touched on a particular machine. The signature may further restrict itself through the specification of the types of packets that it is interested in (that is, SYN packets). Additionally, there may be a requirement that all the probes must originate from a single source. Signatures of this type require some threshold manipulations to make them conform to the utilization patterns on the network they are monitoring. This type of signature may be used to look for very complex relationships as well as the simple statistical example given.

The advantages for heuristic-based signature analysis are that some types of suspicious and/or malicious activity cannot be detected through any other means. The disadvantages are that algorithms may require tuning or modification to better conform to network traffic and limit false positives.

Anomaly-Based Analysis

From what is seen normally, anomaly-based signatures are typically geared to look for network traffic that deviates. The biggest problem with this methodology is to first define what normal is. Some systems have hardcoded definitions of normal, and in this case they could be considered heuristic-based systems. Some systems are built to learn normal, but the challenge with these systems is in eliminating the possibility of improperly classifying abnormal behavior as normal. Also, if the traffic pattern being learned is assumed to be normal, the system must contend with how to differentiate between allowable deviations and those not allowed or representing attack-based traffic. The work in this area has been mostly limited to academia, although there are a few commercial products that claim to use anomaly-based detection methods. A subcategory of this type of detection is the profile-based detection methods. These systems base their alerts on changes in the way that users or systems interact on the network. They incur many of the same limitations and problems that the overarching category has in inferring the intent of the change in behavior.

Statistical anomalies may also be identified on the network either through learning or teaching of the statistical norms for certain types of traffic, for example, systems that detect traffic floods, such as UDP, TCP, or ICMP floods. These algorithms compare the current rate of arrival of traffic with a historical reference; based on this, the algorithms will alert to statistically significant deviations from the historical mean. Often, a user can provide the statistical threshold for the alerts. The advantages for anomaly-based detection are as follows:

- If this method is implemented properly, it can detect unknown attacks.
- This method offers low overhead because new signatures do not have to be developed.

The disadvantages are:

- In general, these systems are not able to give you intrusion data with any granularity. It looks like something terrible may have happened, but the systems cannot say definitively.
- This method is highly dependent on the environment in which the systems learn what normal is.

The following are Freeware tools to monitor and analyze network activities:

- Network Scanner, Nmap, is available from www.insecure.org. Nmap is a free open-source utility to monitor open ports on a network. The MS-Windows version is a zip file by the name nmap-3.75-win32.zip. You also need to download a packet capture library, WinPcap, under Windows. It is available from http://winpcap.polito.it. In addition to these programs, you need a utility to unzip the zipped file, which you can download from various Internet sites.
- PortPeeker is a freeware utility for capturing network traffic for TCP, UDP, or ICMP protocols. With PortPeeker you can easily and quickly see what traffic is being sent to a given port. This utility is available from www.Linklogger.com.
- Port-scanning tools such as Fport 2.0 and SuperScan 4.0 are easy to use and freely available from www.Foundstone.com.
- Network sniffer Ethereal is available from www.ethereal.com. Ethereal is a packet sniffer and analyzer for a variety of protocols.
- EtherSnoop light is a free network sniffer designed for capturing and analyzing the packets going through the network. It captures the data passing through your network Ethernet card, analyzes the data, and represents it in a readable form. EtherSnoop light is a fully configurable network analyzer program for Win32 environments. It is available from www.arechisoft.com.
- A fairly advanced tool, Snort, an open-source NIDS, is available from www.snort.org.
- UDPFlood is a stress testing tool that could be identified as a DoS agent; it is available from www.Foundstone.com.
- An application that allows you to generate a SYN attack with a spoofed address so that the remote host's CPU cycle's get tied up is Attacker, and is available from www.komodia.com.

This page intentionally left blank

CHAPTER 11

Wireless Network Security

Chunming Rong, Erdal Cayirci, Gansen Zhao, Laing Yan
University of Stavanger

With the rapid development of technology in wireless communication and microchips, wireless technology has been widely used in various application areas. The proliferation of wireless devices and wireless networks in the past decade shows the widespread use of wireless technology.

Wireless networks is a general term to refer to various types of networks that communicate without the need of wire lines. Wireless networks can be broadly categorized into two classes based on the structures of the networks: wireless ad hoc networks and cellular networks. The main difference between these two is whether a fixed infrastructure is present.

Three of the well-known cellular networks are the GSM network, the CDMA network, and the 802.11 wireless LAN. The GSM network and the CDMA network are the main network technologies that support modern mobile communication, with most of the mobile phones and mobile networks that are built based on these two wireless networking technologies and their variants. As cellular networks require fixed infrastructures to support the communication between mobile nodes, deployment of the fixed infrastructures is essential. Further, cellular networks require serious and careful topology design of the fixed infrastructures before deployment, because the network topologies of the fixed infrastructures are mostly static and will have a great impact on network performance and network coverage.

Wireless ad hoc networks do not require a fixed infrastructure; thus it is relatively easy to set up and deploy a wireless ad hoc network (see Figure 11.1). Without the fixed infrastructure, the topology of a wireless ad hoc network is dynamic and changes frequently. It is not realistic to assume a static or a specific topology for a wireless ad hoc network. On the other hand, wireless ad hoc networks need to be self-organizing; thus mobile nodes in a wireless ad hoc network can adapt to the change of topology and establish cooperation with other nodes at runtime.

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00011-7

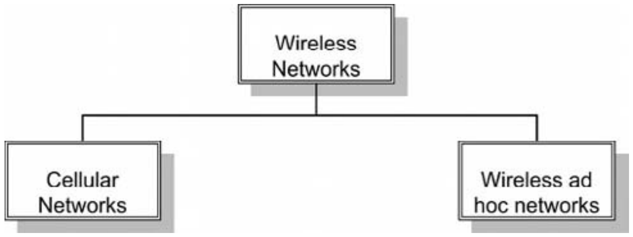


Figure 11.1: Classification of wireless networks.

Besides the conventional wireless ad hoc networks, there are two special types that should be mentioned: wireless sensor networks and wireless mesh networks. Wireless sensor networks are wireless ad hoc networks, most of the network nodes of which are sensors that monitor a target scene. The wireless sensors are mostly deprived devices in terms of computation power, power supply, bandwidth, and other computation resources. Wireless mesh networks are wireless networks with either a full mesh topology or a partial mesh topology in which some or all nodes are directly connected to all other nodes. The redundancy in connectivity of wireless networks provides great reliability and excellent flexibility in network packet delivery.

1. Cellular Networks

Cellular networks require fixed infrastructures to work (see Figure 11.2). A cellular network comprises a fixed infrastructure and a number of mobile nodes. Mobile nodes connect to the fixed infrastructure through wireless links. They may move around from within the range of one base station to outside the range of the base station, and they can move into the

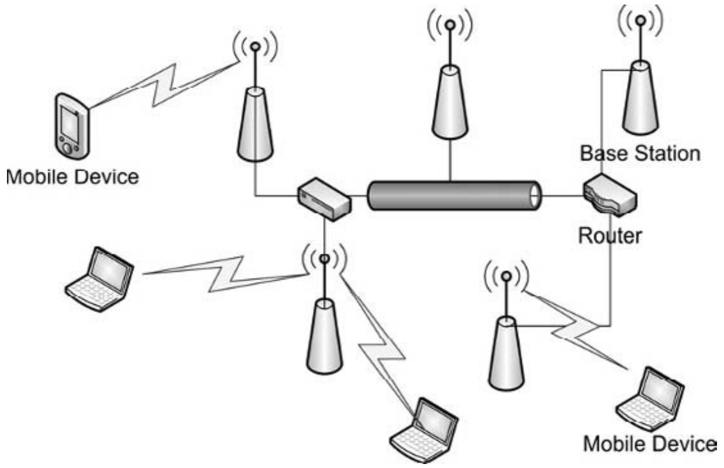


Figure 11.2: Cellular networking.

ranges of other base stations. The fixed infrastructure is stationary, or mostly stationary, including base stations, links between base stations, and possibly other conventional network devices such as routers. The links between base stations can be either wired or wireless. The links should be more substantial than those links between base stations and mobile nodes in terms of reliability, transmission range, bandwidth, and so on.

The fixed infrastructure serves as the backbone of a cellular network, providing high speed and stable connection for the whole network, compared to the connectivity between a base station and a mobile node. In most cases, mobile nodes do not communicate with each other directly without going through a base station. A packet from a source mobile node to a destination mobile node is likely to be first transmitted to the base station to which the source mobile node is connected. The packet is then relayed within the fixed infrastructures until reaching the destination base station to which the destination mobile node is connected. The destination base station can then deliver the packet to the destination mobile node to complete the packet delivery.

Cellular Telephone Networks

Cellular telephone networks offer mobile communication for most of us. With a cellular telephone network, base stations are distributed over a region, with each base station covering a small area. Each part of the small area is called a *cell*. Cell phones within a cell connect to the base station of the cell for communication. When a cell phone moves from one cell to another, its connection will also be migrated from one base station to a new base station. The new base station is the base station of the cell into which the cell phone just moved.

Two of the technologies are the mainstream for cellular telephone networks: the global system for mobile communication (GSM) and code division multiple access (CDMA).

GSM is a wireless cellular network technology for mobile communication that has been widely deployed in most parts of the world. Each GSM mobile phone uses a pair of frequency channels, with one channel for sending data and another for receiving data. Time division multiplexing (TDM) is used to share frequency pairs by multiple mobiles.

CDMA is a technology developed by a company named Qualcomm and has been accepted as an international standard. CDMA assumes that multiple signals add linearly, instead of assuming that colliding frames are completely garbled and of no value. With coding theory and the new assumption, CDMA allows each mobile to transmit over the entire frequency spectrum at all times. The core algorithm of CDMA is how to extract data of interest from the mixed data.

802.11 Wireless LANs

Wireless LANs are specified by the IEEE 802.11 series standard [1], which describes various technologies and protocols for wireless LANs to achieve different targets, allowing the maximum bit rate from 2 Mbits per second to 248 Mbits per second.

Wireless LANs can work in either access point (AP) mode or ad hoc mode, as shown in Figure 11.3. When a wireless LAN is working in AP mode, all communication passes through a base station, called an *access point*. The access point then passes the communication data to the destination node, if it is connected to the access point, or forwards the communication data to a router for further routing and relaying. When working in ad hoc mode, wireless LANs work in the absence of base stations. Nodes directly communicate with other nodes within their transmission range, without depending on a base station.

One of the complications that 802.11 wireless LANs incur is medium access control in the data link layer. Medium access control in 802.11 wireless LANs can be either distributed or centralized control by a base station. The distributed medium access control relies on the Carrier Sense Multiple Access (CSMA) with Collision Avoidance (CSMA/CA) protocol. CSMA/CA allows network nodes to compete to transmit data when a channel is idle and uses the Ethernet binary exponential backoff algorithm to decide a waiting time before retransmission when a collision occurs. CSMA/CA can also operate based on MACAW (Multiple Access with Collision Avoidance for Wireless) using virtual channel sensing. Request packets and clear-to-send (CTS) packets are broadcast before data transmission by the sender and the receiver, respectively. All stations within the range of the sender or the receiver will keep silent in the course of data transmission to avoid interference on the transmission.

The centralized medium access control is implemented by having the base station broadcast a beacon frame periodically and poll nodes to check whether they have data to send. The base station serves as a central control over the allocation of the bandwidth. It allocates bandwidth according to the polling results. All nodes connected to the base station must

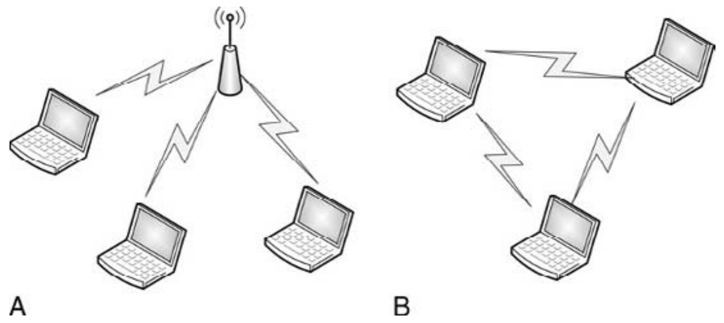


Figure 11.3: (a) A wireless network in AP mode; (b) a wireless network in ad hoc mode.

behave in accordance with the allocation decision made by the base station. With the centralized medium access control, it is possible to provide quality-of-service guarantees because the base station can control on the allocation of bandwidth to a specific node to meet the quality requirements.

2. Wireless Ad Hoc Networks

Wireless ad hoc networks are distributed networks that work without fixed infrastructures and in which each network node is willing to forward network packets for other network nodes. The main characteristics of wireless ad hoc networks are as follows.

- Wireless ad hoc networks are distributed networks that do not require fixed infrastructures to work. Network nodes in a wireless ad hoc network can be randomly deployed to form the wireless ad hoc network.
- Network nodes will forward network packets for other network nodes. Network nodes in a wireless ad hoc network directly communicate with other nodes within their ranges. When these networks communicate with network nodes outside their ranges, network packets will be forwarded by the nearby network nodes and other nodes that are on the path from the source nodes to the destination nodes.
- Wireless ad hoc networks are self-organizing. Without fixed infrastructures and central administration, wireless ad hoc networks must be capable of establishing cooperation between nodes on their own. Network nodes must also be able to adapt to changes in the network, such as the network topology.
- Wireless ad hoc networks have dynamic network topologies. Network nodes of a wireless ad hoc network connect to other network nodes through wireless links. The network nodes are mostly mobile. The topology of a wireless ad hoc network can change from time to time, since network nodes move around from within the range to the outside, and new network nodes may join the network, just as existing network nodes may leave the network.

Wireless Sensor Networks

A wireless sensor network is an ad hoc network mainly comprising sensor nodes, which are normally used to monitor and observe a phenomenon or a scene. The sensor nodes are physically deployed within or close to the phenomenon or the scene. The collected data will be sent back to a base station from time to time through routes dynamically discovered and formed by sensor nodes.

Sensors in wireless sensor networks are normally small network nodes with very limited computation power, limited communication capacity, and limited power supply. Thus a sensor may perform only simple computation and can communicate with sensors and other nodes within a short range. The life spans of sensors are also limited by the power supply.

Wireless sensor networks can be self-organizing, since sensors can be randomly deployed in some inaccessible areas. The randomly deployed sensors can cooperate with other sensors within their range to implement the task of monitoring or observing the target scene or the target phenomenon and to communicate with the base station that collects data from all sensor nodes. The cooperation might involve finding a route to transmit data to a specific destination, relaying data from one neighbor to another neighbor when the two neighbors are not within reach of each other, and so on.

Mesh Networks

One of the emerging technologies of wireless network is wireless mesh networks (WMNs). Nodes in a WMN include mesh routers and mesh clients. Each node in a WMN works as a router as well as a host. When it's a router, each node needs to perform routing and to forward packets for other nodes when necessary, such as when two nodes are not within direct reach of each other and when a route to a specific destination for packet delivery is required to be discovered.

Mesh routers may be equipped with multiple wireless interfaces, built on either the same or different wireless technologies, and are capable of bridging different networks. Mesh routers can also be classified as access mesh routers, backbone mesh routers, or gateway mesh routers. Access mesh routers provide mesh clients with access to mesh networks; backbone mesh routers form the backbone of a mesh network; and a gateway mesh router connects the backbone to an external network.

Each mesh client normally has only one network interface that provides network connectivity with other nodes. Mesh clients are not usually capable of bridging different networks, which is different from mesh routers.

Similar to other ad hoc networks, a wireless mesh network can be self-organizing. Thus nodes can establish and maintain connectivity with other nodes automatically, without human intervention. Wireless mesh networks can divided into backbone mesh networks and access mesh networks.

3. Security Protocols

Wired Equivalent Privacy (WEP) was defined by the IEEE 802.11 standard [2]. WEP is designed to protect linkage-level data for wireless transmission by providing confidentiality, access control, and data integrity, to provide secure communication between a mobile device and an access point in a 802.11 wireless LAN.

Wired Equivalent Privacy

Implemented based on shared key secrets and the RC4 stream cipher [3], WEP's encryption of a frame includes two operations (see Figure 11.4). It first produces a checksum of the data, and then it encrypts the plaintext and the checksum using RC4:

- *Checksumming*. Let c be an integrity checksum function. For a given message M, a checksum $c(M)$ is calculated and then concatenated to the end of M, obtaining a plaintext $P = \langle M, c(M) \rangle$. Note that the checksum $c(M)$ does not depend on the shared key.
- *Encryption*. The shared key k is concatenated to the end of the initialization vector (IV) v, forming $\langle v,k \rangle$. $\langle v,k \rangle$ is then used as the input to the RC4 algorithm to generate a keystream $RC4(v,k)$. The plaintext P is exclusive-or'ed (XOR, denoted by $\oplus$) with the keystream to obtain the ciphertext: $C = P \oplus RC4(v,k)$.

Using the shared key k and the IV v, WEP can greatly simplify the complexity of key distribution because it needs only to distribute k and v but can achieve a relatively very long key sequence. IV changes from time to time, which will force the $RC4$ algorithm to produce a new key sequence, avoiding the situation where the same key sequence is used to encrypt a large amount of data, which potentially leads to several types of attacks [4, 5].

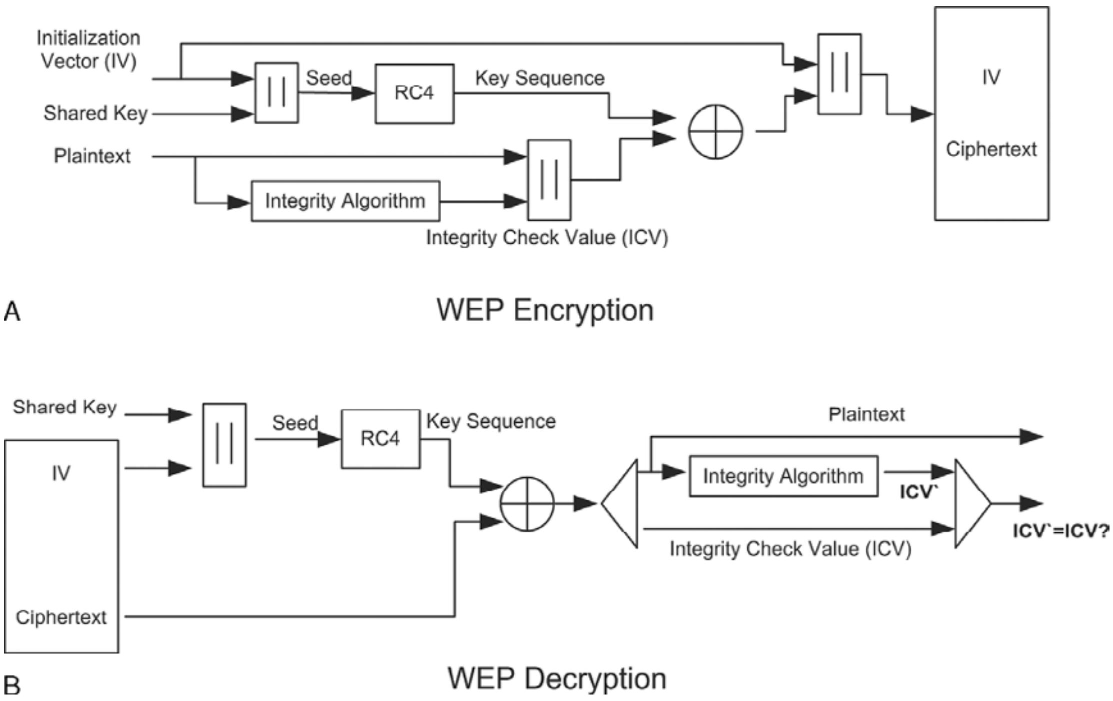


Figure 11.4: WEP encryption and decryption.

WEP combines the shared key k and the IV v as inputs to seed the $RC4$ function. 802.11B [6] specifies that the seed shall be 64 bits long, with 24 bits from the IV v and 40 bits from the shared key k. Bits 0 through 23 of the seed contain bits 0 through 23 of the IV v, and bits 24 through 63 of the seed contain bits 0 through 39 of the shared key k.

When a receiver receives the ciphertext C, it will XOR the ciphertext C with the corresponding keystream to produce the plaintext M' as follows.

WPA and WPA2

Wi-Fi Protected Access (WPA) is specified by the IEEE 802.11i standard, which is aimed at providing stronger security compared to WEP and is expected to tackle most of the weakness found in WEP [7–9].

WPA

WPA has been designed to target both enterprise and consumers. Enterprise deployment of WPA is required to be used with IEEE 802.1x authentication, which is responsible for distributing different keys to each user. Personal deployment of WPA adopts a simpler mechanism, which allows all stations to use the same key. This mechanism is called the *Pre-Shared Key* (PSK) mode.

The WPA protocol works in a similar way to WEP. WPA mandates the use of the $RC4$ stream cipher with a 128-bit key and a 48-bit initialization vector (IV), compared with the 40-bit key and the 24-bit IV in WEP.

WPA also has a few other improvements over WEP, including the Temporal Key Integrity Protocol (TKIP) and the Message Integrity Code (MIC). With TKIP, WPA will dynamically change keys used by the system periodically. With the much larger IV and the dynamically changing key, the stream cipher $RC4$ is able to produce a much longer keystream. The longer keystream improved WPA's protection against the well-known key recovery attacks on WEP, since finding two packets encrypted using the same key sequences is literally impossible due to the extremely long keystream.

With MIC, WPA uses an algorithm named Michael to produce an authentication code for each message, which is termed the *message integrity code*. The message integrity code also contains a frame counter to provide protection over replay attacks.

WPA uses the Extensible Authentication Protocol (EAP) framework [10] to conduct authentication. When a user (supplicant) tries to connect to a network, an authenticator will send a request to the user asking the user to authenticate herself using a specific type of authentication mechanism. The user will respond with corresponding authentication information. The authenticator relies on an authentication server to make the decision regarding the user's authentication.

WPA2

WPA2 is not much different from WPA. Though TKIP is required in WPA, Advanced Encryption Standard (AES) is optional. This is aimed to provide backward compatibility for WPA over hardware designed for WEP, as TKIP can be implemented on the same hardware as those for WEP, but AES cannot be implemented on this hardware. TKIP and AES are both mandatory in WPA2 to provide a higher level of protection over wireless connections. AES is a block cipher, which can only be applied to a fixed length of data block. AES accepts key sizes of 128 bits, 196 bits, and 256 bits.

Besides the mandatory requirement of supporting AES, WPA2 also introduces supports for fast roaming of wireless clients migrating between wireless access points. First, WPA2 allows the caching of a Pair-wise Master Key (PMK), which is the key used for a session between an access point and a wireless client; thus a wireless client can reconnect a recently connected access point without having to reauthenticate. Second, WPA2 enables a wireless client to authenticate itself to a wireless access point that it is moving to while the wireless client maintains its connection to the existing access point. This reduces the time needed for roaming clients to move from one access point to another, and it is especially useful for timing-sensitive applications.

SPINS: Security Protocols for Sensor Networks

Sensor nodes in sensor networks are normally low-end devices with very limited resources, such as memory, computation power, battery, and network bandwidth.

Perrig et al. [11] proposed a family of security protocols named SPINS, which were specially designed for low-end devices with severely limited resources, such as sensor nodes in sensor networks. SPINS consists of two building blocks: Secure Network Encryption Protocol (SNEP) and the "micro" version of the Timed, Efficient, Streaming, Loss-tolerant Authentication Protocol (μTESLA). SNEP uses symmetry encryption to provide data confidentiality, two-party data authentication, and data freshness. μTESLA provides authentication over broadcast streams. SPINS assumes that each sensor node shares a master key with the base station. The master key serves as the base of trust and is used to derive all other keys.

SNEP

As illustrated in Figure 11.5, SNEP uses a block cipher to provide data confidentiality and message authentication code (MAC) to provide authentication. SNEP assumes a shared counter C between the sender and the receiver and two keys, the encryption key K_{encr} and the authentication key K_{mac}.

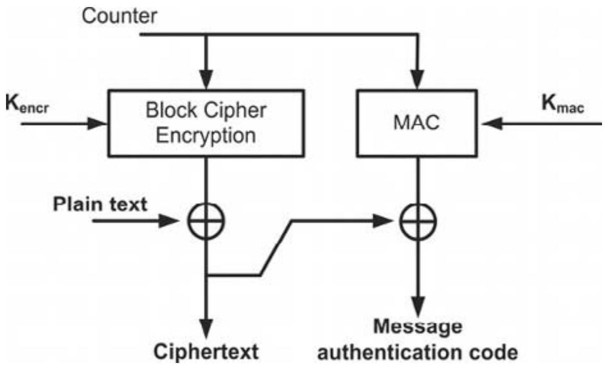


Figure 11.5: Sensor Network Encryption Protocol (SNEP).

For an outgoing message *D*, SNEP processes it as follows:

- The message *D* is first encrypted using a block cipher in counter mode with the key K_{encr} and the counter *C*, forming the encrypted text $E = \{D\}_{<Kencr,C>}$.
- A message authentication code is produced for the encrypted text *E* with the key K_{mac} and the counter *C*, forming the MAC $M = MAC(K_{mac},C|E)$ where *MAC()* is a one-way function and $C|E$ stands for the concatenation of *C* and *E*.
- SNEP increments the counter *C*.

To send the message *D* to the recipient, SNEP actually sends out *E* and *M*. In other words, SNEP encrypts *D* to *E* using the shared key K_{encr} between the sender and the receiver to prevent unauthorized disclosure of the data, and it uses the shared key K_{mac}, known only to the sender and the receiver, to provide message authentication. Thus data confidentiality and message authentication can both be implemented.

The message *D* is encrypted with the counter *C*, which will be different in each message. The same message *D* will be encrypted differently even it is sent multiple times. Thus semantic security is implemented in SNEP. The MAC is also produced using the counter *C*; thus it enables SNEP to prevent replying to old messages.

µ*TESLA*

TESLA [12–14] was proposed to provide message authentication for multicast. TESLA does not use any asymmetry cryptography, which makes it lightweight in terms of computation and overhead of bandwidth.

µTESLA is a modified version of TESLA, aiming to provide message authentication for multicasting in sensor networks. The general idea of µTESLA is that the sender splits the sending time into intervals. Packets sent out in different intervals are authenticated with different keys. Keys to authenticate packets will be disclosed after a short delay, when the keys are no longer used to send out messages. Thus packets can be authenticated when the authentication keys have been disclosed. Packets will not be tampered with while they are in transit since the keys have not been disclosed yet. The disclosed authentication keys can be verified using previous known keys to prevent malicious nodes from forging authentication keys.

μTESLA has four phases: sender setup, sending authenticated packets, bootstrapping new receivers, and authenticating packets. In the sender setup phase, a sender generates a chain of keys, K_i ($0 \leq i \leq n$). The keychain is a one-way chain such that K_i can be derived from K_j if $i \leq j$, such as a keychain K_i (i = 0, . . . , n), $K_i = F(K_{i+1})$, where F is a one-way function. The sender also decides on the starting time T_0, the interval duration T_{int}, and the disclosure delay d (unit is interval), as shown in Figure 11.6.

To send out authenticated packets, the sender attaches a MAC with each packet, where the MAC is produced using a key from the keychain and the data in the network packet. μTESLA has specific requirements on the use of keys for producing MACs. Keys are used in the same order as the key sequence of the keychain. Each of the keys is used in one interval only. For the interval $T_i = T_0 + i \times T_{int}$, the key K_i is used to produce the MACs for the messages sent out in the interval T_i. Keys are disclosed with a fixed delay d such that the key K_i used in interval T_i will be disclosed in the interval T_{i+d}. The sequence of key usage and the sequence of key disclosure are demonstrated in Figure 11.6.

To bootstrap a new receiver, the sender needs to synchronize the time with the receiver and needs to inform the new receiver of a key K_j that is used in a past interval T_j, the interval duration T_{int}, and the disclosure delay d. With a previous key K_j, the receiver will be able to verify any key K_p where $j \leq p$ using the one-way keychain's property. After this, the new receiver will be able to receive and verify data in the same way as other receivers that join the communication prior to the new receiver.

To receive and authenticate messages, a receiver will check all incoming messages if they have been delayed for more than d. Messages with a delay greater than d will be discarded,

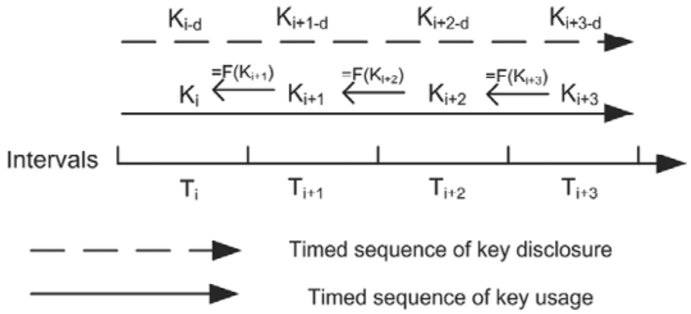


Figure 11.6: Sequences of intervals, key usages, and key disclosure.

since they are suspect as fake messages constructed after the key has been disclosed. The receiver will buffer the remaining messages for at least d intervals until the corresponding keys are disclosed. When a key K_i is disclosed at the moment $T_i + d$, the receiver will verify K_i using K_{i-1} by checking if $K_{i-1} = F(K_i)$. Once the key K_i is verified, K_i will be used to authenticate those messages sent in the interval T_i.

4. Secure Routing

Secure Efficient Ad hoc Distance (SEAD) [15] vector routing is designed based on Destination-Sequenced Distance Vector (DSDV) routing [16]. SEAD augments DSDV with authentication to provide security in the construction and exchange of routing information.

SEAD

Distance vector routing works as follows. Each router maintains a routing table. Each entry of the table contains a specific destination, a metric (the shortest distance to the destination), and the next hop on the shortest path from the current router to the destination. For a packet that needs to be sent to a certain destination, the router will look up the destination from the routing table to get the matching entry. Then the packet is sent to the next hop specified in the entry.

To allow routers to automatically discover new routes and maintain their routing tables, routers exchange routing information periodically. Each router advises its neighbors of its own routing information by broadcasting its routing table to all its neighbors. Each router will update its routing table according to the information it hears from its neighbors. If a new destination is found from the information advertised by a neighbor, a new entry is added to the routing table with the metric recalculated based on the advertised metric and the linking between the router and the neighbor. If an existing destination is found, the corresponding entry is updated only when a new path that is shorter than the original one has been found. In this case, the metric and the next hop for the specified destination are modified based on the advertised information.

Though distance vector routing is simple and effective, it suffers from possible routing loops, also known as the counting to infinity problem. DSDV [17] is one of the extensions to distance vector routing to tackle this issue. DSDV augments each routing update with a sequence number, which can be used to identify the sequence of routing updates, preventing routing updates being applied in an out-of-order manner. Newer routing updates are advised with sequence numbers greater than those of the previous routing updates. In each routing update, the sequence number will be incremented to the next even number. Only when a broken link has been detected will the router use the next odd sequence number as the sequence number for the new routing update that is to be advertised to all its neighbors. Each router maintains an even sequence number to identify the sequence of every routing update. Neighbors will only accept newer routing updates by discarding routing updates with sequence numbers less than the last sequence number heard from the router.

SEAD provides authentication on metrics' lower bounds and senders' identities by using the one-way hash chain. Let H be a hash function and x be a given value. A list of values is computed as follows:

$$h_0, h_1, h_2, \ldots, h_n$$

where $h_0 = x$ and $h_{i+1} = H(h_i)$ for $0 \leq i \leq n$. Given any value h_k that has been confirmed to be in the list, to authenticate if a given value d is on the list or not one can compute if d can be derived from h_k by applying H a certain number of times, or if h_k can be derived from d by applying H to d a certain number of times. If either d can be derived from h_k or h_k can be derived from d within a certain number of steps, it is said that d can be authenticated by h_k.

SEAD assumes an upper bound $m - 1$ on the diameter of the ad hoc network, which means that the metric of a routing entry will be less than m. Let $h_0, h_1, h_2, \ldots, h_n$ be a hash chain where $n = m \times k$ and $k \in Z+$.

For an update with the sequence number i and the metric value of j, the value $h_{(k-i)m+j}$ is used to authenticate the routing update entry.

By using $h_{(k-i)m+j}$ to authenticate the routing update entry, a node is actually disclosing the value $h_{(k-i)m+j}$ and subsequently all h_p where $p \geq (k - i)m + j$, but not any value h_q where $q \leq (k - i)m + j$.

Using a hash value corresponding to the sequence number and metric in a routing update entry allows the authentication of the update and prevents any node from advertising a route to some destination, forging a greater sequence number or a smaller metric.

To authenticate the update, a node can use any given earlier authentic hash value h_p from the same hash chain to authenticate the current update with sequence number i and metric j. The current update uses the hash value $h_{(k-i)m+j}$ and $(k - i)m + j \leq p$, thus h_p can be computed from $h_{(k-i)m+j}$ by applying H for $(k - i)m + j - p$ times.

The disclosure of $h_{(k-i)m+j}$ does not disclose any value h_q where $q \leq (k - i)m + j$. Let a fake update be advised with a sequence number p and metric q, where $p \geq i$ and $q \leq j$, or $q \leq j$. The fake update will need to use the hash value $h_{(k-p)m+q}$. If the sequence number p is greater than i or the metric q is less than j, $(k - p)m + q < (k - i)m + j$. This means that a hash value $h_{(k-p)m+q}$ that has not been disclosed is needed to authenticate the update. Since the value $h_{(k-p)m+q}$ has not been disclosed, the malicious node will not be able to have it to fake a routing update.

Ariadne

Ariadne [18] is a secure on-demand routing protocol for ad hoc networks. Ariadne is built on the Dynamic Source Routing protocol (DSR) [19].

Routing in Ariadne is divided into two stages: the route discovery stage and the route maintenance stage. In the route discovery stage, a source node in the ad hoc network tries to find a path to a specific destination node. The discovered path will be used by the source node as the path for all communication from the source node to the destination node until the discovered path becomes invalid. In the route maintenance stage, network nodes identify broken paths that have been found. A node sends a packet along a specified route to some destination. Each node on the route forwards the packet to the next node on the specified route and tries to confirm the delivery of the packet to the next node. If a node fails to receive an acknowledgment from the next node, it will signal the source node using a ROUTE ERROR packet that a broken link has been found. The source node and other nodes on the path can then be advised of the broken link.

The key security features Ariadne adds onto the route discovery and route maintenance are node authentication and data verification for the routing relation packets. Node authentication is the process of verifying the identifiers of nodes that are involved in Ariadne's route discovery and route maintenance, to prevent forging routing packets. In route discovery, a node sends out a ROUTE REQUEST packet to perform a route discovery. When the ROUTE REQUEST packet reaches the destination node, the destination node verifies the originator identity before responding. Similarly, when the source node receives a ROUTE REPLY packet, which is a response to the ROUTE REQUEST packet, the source node will also authenticate the identity of the sender. The authentication of node identities can be of one of the three methods: TELSA, digital signatures, and Message Authentication Code (MAC).

Data verification is the process of verifying the integrity of the node list in route discovery for the prevention of adding and removing nodes from the node list in a ROUTE RQUEST. To build a full list of nodes for a route to a destination, each node will need to add itself into the node list in the ROUTE REQUEST when it forwards the ROUTE REQUEST to its neighbor. Data verification protects the node list by preventing unauthorized adding of nodes and unauthorized removal of nodes.

ARAN

Authenticated Routing for Ad hoc Networks (ARAN) [20] is a routing protocol for ad hoc networks with authentication enabled. It allows routing messages to be authenticated at each node between the source nodes and the destination nodes. The authentication that ARAN has implemented is based on cryptographic certificates.

ARAN requires a trusted certificate server, the public key of which is known to all valid nodes. Keys are assumed to have been established between the trusted certificate server and nodes. For each node to enter into a wireless ad hoc network, it needs to have a certificate issued by the trusted server. The certificate contains the IP address of the node, the public key of the node, a time stamp indicating the issue time of the certification, and the expiration time of the certificate. Because all nodes have the public key of the trusted server, a certificate can be verified by all nodes to check whether it is authentic. With an authentic certificate and the corresponding private key, the node that owns the certificate can authenticate itself using its private key.

To discover a route from a source node to the destination node, the source node sends out a route discovery packet (RDP) to all its neighbors. The RDP is signed by the source node's private key and contains a nonce, a time stamp, and the source node's certificate. The time stamp and the nonce work to prevent replay attacks and flooding of the RDP.

The RDP is then rebroadcast in the network until it reaches the destination. The RDP is rebroadcast with the signature and the certificate of the rebroadcaster. On receiving an RDP, each node will first verify the source's signature and the previous node's signature on the RDP.

On receiving an RDP, the destination sends back a reply packet (REP) along the reverse path to the source after validating the RDP. The REP contains the nonce specified in the RDP and the signature from the destination node.

The REP is unicast along the reverse path. Each node on the path will put its own certificate and its own signature on the RDP before forwarding it to the next node. Each node will also verify the signatures on the RDP. An REP is discarded if one or more invalid signatures are found on the REP.

When the source receives the REP, it will first verify the signatures and then the nonce in the REP. A valid REP indicates that a route has been discovered. The node list on a valid REP suggests an operational path from the source node to the destination node that is found.

As an on-demand protocol, nodes keep track of route status. If there has been no traffic for a route's lifetime or a broken link has been detected, the route will be deactivated. Receiving data on an inactive route will force a node to signal an error state by using an error (ERR) message. The ERR message is signed by the node that produces it and will be forwarded to the source without modification. The ERR message contains a nonce and a time stamp to ensure that the ERR message is fresh.

SLSP

Secure Link State Routing Protocol (SLSP) [21] is a secure routing protocol for ad hoc network building based on link state protocols. SLSP assumes that each node has a public/private key pair and has the capability of signing and verifying digital signatures. Keys are bound with the Medium Access Code and the IP address, allowing neighbors within transmission range to uniquely verify nodes if public keys have been known prior to communication.

In SLSP, each node broadcasts its IP address and the MAC to its neighbor with its signature. Neighbors verify the signature and keep a record of the pairing IP address and the MAC. The Neighbor Lookup Protocol (NLP) of SLSP extracts and retains the MAC and IP address of each network frame received by a node. The extracted information is used to maintain the mapping of MACs and IP addresses.

Nodes using SLSP periodically send out link state updates (LSUs) to advise the state of their network links. LSU packets are limited to propagating within a zone of their origin node, which is specified by the maximum number of hops. To restrict the propagation of LSU packets, each LSU packet contains the *zone radius* and the *hops traversed* fields. Let the maximum hop be R; X, a random number; and H be a hash function. *Zone–radius* will be initialized to $H^R(X)$ and *hops_traversed* be initialized to $H(X)$. Each LSU packet also contains a TTL field initialized as $R - 1$. If $TTL < 0$ or $H(hops\text{–}traversed) = zone\text{–}radius$, a node will not rebroadcast the LSU packet. Otherwise, the node will replace the *hops–traversed* field with $H(hops\text{–}traversed)$ and decrease TTL by one. In this way, the hop count is authenticated. SLSP also uses signatures to protect LSU packets. Receiving nodes can verify the authenticity and the integrity of the received LSU packets, thus preventing forging or tampering with LSU packets.

5. Key Establishment

Because wireless communication is open and the signals are accessible by anyone within the vicinity, it is important for wireless networks to establish trust to guard the access to the networks. Key establishment builds relations between nodes using keys; thus security services, such as authentication, confidentiality, and integrity can be achieved for the communication between these nodes with the help of the established keys.

The dynamically changing topology of wireless networks, the lack of fixed infrastructure of wireless ad hoc and sensor networks, and the limited computation and energy resources of sensor networks have all added complication to the key establishment process in wireless networks.

Bootstrapping

Bootstrapping is the process by which nodes in a wireless network are made aware of the presence of others in the network. On bootstrapping, a node gets its identifying credentials that can be used in the network the node is trying to join. Upon completion of the bootstrapping, the wireless network should be ready to accept the node as a valid node to join the network.

To enter a network, a node needs to present its identifying credential to show its eligibility to access the network. This process is called *preauthentication*. Once the credentials are accepted, network security associations are established with other nodes.

These network security associations will serve as further proof of authorization in the network. Security associations can be of various forms, including symmetric keys, public key pairs, hash key chains, and so on. The security associations can be used to authenticate nodes. Security associations may expire after a certain period of time and can be revoked if necessary. For example, if a node is suspected of being compromised, its security association will be revoked to prevent the node accessing the network. The actual way of revocation depends on the form of the security associations.

Bootstrapping in Wireless ad hoc Networks

Wireless ad hoc networks bring new challenges to the bootstrapping process by their lack of a centralized security infrastructure. It is necessary to build a security infrastructure in the bootstrapping phase. The trust infrastructure should be able to accept nodes with valid credentials to enter the network but stop those nodes without valid credentials from joining the network and establish security association between nodes within the network.

To build such a trust infrastructure, we can use any one of the following three supports: prior knowledge, trusted third parties, or self-organizing capability. Prior knowledge is information that has been set on valid nodes in advance, such as predistributed secrets or preset shared keys. This information can be used to distinguish legitimate nodes from malicious ones. Only nodes with prior knowledge will be accepted to enter the network. For example, the predistributed secrets can be used to authenticate legitimate nodes, so the network can simply reject those nodes without the predistributed secrets so that they can't enter the network.

Trusted third parties can also be used to support the establishment of the trust infrastructure. The trusted third party can be a Certificate Authority (CA), a base station of the wireless network, or any nodes that are designated to be trusted. If trusted third parties are used, all nodes must mutually agree to trust them and derive their trust on others from the trusted third parties. One of the issues with this method is that trusted third parties are required to be available for access by all nodes across the whole network, which is a very strong assumption for wireless networks as well as an impractical requirement.

It is desirable to have a self-organizing capability for building the trust infrastructure for wireless networks, taking into account the dynamically changing topology of wireless ad hoc networks. Implementing a self-organizing capability for building the trust infrastructure often requires an out-of-band authenticated communication channel or special hardware support, such as tamper-proof hardware tokens.

Bootstrapping in Wireless Sensor Networks

Bootstrapping nodes in wireless sensor networks is also challenging for the following reasons:

- *Node capture.* Sensor nodes are normally deployed in an area that is geographically close or inside the monitoring environment, which might not be a closed and confined area under guard. Thus sensor nodes are vulnerable to physical capture because it might be difficult to prevent physical access to the area.
- *Node replication.* Once a sensor node is compromised, it is possible for adversaries to replicate sensor nodes by using the secret acquired from the compromised node. In this case, adversaries can produce fake legitimate node that cannot be distinguished by the network.
- *Scalability.* A single-sensor network may comprise a large number of sensor nodes. The more nodes in a wireless sensor network, the more complicated it is for bootstrapping.
- *Resource limitation.* Sensor nodes normally have extremely limited computation power and memory as well as limited power supply and weak communication capability. This makes some of the deliberate algorithms and methods not applicable to wireless sensor networks. Only those algorithms that require a moderate amount of resources can be implemented in wireless sensor networks.

Bootstrapping a sensor node is achieved using an incremental communication output power level to discover neighbors nearby. The output power level is increased step by step from the minimum level to the maximum level, to send out a HELLO message. This will enable the sensor node to discover neighbors in the order of their distance from the sensor node, from the closest to the farthest away.

Key Management

Key management schemes can be classified according to the way keys are set up (see Figure 11.7). Either keys are managed based on the contribution from all participating nodes in the network or they are managed based on a central node in the network. Thus key management schemes can be divided into contributory key management schemes, in which all nodes work equally together to manage the keys, and distributed key management schemes, in which only one central node is responsible for key management [22].

Classification

The distributed key management scheme can be further divided into symmetric schemes and public key schemes. Symmetric key schemes are based on private key cryptography, whereby shared secrets are used to authenticate legitimate nodes and to provide secure communication between them. The underlying assumption is that the shared secrets are known only to

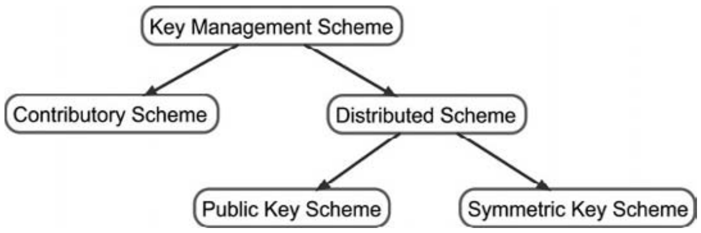


Figure 11.7: Key management schemes.

legitimate nodes involved in the interaction. Thus proving the knowledge of the shared secrets is enough to authenticate legitimate nodes. Shared secrets are distributed via secure channels or out-of-band measures. Trust on a node is established if the node has knowledge of a shared secret.

Public key schemes are built on public key cryptography. Keys are constructed in pairs, with a private key and a public key in each pair. Private keys are kept secret by the owners. Public keys are distributed and used to authenticate nodes and to verify credentials. Keys are normally conveyed in certificates for distribution. Certificates are signed by trusted nodes for which the public keys have been known and validated. Trust on the certificates will be derived from the public keys that sign the certificates. Note that given g^i *(mod p)* and g^j *(mod p)*, it is hard to compute $g^{i * j}$ *(mod p)* without the knowledge of i and j.

Contributory Schemes

Diffie–Hellman (D-H) [23] is a well-known algorithm for establishing shared secrets. The D-H algorithm's strength depends on the discrete log problem: It is hard to calculate s if given the value g^s (mod p), where p is a large prime number.

Diffie–Hellman Key Exchange

D-H was designed for establishing a shared secret between two parties, namely node A and node B. Each party agrees on a large prime number p and a generator g. A and B each choose a random value i and j, respectively. A and B are then exchanged with the public values g^i (mod p) and g^j (mod p). On the reception of g^j (mod p) from B, A is then able to calculate the value $g^{j \times i}$ (mod p). Similarly, B computes $g^{i \times j}$ (mod p). Thus a shared secret, $g^{i \times j}$ (mod p), has been set up between A and B.

ING

Ingemarsson, Tang, and Wong (ING) [24] extends the D-F key exchange to a group of n members, $d_1, \ldots, d_n$. All group members are organized in a ring, where each member has a left neighbor and a right neighbor. Node d_i has a right neighbor d_{i-1} and a left neighbor d_{i+1}. Note that for node d_i, its right neighbor is d_{i+1}; for node d_{i+1}, its left neighbor is d_i.

Same as the D-F algorithm, all members in an ING group assume a large prime number p and a generator g. Initially, node d_i will choose a random number r_i. At the first round of key exchange, node d_i will compute $g^{r}{}_{i}$ (mod p) and send it to its left neighbor d_{i+1}. At the same time, node d_i also receives the public value $g^{r}{}_{i-1}$ (mod p) from its right neighbor d_{i-1}. From the second round on, let q be the value that node d_i received in the previous round, node d_i will compute a new public value $q^{d}{}_{i}$ (mod p). After $n - 1$ rounds, the node d_i would have received a public value, g^k (mod p) where $k = \prod_{m=1}{}^{i-1} r_m \times \prod_{s=i+1}{}^{n} r_s$, from its right neighbors. With the public value received at the $n-1$th round, the node d_i can raise it to the power of r^i to compute the value g^l (mod p) where $l = \prod_{m=1}{}^{n}{}_{rm}$.

Hypercube and Octopus (H&O)

The Hypercube protocol [25] assumes that there are 2^d nodes joining to establish a shared secret and all nodes are organized as a d-dimensional vector space $GF(2)^d$ Let $b_1, \ldots, b_d$ be the basic of $GF(2)^d$ The hypercube protocol takes d rounds to complete:

- In the first round, every participant $v \in GF(2)^d$ chooses a random number r_v and conducts a D-H key exchange with another participant $v + b_1$, with the random values r_v and r_{v+b1}, respectively.
- In the ith round, every participant $v \in GF(2)^d$ performances a D-H key exchange with the participant $v + b_i$, where both v and $v + b_i$ use the value generated in the previous round as the random number for D-H key exchange.

This algorithm can be explained using a complete binary tree to make it more comprehensible. All the nodes are put in a complete binary tree as leaves, with leaves at the 0–level and the root at the d-level. D-H key exchanges are performed from the leaves up to the root. The key exchange takes d rounds:

- In the first round, each leaf chooses a random number k and performs a D-H key exchange with its sibling leaf, which has a random number j, and the resulting value $\mathrm{g}^{\mathrm{k} \times \mathrm{j}}$ (mod p) is saved as the random value for the parent node of the above two leaves.
- In the ith round, each node at the $i - 1$ level performs a D-H key exchange with its sibling node using the random numbers m and n, respectively, that they received in the previous round. The resulting value $g^{m \times n}$ (mod p) is saved as the random value for the parent node of the above two nodes.

After d rounds, the root of the complete binary tree contains the established shared secret s.

The hypercube protocol assumes that there are 2^{d} network nodes. The octopus protocol removes the assumption and extends the hypercube protocol to work with an arbitrary number of nodes. Thus the octopus protocol can be used to establish a shared key for a node set containing an arbitrary number of nodes.

Distributed Schemes

A partially distributed threshold CA scheme [26] works with a normal PKI system where a CA exists. The private key of the CA is split and distributed over a set of *n* server nodes using a (*k*,*n*) secret-sharing scheme [27]. The (*k*,*n*) secret-sharing scheme allows any *k* or more server nodes within the *n* server nodes to work together to reveal the CA's private key. Any set of nodes with fewer than *k* nodes will not be able to reveal the CA's private key. With the threshold signature scheme [28], any *k* of the *n* nodes can cooperate to sign a certificate. Each of the *k* nodes produces a piece of the signature on the request of signing a given certificate. With all the *k* pieces of the signature, a valid signature, which is the same as the one produced using the CA's private key, can be produced by combining the *k* pieces of the signature.

Partially Distributed Threshold CA Scheme

In this way, the partial distributed threshold CA scheme can avoid the bottleneck of the centralized CA of conventional PKI infrastructures. As long as there are at least *k* of the *n* nodes available, the network can always issue and sign new certificates. Attacks to any single node will not bring the whole CA down. Only when an attack manages to paralyze $n - k$ or more nodes will the CA's signing service not be available.

To further improve the security of the private key that is distributed over the *n* nodes, proactive security [29] can be imposed. Proactive security forces the private key shares to be refreshed periodically. Each refreshment will invalidate the previous share held by a node. Attacks on multiple nodes must complete within a refresh period to succeed. To be specific, only when an attack can compromise *k* or more nodes within a refresh period can the attack succeed.

While conventional PKI systems depend on directories to publish public key certificates, it is suggested that certificates should be disseminated to communication peers when establishing a communication channel with the partial distributed threshold CA scheme. This is due to the fact that the availability of centralized directories cannot be guaranteed in wireless networks. Therefore it is not realistic to assume the availability of a centralized directory.

Self-Organized Key Management (PGP-A)

A self-organized key management scheme (PGP-A) [30] has its basis in the Pretty Good Privacy (PGP) [31] scheme. PGP is built based on the "web of trust" model, in which all nodes have equal roles in playing a CA. Each node generates its own public/private key pair and signs other nodes' public keys if it trusts the nodes. The signed certificates are kept by nodes in their own certificate repositories instead of being published by centralized directories in the X.509 PKI systems [32].

PGP-A treats trust as transitive, so trust can be derived from a trusted node's trust on another node, that is, if node A trusts node B, and node B trusts node C, then A should also trust C if A knows the fact that node B trusts node C.

To verify a key of a node u, a node j will merge its certificate repository with those of j's trusted nodes, and those of the nodes trusted by j's trusted nodes, and so forth. In this way, node j can build up a web of trust in which node j is at the center of the web and j's directly trusted nodes as node j's neighbors. Node l is linked with node k if node k trusts node l. Node j can search the web of trust built as above to find a path from j to u. If such as path exists, let it be a sequence of nodes S: $node_i$ where $i = 1, \ldots, n$, n be the length of the path, and $node_1 = j$ and $node_n = u$. This means that $node_i$ trust $node_{i+1}$ for all $i = 1, \ldots, n - 1$. Therefore u can be trusted by j. The path S represents a verifiable chain of certificates. PGP-A does not guarantee that a node u that should be trusted by node j will always be trusted by node j, since there are chances that the node j fails to find a path from node j to node u in the web of trust. This might be due to the reason that node j has not acquired enough certificates from its trusted nodes to cover the path from node j to node u.

Self-healing Session Key Distribution

The preceding two key management schemes are public key management schemes. The one discussed here, a self-healing session key distribution [33], is a symmetric key management scheme. In such a scheme, keys can be distributed either by an online key distribution server or by key predistribution. A key predistribution scheme normally comprises the key predistribution phase, the shared-key discovery phase, and the path key establishment phase.

In the key predistribution phase, a key pool of a large number of keys is created. Every key can be identified by a unique key identifier. Each network node is given a set of keys from the key pool. The shared-key discovery phase begins when a node tries to communicate with the others. All nodes exchange their key identifiers to find out whether there are any keys shared with others. The shared keys can then be used to establish a secure channel for communication. If no shared key exists, a key path will need to be discovered. The key path is a sequence of nodes with which all adjacent nodes share a key. With the key path, a message can travel from the first node to the last node securely, by which a secure channel can be established between the first node and the last node.

The self-healing session key distribution (S-HEAL) [34] assumes the existence of a group manager and preshared secrets. Keys are distributed from the group manager to group members. Let h be a polynomial, where for a node i, node i knows about $h(i)$. Let K be the group key to be distributed, K is covered by h in the distribution: $f(x) = h(x) + K$. The polynomial $f(x)$ is the information that the group manager sends out to all its group members. For node j, node j will calculate $K = f(j) - h(j)$ to reveal the group key. Without the knowledge of $h(j)$, node j will not be able to recover K.

To enable revocation in S-HEAL, the polynomial $h(x)$ is replaced by a bivariate polynomial $s(x,y)$. The group key is covered by the bivariate polynomial $s(x,y)$ when it is distributed to group members, in the way that $f(N,x) = s(N,x) + K$. Node i must calculate $s(N,i)$ to recover K. The revocation enabled S-HEAL tries to stop revoked nodes to calculate $s(N,i)$, thus preventing them to recover K.

Let s of degree t; then $t + 1$ values are needed to compute $s(x,i)$. Assuming that $s(i,i)$ is predistributed to node i, node i will need another t values to recover $s(N,i)$, namely $s(r_1,x), \ldots, s(r_t,x)$. These values will be disseminating to group members together with the key update. If the group manager wants to revoke node i, the group manager can set one of the values $s(r_1,x), \ldots, s(r_t,x)$ to $s(i,x)$. In this case, node i obtains only t values instead of $t + 1$ values. Therefore, node i will not be able to compute $s(x,i)$, thus it will not be able to recover K. This scheme can only revoke maximum t nodes at the same time.

References

[1] L. M. S. C. of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, technical report, IEEE Standard 802.11, 1999 ed., 1999.

[2] L. M. S. C. of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, technical report, IEEE Standard 802.11, 1999 ed., 1999.

[3] Rivest RL. The RC4 encryption algorithm, RSA Data Security, Inc., March 1992 technical report..

[4] Dawson E, Nielsen L. Automated cryptanalysis of XOR plaintext strings. Cryptologia 1996;20(2), April.

[5] Singh S. The code book: the evolution of secrecy from Mary, Queen of Scots, to quantum cryptography. Doubleday, 1999.

[6] L. M. S. C. of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, technical report, IEEE Standard 802.11, 1999 ed.; 1999.

[7] Arbaugh WA. An inductive chosen plaintext attack against WEP/WEP2, IEEE Document 802.11-01/230, May 2001.

[8] Walker JR. Unsafe at any key size; an analysis of the WEP encapsulation, IEEE Document 802.11-00/362, October 2000.

[9] Borisov N, Goldberg I, Wagner D. Intercepting Mobile Communications: The Insecurity of 802.11, MobiCom, 2001.

[10] Aboba B, Blunk L, Vollbrecht J, Carlson J, Levkowetz EH. Extensible Authentication Protocol (EAP), request for comment. Network Working Group; 2004.

[11] Perrig A, Szewczyk R, Wen V, Culler D, Tygar JD. SPINS: Security protocols for sensor networks, MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking, 2001.

[12] Perrig A, Canetti R, Xiaodong Song D, Tygar JD. Efficient and secure source authentication for multicast, NDSS 01: Network and Distributed System Security Symposium, 2001.

[13] Perrig A, Tygar JD, Song D, Canetti R. Efficient authentication and signing of multicast streams over lossy channels, SP '00: Proceedings of the 2000 IEEE Symposium on Security and Privacy; 2000.

[14] Perrig A, Canetti R, Tygar JD, Song D. RSA CryptoBytes. 2002:5.

[15] Hu Y-C, Johnson DB, Perrig A. SEAD. Secure efficient distance vector routing for mobile wireless ad hoc networks. WMCSA '02: Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications, Washington, DC: IEEE Computer Society; 2002. p. 3.

[16] Perkins CE, Bhagwat P. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. SIGCOMM Comput. Commun Rev 1994;24(4):234–44.

[17] Perkins CE, Bhagwat P. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. SIGCOMM Comput. Commun Rev 1994;24(4):234–44.

[18] Hu Y-C, Perrig A, Johnson D. Ariadne: a secure on-demand routing protocol for ad hoc networks. Wireless Networks Journal 2005;11(1).

[19] Johnson DB, Maltz DA. Dynamic source routing in ad hoc wireless networks. *Mobile Computing*. Kluwer Academic Publishers; 1996. p. 153–81.

[20] Sanzgiri K, Dahill B, Levine BN, Shields C, Belding-Royer EM. A secure routing protocol for ad hoc networks. 10th IEEE International Conference on Network Protocols (ICNP'02), 2002.

[21] Papadimitratos P, Haas ZJ. Secure link state routing for mobile ad hoc networks. *saint-w, 00*, 2003.

[22] Cayirci E, Rong C. Security in wireless ad hoc, sensor, and mesh networks. John Wiley & Sons; 2008.

[23] Diffie W, Hellman ME. New directions in cryptography. IEEE Transactions on Information Theory 1976;IT-22(6):644–54.

[24] Ingemarsson I, Tang D, Wong C. A conference key distribution system. IEEE *Transactions on Information Theory* 1982;28(5):714–20, September.

[25] Becker K, Wille U. Communication complexity of group key distribution. ACM conference on computer and communications security 1998.

[26] Zhou L, Haas ZJ. Securing ad hoc networks. IEEE Network 1999;13:24–30.

[27] Shamir A. How to share a secret. Comm. ACM 1979;22(11).

[28] Desmedt Y. Some recent research aspects of threshold cryptography. ISW 1997;158–73.

[29] Canetti R, Gennaro A, Herzberg D. Naor, Proactive security: Long-term protection against break-ins. CryptoBytes 1997;3(1), Spring.

[30] Capkun S, Buttyán L, Hubaux J-P. Self-organized public-key management for mobile ad hoc networks. IEEE Trans Mob Comput 2003;2(1):52–64.

[31] Zimmermann P. The Official PGP User's Guide. The MIT Press; 1995.

[32] ITU-T. Recommendation X.509, ISO/IEC 9594-8. Information Technology: Open Systems Interconnection–The Directory: Public-key and Attribute Certificate Frameworks. 4th ed. 2000, ITU.

[33] Staddon J, Miner SK, Franklin MK, Balfanz D, Malkin M, Dean D. Self-healing key distribution with revocation. IEEE Symposium on Security and Privacy 2002.

[34] Staddon J, Miner SK, Franklin MK, Balfanz D, Malkin M, Dean D. Self-healing key distribution with revocation. IEEE Symposium on Security and Privacy 2002.

CHAPTER 12

Cellular Network Security

Peng Liu, Thomas F. LaPorta, Kameswari Kotapati
Pennsylvania State University

In recent years, cellular networks have become open public networks to which end subscribers have direct access. This has greatly increased the threats to the cellular network. Though cellular networks have vastly advanced in their performance abilities, the security of these networks still remains highly outdated. As a result, they are one of the most insecure networks today—so much so that using simple off-the-shelf equipment, any adversary can cause major network outages affecting millions of subscribers.

In this chapter, we address the security of the cellular network. We educate readers on the current state of security of the network and its vulnerabilities. We also outline the cellular network specific attack taxonomy, also called the *three-dimensional attack taxonomy*. We discuss the vulnerability assessment tools for cellular networks. Finally, we provide insights as to why the network is so vulnerable and why securing it can prevent communication outages during emergencies.

1. Introduction

Cellular networks are high-speed, high-capacity voice and data communication networks with enhanced multimedia and seamless roaming capabilities for supporting cellular devices. With the increase in popularity of cellular devices, these networks are used for more than just entertainment and phone calls. They have become the primary means of communication for finance-sensitive business transactions, lifesaving emergencies, and life-/mission-critical services such as E-911. Today these networks have become the lifeline of communications.

A breakdown in the cellular network has many adverse effects, ranging from huge economic losses due to financial transaction disruptions; loss of life due to loss of phone calls made to emergency workers; and communication outages during emergencies such as the September 11, 2001, attacks. Therefore, it is a high priority for the cellular network to function accurately.

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00012-9

It must be noted that it is not difficult for unscrupulous elements to break into the cellular network and cause outages. The major reason for this is that cellular networks were not designed with security in mind. They evolved from the old-fashioned telephone networks that were built for performance. To this day, the cellular network has numerous well-known and unsecured vulnerabilities providing access to adversaries. Another feature of cellular networks is network relationships (also called *dependencies*) that cause certain types of errors to propagate to other network locations as a result of regular network activity. Such propagation can be very disruptive to the network, and in turn it can affect subscribers. Finally, Internet connectivity to the cellular network is another major contributor to the cellular network's vulnerability because it gives Internet users direct access to cellular network vulnerabilities from their homes.

To ensure that adversaries do not access the network and cause breakdowns, a high level of security must be maintained in the cellular network. However, though great efforts have been made to improve the cellular network in terms of support for new and innovative services, greater number of subscribers, higher speed, and larger bandwidth, very little has been done to update the security of the cellular network. Accordingly, these networks have become highly attractive targets to adversaries, not only because of their lack of security but also due to the ease with which these networks can be exploited to affect millions of subscribers.

In this chapter we analyze the security of cellular networks. Toward understanding the security issues in cellular networks, the rest of the chapter is organized as follows. We present a comprehensive overview of cellular networks with a goal of providing a fundamental understanding of their functioning. Next we present the current state of cellular network security through an in-depth discussion on cellular network vulnerabilities and possible attacks. In addition, we present the cellular network specific attack taxonomy. Finally, we present a review of current cellular network vulnerability assessment techniques and conclude with a discussion.

2. Overview of Cellular Networks

The current cellular network is an evolution of the early-generation cellular networks that were built for optimal performance. These early-generation cellular networks were proprietary and owned by reputable organizations. They were considered secure due to their proprietary ownership and their *closed nature*, that is, their control infrastructure was unconnected to any public network (such as the Internet) to which end subscribers had direct access. Security was a nonissue in the design of these networks.

Recently, connecting the Internet to the cellular network has not only imported the Internet vulnerabilities to the cellular network, it has also given end subscribers direct access to the control infrastructure of the cellular network, thereby opening the network. Also, with the increasing demand for these networks, a large number of new network operators have come into the picture. Thus, the current cellular environment is no longer a safe, closed network but rather an insecure, open network with many unknown network operators having nonproprietary access to it. Here we present a brief overview of the cellular network architecture.

Overall Cellular Network Architecture

Subscribers gain access to the cellular network via radio signals enabled by the radio access network, as shown in Figure 12.1. The radio access network is connected to the wireline portion of the network, also called the *core network*. Core network functions include servicing subscriber requests and routing traffic. The core network is also connected to the Public Switched Telephone Network (PSTN) and the Internet, as illustrated in Figure 12.1 [1].

The PSTN is the circuit-switched public voice telephone network that is used to deliver voice telephone calls on the *fixed landline telephone network*. The PSTN uses Signaling System No. 7 (SS7), a set of telephony signaling protocols defined by the International Telecommunication Union (ITU) for performing telephony functions such as call delivery, call routing, and billing. The SS7 protocols provide a universal structure for telephony network signaling, messaging, interfacing, and network maintenance. PSTN connectivity to the core network enables mobile subscribers to call fixed network subscribers, and vice versa. In the past, PSTN networks were also closed networks because they were unconnected to other public networks.

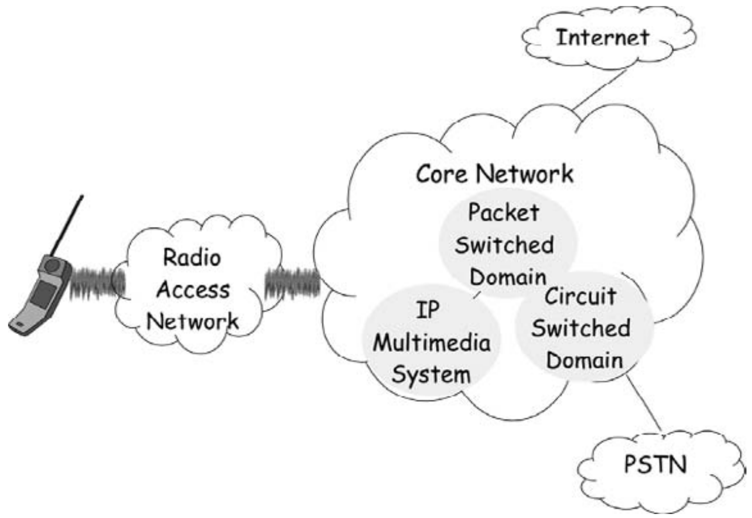


Figure 12.1: Cellular network architecture.

The core network is also connected to the Internet. Internet connectivity allows the cellular network to provide innovative multimedia services such as weather reports, stock reports, sports information, chat, and electronic mail. Interworking with the Internet is possible using protocol gateways, federated databases, and multiprotocol mobility managers [2]. Interworking with the Internet has created a new generation of services called *cross-network services*. These are multivendor, multidomain services that use a combination of Internet-based data and data from the cellular network to provide a variety of services to the cellular subscriber. A sample cross-network service is the *Email Based Call Forwarding Service* (CFS), which uses Internet-based email data (in a mail server) to decide on the call-forward number (in a call-forward server) and delivers the call via the cellular network.

From a functional viewpoint, the core network may also be further divided into the circuit-switched (CS) domain, the packet-switched (PS) domain, and the IP Multimedia Subsystem (IMS). In the following, we further discuss the core network organization.

Core Network Organization

Cellular networks are organized as collections of interconnected *network areas*, where each network area covers a fixed geographical region (as shown in Figure 12.2). Every subscriber is affiliated with two networks: the *home network* and the *visiting network*.

Every subscriber is permanently assigned to the home network, from which they can roam onto other visiting networks. The home network maintains the subscriber profile and current subscriber location. The visiting network is the network where the subscriber is currently roaming. It provides radio resources, mobility management, routing, and services for roaming subscribers. The visiting network provides service capabilities to the subscribers on behalf of the home environment [3].

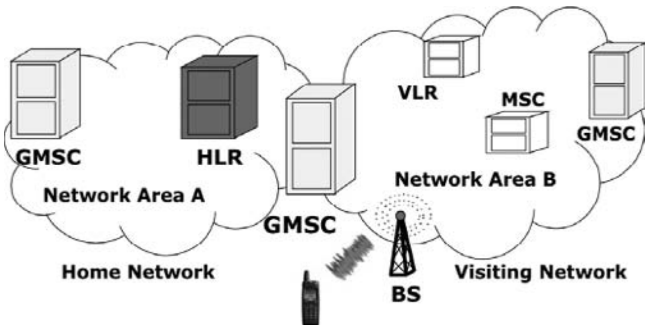


Figure 12.2: Core network organization.

The core network is facilitated by network servers (also called *service nodes*). Service nodes are composed of (1) a variety of *data sources* (such as cached read-only, updateable, and shared data sources) to store data such as subscriber profile and (2) *service logic* to perform functions such as computing data items, retrieving data items from data sources, and so on.

Service nodes can be of different types, with each type assigned specific functions. The major service node types in the circuit-switched domain include the Home Location Register (HLR), the Visitor Location Register (VLR), the Mobile Switching Center (MSC), and the Gateway Mobile Switching Center (GMSC) [4].

All subscribers are permanently assigned to a fixed HLR located in the home network. The HLR stores permanent subscriber profile data and relevant temporary data such as current subscriber location (pointer to VLR) of all subscribers assigned to it. Each network area is assigned a VLR. The VLR stores temporary data of subscribers currently roaming in its assigned area; this subscriber data is received from the HLR of the subscriber. Every VLR is always associated with an MSC. The MSC acts as an interface between the radio access network and the core network. It also handles circuit-switched services for subscribers currently roaming in its area. The GMSC is in charge of routing the call to the actual location of the mobile station. Specifically, the GMSC acts as interface between the fixed PSTN network and the cellular network. The radio access network comprises a transmitter, receiver, and speech transcoder called the base station (BS) [5].

Service nodes are geographically distributed and service the subscriber through collaborative functioning of various network components. Such collaborative functioning is possible due to the network relationships (called dependencies). A *dependency* means that a network component must rely on other network components to perform a function. For example, there is a *dependency* between service nodes to service subscribers. Such a dependency is made possible through signaling messages containing data items. Service nodes typically request other service nodes to perform specific operations by sending them signaling messages containing data items with predetermined values. On receiving signaling messages, service nodes realize the operations to perform based on values of data items received in signaling messages. Further, dependencies may exist between data items so that received data items may be used to derive other data items. Several application layer protocols are used for signaling messages. Examples of signaling message protocols include Mobile Application Part (MAP), ISDN User Part (ISUP), and Transaction Capabilities Application Part (TCAP) protocols.

Typically in the cellular network, to provide a specific service a preset group of signaling messages is exchanged between a preset group of service node types. The preset group of signaling messages indicates the operations to be performed at the various service nodes and is called a *signal flow*. In the following, we use the *call delivery service* [6] to illustrate a signal flow and show how the various geographically distributed service nodes function together.

Call Delivery Service

The *call delivery service* is a basic service in the circuit-switched domain. It is used to deliver incoming calls to any subscriber with a mobile device regardless of their location. The signal flow of the call delivery service is illustrated in Figure 12.3. The call delivery service signal flow comprises MAP messages `SRI, SRI_ACK, PRN`, and `PRN_ACK`; ISUP message `IAM`; and TCAP messages `SIFIC`, `Page MS`, and `Page`.

Figure 12.3 illustrates the exchange of signal messages between different network areas. It shows that when a subscriber makes a call using his mobile device, the call is sent in the form of a signaling message `IAM` to the nearest GMSC, which is in charge of routing calls and passing voice traffic between different networks. This signaling message `IAM` contains data items such as *called number* that denotes the mobile phone number of the subscriber receiving this call. The *called number* is used by the GMSC to locate the address of the HLR (home network) of the called party. The GMSC uses this address to send the signaling message `SRI`.

The `SRI` message is an intimation to the HLR of the arrival of an incoming call to a subscriber with *called number* as mobile phone number. It contains data items such as the *called number* and *alerting pattern*. The *alerting pattern* denotes the pattern (*packet-switched data*, *short message service*, or *circuit-switched call*) used to alert the subscriber receiving the call. The HLR uses the *called number* to retrieve from its database the current location (pointer to VLR) of the subscriber receiving the call. The HLR uses this subscriber location to send the VLR the message `PRN`. The `PRN` message is a request for

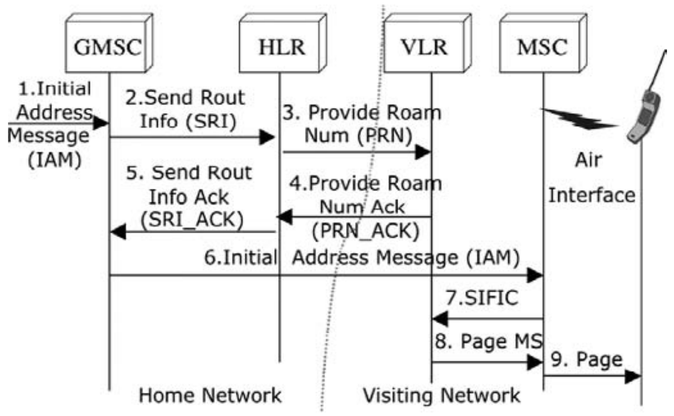

GMSC
HLR
VLR
MSC
1.Initial Address Message (IAM)
2.Send Rout Info (SRI)
3. Provide Roam Num (PRN)
Air Interface
5. Send Rout Info Ack (SRI_ACK)
4.Provide Roam Num Ack (PRN_ACK)
6.Initial Address Message (IAM)
7.SIFIC
8. Page MS
9. Page
Home Network
Visiting Network

Figure 12.3: Signal flow in the call delivery service.

call routing information (also called *roaming number*) from the VLR where the subscriber is currently roaming. The PRN message contains the *called number*, *alerting pattern*, and other *subscriber call profile* data items.

The VLR uses the *called number* to store the *alerting pattern* and *subscriber call profile* data items and assign the *roaming number* for routing the call. This *roaming number* data item is passed on to the HLR (in message PRN_ACK), which forwards it to the GMSC (in message SRI_ACK). The GMSC uses this *roaming number* to route the call (message IAM) to the MSC where the subscriber is currently roaming. On receipt of the message IAM, the MSC assigns the *called number* resources for the call and also requests the *subscriber call profile* data items, and *alerting pattern* for the *called number* (using message SIFIC) from the VLR, and receives the same in the Page MS message. The MSC uses the *alerting pattern* in the incoming call profile to *derive* the *page type* data item. The *page type* data item denotes the manner in which to alert the mobile station. It is used to page the mobile subscriber (using message Page). Thus subscribers receive incoming calls irrespective of their locations in the network.

If data item values are inaccurate, the network can misoperate and subscribers will be affected. Hence, accurate functioning of the network is greatly dependent on the integrity of data item values. Thus signal flows allow the various service nodes to function together, ensuring that the network services its subscribers effectively.

3. The State of the Art of Cellular Network Security

This part of the chapter presents the current state of the art of cellular network security. Because the security of the cellular network is the security of each aspect of the network, that is, radio access network, core network, Internet connection, and PSTN connection, we detail the security of each in detail.

Security in the Radio Access Network

The radio access network uses radio signals to connect the subscriber's cellular device with the core wireline network. Hence it would seem that attacks on the radio access network could easily happen because anyone with a transmitter/receiver could capture these signals. This was very true in the case of early-generation cellular networks (first and second generations), where there were no guards against eavesdropping on conversations between the cellular device and BS; cloning of cellular devices to utilize the network resources without paying; and cloning BSs to entice users to camp at the cloned BS in an attack called a *false base station attack*, so that the target user provides secret information to the adversary.

In the current generation (third-generation) cellular network, all these attacks can be prevented because the network provides adequate security measures. Eavesdropping on signals between the cellular device and BS is not possible, because cipher keys are used to encrypt these signals. Likewise, replay attacks on radio signals are voided by the use of nonrepeated random values. Use of integrity keys on radio conversations voids the possibility of deletion and modification of conversations between cellular devices and BSs. By allowing the subscriber to authenticate the network, and vice versa, this generation voids the attacks due to cloned cellular devices and BSs. Finally, as the subscriber's identity is kept confidential by only using a temporary subscriber identifier on the radio network, it is also possible to maintain subscriber location privacy [7].

However, the current generation still cannot prevent a denial-of-service attack from occurring if a large number of registration requests are sent via the radio access network (BS) to the visiting network (MSC). Such a DoS attack is possible because the MSC cannot realize that the registration requests are fake until it attempts to authenticate each request and the request fails. To authenticate each registration request, the MSC must fetch the authentication challenge material from the corresponding HLR. Because the MSC is busy fetching the authentication challenge material, it is kept busy and the genuine registration requests are lost [8]. Overall there is a great improvement in the radio network security in the current third-generation cellular network.

Security in Core Network

Though the current generation network has seen many security improvements in the radio access network, the security of the core network is not as improved. Core network security is the security at the service nodes and security on links (or wireline signaling message) between service nodes.

With respect to wireline signaling message security, of the many wireline signaling message protocols, protection is only provided for the Mobile Application Part (MAP) protocol. The MAP protocol is the cleartext application layer protocol that typically runs on the security-free SS7 protocol or the IP protocol. MAP is an essential protocol and it is primarily used for message exchange involving subscriber location management, authentication, and call handling. The reason that protection is provided for only the MAP protocol is that it carries authentication material and other subscriber-specific confidential data; therefore, its security was considered top priority and was standardized [9–11]. Though protection for other signaling message protocols was also considered important, it was left as an improvement for the next-generation network [12].

Security for the MAP protocol is provided in the form of the newly proposed protocol called Mobile Application Part Security (MAPSec) when MAP runs on the SS7 protocol stack, or Internet Protocol Security (IPSec) when MAP runs on the IP protocol.

Both MAPSec and IPSec, protect MAP messages on the link between service nodes by negotiating security associations. Security associations comprise keys, algorithms, protection profiles, and key lifetimes used to protect the MAP message. Both MAPSec and IPSec protect MAP messages by providing source service node authentication and message encryption to prevent eavesdropping, MAP corruption, and fabrication attacks.

It must be noted that though MAPSec and IPSec are deployed to protect individual MAP messages on the link between service nodes, signaling messages typically occur as a group in a signal flow, and hence signaling messages must be protected not only on the link but also in the intermediate service nodes. Also, the deployment of MAPSec and IPSec is optional; hence if any service provider chooses to omit MAPSec/IPSec's deployment, the efforts of all other providers are wasted. Therefore, to completely protect MAP messages, MAPSec/IPSec must be used by every service provider.

With respect to wireline service nodes, while MAPSec/IPSec protects links between service nodes, there is no standardized method for protecting service nodes [13]. Remote and physical access to service nodes may be subject to operator's security policy and hence could be exploited (insider or outsider) if the network operator is lax with security. Accordingly, the network suffers from the possibility of node impersonation, corruption of data sources, and service logic attacks. For example, unauthorized access to the HLR could deactivate customers or activate customers not seen by the building system. Similarly, unauthorized access to the MSC could cause outages for a large number of users in a given network area.

Corrupt data sources or service logic in service nodes have the added disadvantage of propagating this corruption to other service nodes in the network [14–16] via signaling messages. This fact was recently confirmed by a security evaluation of cellular networks [17] that showed the damage potential of a compromised service node to be much greater than the damage potential of compromised signaling messages. Therefore, it is of utmost importance to standardize a scheme for protecting service nodes in the interest of not only preventing node impersonation attacks but also preventing the corruption from propagating to other service nodes.

In brief, the current generation core network is lacking in security for all types of signaling messages, for MAP signaling messages in service nodes, and a standardized method for protecting service nodes. To protect all types of signaling message protocols and ensure that messages are secured not only on the link between service nodes but also on the intermediate service nodes (that is, secured end to end), and prevent service logic corruption from propagating to other service nodes, the End-to-End Security (EndSec) protocol was proposed [18].

Because signaling message security essentially depends on security of data item values contained in these messages, EndSec focuses on securing data items. EndSec requires every data item to be signed by its source service nodes using public key encryption. By requiring signatures, if data items are corrupt by compromised intermediate service nodes en route, the compromised status of the service node is revealed to the service nodes receiving the corrupt data items. Revealing the compromised status of service nodes prevents corruption from propagating to other service nodes, because service nodes are unlikely to accept corrupt data items from compromised service nodes.

EndSec also prevents misrouting and node impersonation attacks by requiring every service node in a signal flow to embed the `PATH` taken by the signal flow in every EndSec message. Finally, EndSec introduces several control messages to handle and correct the detected corruption. Note that EndSec is not a standardized protocol.

Security Implications of Internet Connectivity

Internet connectivity introduces the biggest threat to the security of the cellular network. This is because cheap PC-based equipment with Internet connectivity can now access gateways connecting to the core network. Therefore, any attack possible in the Internet can now filter into the core cellular network via these gateways. For example, Internet connectivity was the reason for the slammer worm to filter into the E-911 service in Bellevue, Washington, making it completely unresponsive [19]. Other attacks that can filter into the core network from the Internet include spamming and phishing of short messages [20].

We expect low-bandwidth DoS attacks to be the most damaging attacks brought on by Internet connectivity [21–23]. These attacks demonstrate that by sending just 240 short messages per second, it is possible to saturate the cellular network and cause the MSC in charge of the region to be flooded and lose legitimate short messages per second. Likewise, it shows that it is possible to cause a specific user to lose short messages by flooding that user with a large number of messages, causing a buffer overflow. Such DoS attacks are possible because the short message delivery time in the cellular network is much greater than the short message submission time using Internet sites [24].

Also, short messages and voices services use the same radio channel, so contention for these limited resources may still occur and cause a loss of voice service. To avoid loss of voice services due to contention, separation of voice and data services on the radio network has been suggested [16]. However, such separation requires major standardization and overhaul of the network and is therefore unlikely be implemented very soon. Other minor techniques such as queue management and resource provisioning have been suggested [25].

Though such solutions could reduce the impact of short message flooding, they cannot eliminate other types of low-bandwidth, DoS attacks such as attacks on connection setup and teardown of data services. The root cause for such DoS attacks from the Internet to the core network was identified as the difference in the design principles of these networks. Though the Internet makes no assumptions on the content of traffic and simply passes it on to the next node, the cellular network identifies the traffic content and provides a highly tailored service involving multiple service nodes for each type of traffic [26].

Until this gap is bridged, such attacks will continue, but bridging the gap itself is a major process because either the design of the cellular network must be changed to match the Internet design, or vice versa, which is unlikely to happen soon. Hence a temporary fix would be to secure the gateways connecting the Internet and core network. As a last note, Internet connectivity filters attacks not only into the core network, but also into the PSTN network. Hence PSTN gateways must also be guarded.

Security Implications of PSTN Connectivity

PSTN connectivity to the cellular network allows calls between the fixed and cellular networks. Though the PSTN was a closed network, the security-free SS7 protocol stack on which it is based was of no consequence. However, by connecting the PSTN to the core network that is in turn connected to the Internet, the largest open public network, the SS7-based PSTN network has "no security left" [27].

Because SS7 protocols are plaintext and have no authentication features, it is possible to introduce fake messages, eavesdrop, cause DoS by traffic overload, and incorrectly route signaling messages. Such introduction of SS7 messages into the PSTN network is very easily done using inexpensive PC-based equipment. Attacks in which calls for 800 and 900 numbers were rerouted to 911 servers so that legitimate calls were lost are documented [28]. Such attacks are more so possible due to the IP interface of the PSTN service nodes and Web-based control of these networks.

Because PSTN networks are to be outdated soon, there is no interest in updating these networks. So, they will remain "security free" until their usage is stopped [29].

So far, we have addressed the security and attacks on each aspect of the cellular network. But an attack that is common to all the aspects of the cellular network is the *cascading attack.* Next we detail the cascading attack and present vulnerability assessment techniques to identify the same.

4. Cellular Network Attack Taxonomy

In this part of the chapter, we present the cellular network specific attack taxonomy. This attack taxonomy is called the *three-dimensional taxonomy* because attacks are classified based on the following three dimensions: (1) adversary's physical access to the network when the attack is launched; (2) type of attack launched; and (3) vulnerability exploited to launch the attack.

The three-dimensional attack taxonomy was motivated by the *cellular network specific abstract model*, which is an atomic model of cellular network service nodes. It enabled better study of interactions within the cellular network and aided in derivation of several insightful characteristics of attacks on the cellular network.

The abstract model not only led to the development of the three-dimensional attack taxonomy that has been instrumental in uncovering (1) *cascading attacks*, a type of attack in which the adversary targets a specific network location but attacks another location, which in turn propagates the attack to the target location, and (2) *cross-infrastructure cyber attack*, a new breed of attack in which the cellular network may be attacked from the Internet [30]. In this part of the chapter we further detail the three-dimensional attack taxonomy and cellular network abstract model.

Abstract Model

The abstract model dissects functionality of the cellular network to the basic atomic level, allowing it to systematically isolate and identify vulnerabilities. Such identification of vulnerabilities allows attack classification based on vulnerabilities, and isolation of network functionality aids in extraction of interactions between network components, thereby revealing new vulnerabilities and attack characteristics.

Because service nodes in the cellular network comprise sophisticated *service logic* that performs numerous network functions, the abstract model logically divides the service logic into basic atomic units, called *agents* (represented by the elliptical shape in Figure 12.4). Each agent performs a single function. Service nodes also manage data, so the abstract model also logically divides data sources into data units specific to the agents they support. The abstract model also divides the data sources into *permanent* (represented by the rectangular shape in Figure 12.4) or *cached* (represented by the triangular shape in Figure 12.4) from other service nodes.

The abstract model developed for the CS domain is illustrated in Figure 12.4. It shows agents, permanent, and cached data sources for the CS service nodes. For example, the *subscriber locator agent* in the HLR is the agent that tracks the subscriber location information. It receives and responds to location requests during an incoming call and stores a subscriber's location every time they move. This location information is stored in the *location data source*. Readers interested in further details may refer elsewhere [31, 32].

Abstract Model Findings

The abstract model had lead to many interesting findings. We outline them as follows.

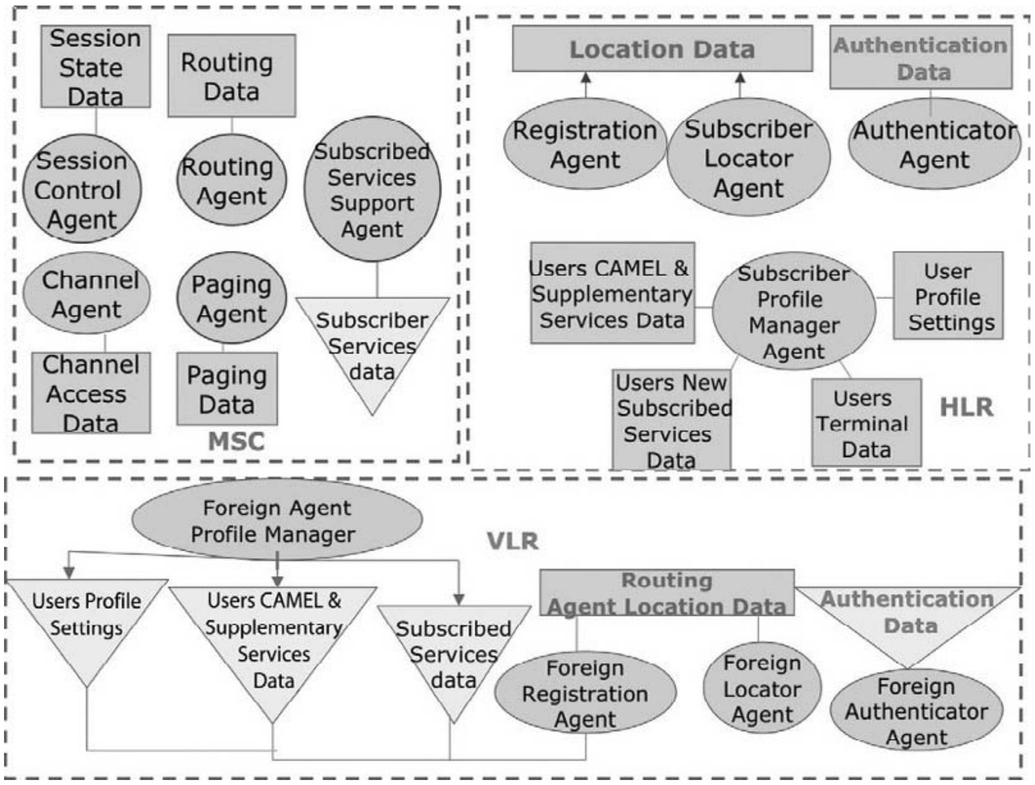


Figure 12.4: Abstract model of circuit-switched service nodes.

Interactions

To study the network interactions, service nodes in signal flows (e.g., call delivery service) were replaced by their corresponding abstract model agents and data sources. Such an abstract model-based signal flow based on the call delivery service is shown in Figure 12.5.

In studying the abstract model signal flow, it was observed that interactions happen (1) between agents typically using procedure calls containing data items; (2) between agents and data sources using queries containing data items; and (3) between agents belonging to different service nodes using signaling messages containing data items.

The common behavior in all these interactions is that they typically involve *data items* whose values are set or modified in agents or data source, or it involves data items passed between agents, data sources, or agents and data sources. Hence, the value of a data item not only can be corrupt in an agent or data source, it can also be easily passed on to other agents, resulting in propagation of corruption. This propagation of corruption is called the *cascading effect*, and attacks that exhibit this effect are called *cascading attacks*. In the following, we present a sample of the cascading attack.

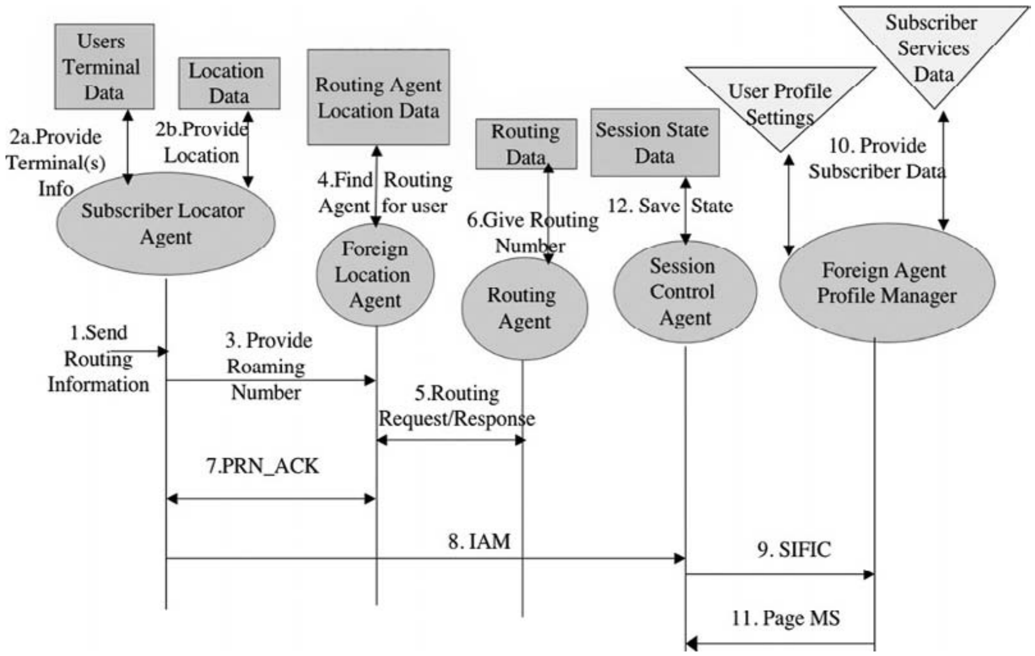


Figure 12.5: Abstract model-based signal flow for the call delivery service.

Sample Cascading Attack

In this sample cascading attack, cascading due to corrupt data items and ultimately their service disruption are illustrated in Figure 12.6. Consider the call delivery service explained previously. Here the adversary may corrupt the *roaming number* data item (used to route the call) in the VLR. This corrupt *roaming number* is passed on in message `PRN_ACK` to the HLR, which in turn passes this information to the GMSC. The GMSC uses the incorrect *roaming number* to route the call to the incorrect MSC_B, instead of the correct MSC_A. This results in the caller losing the call or receiving a wrong-number call. Thus corruption cascades and results in service disruption.

The type of corruption that can cascade is *system-acceptable incorrect value corruption*, a type of corruption in which corrupt values taken on system-acceptable values, albeit incorrect values. Such a corruption can cause the roaming number to be incorrect but a system-acceptable value.

Note that it is easy to cause such system-acceptable incorrect value corruption due to the availability of Web sites that refer to proprietary working manuals of service nodes such as the VLR [33, 34]. Such command insertion attacks have become highly commonplace, the most infamous being the telephone tapping of the Greek government and top-ranking civil servants [35].

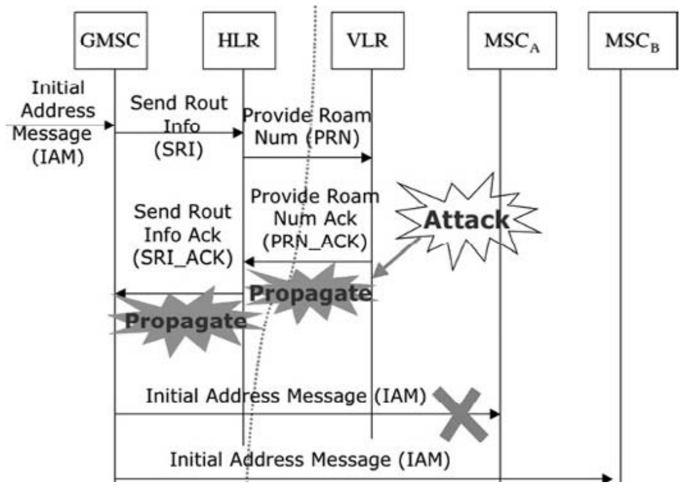

GMSC
HLR
VLR
MSC$_A$
MSC$_B$
Initial Address Message (IAM)
Send Rout Info (SRI)
Provide Roam Num (PRN)
Provide Roam Num Ack (PRN_ACK)
Send Rout Info Ack (SRI_ACK)
Attack
Propagate
Propagate
Initial Address Message (IAM)
Initial Address Message (IAM)

Figure 12.6: Sample cascading attacks in the call delivery service.

Cross-Infrastructure Cyber Cascading Attacks

When cascading attacks cross into the cellular networks from the Internet through *cross-network services*, they're called *cross-infrastructure cyber cascading attacks*. This attack is illustrated on the CFS in Figure 12.7.

As the CFS forwards calls based on the emails received, corruption is shown to propagate from the mail server to a call-forward (CF) server and finally to the MSC. In the attack, using any standard mail server vulnerabilities, the adversary may compromise the mail server and corrupt the email data source by deleting emails from people the victim is expecting to call. The CF server receives and caches incorrect email from the mail server.

When calls arrive for the subscriber, the call-forwarding service is triggered, and the MSC queries the CF server on how to forward the call. The CF server checks its incorrect email cache, and because there are no emails from the caller, it responds to the MSC to forward the

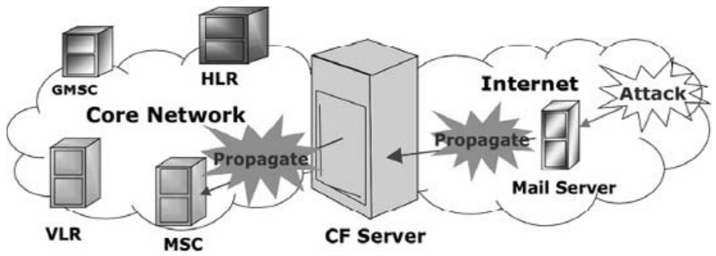


Figure 12.7: Cross-infrastructure cyber cascading attacks on call-forward service.

call to the victim's voicemail when in reality the call should have been forwarded to the cellular device. Thus the effect of the attack on the mail server propagates to the cellular network service node. This is a classic example of a cross-infrastructure cyber cascading attack, whereby the adversary gains access to the cross-network server, and attacks by modifying data in the data source of the cross-network server.

Note that it has become highly simplified to launch such attacks due to easy accessibility to the Internet and subscriber preference for Internet-based cross-network services.

Isolating Vulnerabilities

From the abstract model, the major network components that are vulnerable to adversaries are (1) data sources, (2) agents (more generally called service logic), and (3) signaling messages. By exploiting each of these vulnerabilities, data items that are crucial to the correct working of the network can be corrupted, leading to ultimate service disruption through cascading effects.

In addition, the effect of corrupt signaling messages is different from the effect of corrupt data sources. By corrupting data items in a data source of a service node, all the subscribers attached to this service node may be affected. However, by corrupting a signaling message, only the subscribers (such as the caller and called party in case of call delivery service) associated with the message are affected. Likewise, corrupting the agent in the service node can affect all subscribers using the agent in the service node. Hence, in the three-dimensional taxonomy, a vulnerability exploited is considered as an attack dimension, since the effect on each vulnerability is different.

Likewise, the adversary's physical access to the network also affects how the vulnerability is exploited and how the attack cascades. For example, consider the case when a subscriber has access to the air interface. The adversary can only affect messages on the air interface. Similarly, if the adversary has access to a service node, the data sources and service logic may be corrupted. Hence, in the three-dimensional taxonomy, the physical access is considered a category as it affects how the vulnerability is exploited and its ultimate effect on the subscriber.

Finally, the way the adversary chooses to launch an attack ultimately affects the service in a different way. Consider a passive attack such as *interception*. Here the service is not affected, but it can have a later effect on the subscriber, such as identity theft or loss of privacy. An active attack such as *interruption* can cause complete service disruption. Hence, in the three-dimensional taxonomy, the attack means are considered a category due the ultimate effect on service.

In the next part of the chapter, we detail the cellular network specific three-dimensional taxonomy and the way the previously mentioned dimensions are incorporated.

Three-Dimensional Attack Taxonomy

The three dimensions in the taxonomy include Dimension I: Physical Access to the Network, Dimension II: Attack Categories, and Dimension III: Vulnerability Exploited. In the following, we outline each dimension.

Dimension I: Physical Access to the Network

In this dimension, attacks are classified based on the adversary's level of physical access to the cellular network. Dimension I may be further classified into *single infrastructure attacks* (Levels I–III) and *cross-infrastructure cyber attacks* (Levels IV and V):

Level I: Access to air interface with physical device. Here the adversary launches attacks via access to the radio access network using standard inexpensive "off-the-shelf" equipment [36]. Attacks include false base station attacks, eavesdropping, and man-in-the-middle attacks and correspond to attacks previously mentioned.

Level II: Access to links connecting core service nodes. Here the adversary has access to links connecting to core service nodes. Attacks include disrupting normal transmission of signaling messages and correspond to message corruption attacks previously mentioned.

Level III: Access core service nodes. In this case, the adversary could be an insider who managed to gain physical access to core service nodes. Attacks include editing the service logic or modifying data sources, such as subscriber data (profile, security and services) stored in the service node and corresponding to corrupt service logic, data source, and node impersonation attacks previously mentioned.

Level IV: Access to links connecting the Internet and the core network service nodes. This is a cross-infrastructure cyber attack. Here the adversary has access to links connecting the core network and Internet service nodes. Attacks include editing and deleting signaling messages between the two networks. This level of attack is easier to achieve than Level II.

Level V: Access to Internet servers or cross-network servers: This is a cross-infrastructure cyber attack. Here the adversary can cause damage by editing the service logic or modifying subscriber data (profile, security and services) stored in the cross-network servers. Such an attack was previously outlined earlier in the chapter. This level of attack is easier to achieve than Level III.

Dimension II: Attack Type

In this dimension, attacks are classified based on the type of attack. The attack categories are based on Stallings [37]:

Interception. The adversary intercepts signaling messages on a cable (Level II) but does not modify or delete them. This is a passive attack. This affects the privacy of the subscriber and the network operator. The adversary may use the data obtained from interception to analyze traffic and eliminate the competition provided by the network operator.

Fabrication or replay. In this case, the adversary inserts spurious messages, data, or service logic into the system, depending on the level of physical access. For example, in a Level II, the adversary inserts fake signaling messages, and in a Level III, the adversary inserts fake service logic or fake subscriber data into this system.

Modification of resources. Here the adversary modifies data, messages, or service logic. For example, in a Level II, the adversary modifies signaling messages on the link, and in a Level III, the adversary modifies service logic or data.

Denial of service. In this case, the adversary takes actions to overload the network results in legitimate subscribers not receiving service.

Interruption. Here the adversary causes an interruption by destroying data, messages, or service logic.

Dimension III: Vulnerability exploited

In this dimension, attacks are classified based on the vulnerability exploited to cause the attack. Vulnerabilities exploited are explained as follows.

Data. The adversary attacks the data stored in the system. Damage is inflicted by modifying, inserting, and deleting the data stored in the system.

Messages. The adversary adds, modifies, deletes, or replays signaling messages.

Service logic. Here the adversary inflicts damage by attacking the service logic running in the various cellular core network service nodes.

Attack classification. In classifying attacks, we can group them according to *Case 1: Dimension I versus Dimension II*, and *Case 2: Dimension II versus Dimension III*. Note that the Dimension I versus Dimension III case can be transitively inferred from Case 1 and Case 2.

Table 12.1 shows a sample tabulation of Level 1 attacks grouped in Case 1. For example, with Level 1 access an adversary causes interception attacks by observing traffic and eavesdropping. Likewise, fabrication attacks due to Level I include sending spurious registration messages. Modification of resources due to Level 1 includes modifying conversations in the radio access network. DoS due to Level 1 occurs when a large number of fake registration messages are sent to keep the network busy so as to not provide service to legitimate subscribers. Finally, interruption attacks due to Level 1 occur when adversaries jam the radio access channel so that legitimate subscribers cannot access the network. For further details on attack categories, refer to Kotapati [38].

Table 12.1: Sample Case 1 Classification

	Interception	Fabrication/ Insertion	Modification of Resources	Denial of Service	Interruption
Level I	Observe time, rate, length, source, and destination of victim's locations. With modified cellular devices, eavesdrop on victim.	Using modified cellular devices, the adversary can send spurious registration messages to the target network. Likewise, using modified base stations, the adversary can signal victims to camp at their locations.	With a modified base station and cellular devices, the adversary modifies conversations between subscribers and their base stations.	The adversary can cause DoS by sending a large number of fake registration messages.	Jam victims' traffic channels so that victims cannot access the channels. Broadcast at a higher intensity than allowed, thereby hogging the bandwidth.

5. Cellular Network Vulnerability Analysis

Regardless of how attacks are launched, if attack actions cause a system-acceptable incorrect value corruption, the corruption propagates, leading to many unexpected cascading effects. To detect remote cascading effects and identify the origin of cascading attacks, cellular network vulnerability assessment tools were developed.

These tools, the *Cellular Network Vulnerability Assessment Toolkit* (CAT) and the *advanced Cellular Network Vulnerability Assessment Toolkit* (aCAT) [39, 40] allow users to input the data item that might be corrupted and output an attack graph. The CAT attack graph not only shows the network location and the network service where the corruption might originate, it also shows the various messages and service nodes through which the corruption propagates.

An attack graph is a diagrammatic representation of an attack on a real system. It shows various ways an adversary can break into a system or cause corruption and the various ways in which the corruption may propagate within the system. Attack graphs are typically produced manually by red teams and used by systems administrators for protection. CAT and aCAT attack graphs allow users to trace the effect of an attack through the network and determine its side effects, thereby making them the ultimate service disruption.

The cellular network is at the nascent stage of development with respect to security, so it is necessary to evaluate security protocols before deploying them. Hence, aCAT can be extended with security protocol evaluation capabilities into a tool [41] called *Cellular Network Vulnerability Assessment Toolkit for evaluation* (eCAT). eCAT allows users to quantify the benefits of security solutions by removing attack effects from attack graphs based on the defenses provided. One major advantage of this approach is that solutions may be evaluated before expensive development and deployment.

It must be noted that developing such tools—CAT, aCAT, and eCAT—presented many challenges: (1) cellular networks are extremely complex systems; they comprise several types of service nodes and control protocols, contain hundreds of data elements, and support hundreds of services; hence developing such toolkits requires in-depth working knowledge of these systems; and (2) every deployment of the cellular network comprises a different physical configuration; toolkits must be immune to the diversity in physical configuration; and finally (3) attacks cascade in the network due to regular network activity as a result of dependencies; toolkits must be able to track the way that corruption cascades due to network dependencies.

The challenge of in-depth cellular network knowledge was overcome by incorporating the toolkits with cellular network specifications defined by the Third Generation Partnership Project (3GPP) and is available at no charge [42]. The 3GPP is a telecommunications standards body formed to produce, maintain, and develop globally applicable "technical specifications and technical reports" for a third-generation mobile system based on evolved GSM core networks and the radio access technologies that they support [43].

Usage of specifications allows handling of the diversity of physical configuration, as they detail the functional behavior and not the implementation structure of the cellular network. Specifications are written using simple flow-like diagrams called the Specification and Description Language (SDL) [44] and are referred to as *SDL specifications*. Equipment and service providers use these SDL specifications as the basis for their service implementations.

Corruption propagation is tracked by incorporating the toolkits with novel dependency and propagation models to trace the propagation of corruption. Finally, Boolean properties are superimposed on the propagation model to capture the impact of security solutions.

CAT is the first version of the toolkit developed for cellular network vulnerability assessment. CAT works by taking user input of *seeds* (data items directly corrupted by the adversary and the cascading effect of which leads to a goal) and *goals* (data parameters that are derived incorrectly due to the direct corruption of seeds by the adversary) and uses SDL specification to identify cascading attacks. However, SDL is limited in its expression of relationships and inexplicit in its assumptions and hence cannot capture all the dependencies; therefore CAT misses several cascading attacks.

To detect a complete set of cascading effects, CAT was enhanced with new features, to aCAT. The new features added to aCAT include (1) a network dependency model that explicitly specifies the exact dependencies in the network; (2) infection propagation rules that identify the reasons that cause corruption to cascade; and (3) a small amount of expert knowledge. The network dependency model and infection propagation rules may be applied to SDL specifications and help alleviate their limited expression capability. The expert knowledge helps capture the inexplicit assumptions made by SDL.

In applying these features, aCAT captures all those dependencies that were previously unknown to CAT, and thereby aCAT was able to detect a complete set of cascading effects. Through extensive testing of aCAT, several interesting attacks were found and the areas where SDL is lacking was identified.

To enable evaluation of new security protocols, aCAT was extended to eCAT. eCAT uses Boolean probabilities in attack graphs to detect whether a given security protocol can eliminate a certain cascading effect. Given a security protocol, eCAT can measure effective coverage, identify the types of required security mechanisms to protect the network, and identify the most vulnerable network areas. eCAT was also used to evaluate MAPSec, the new standardized cellular network security protocol. Results from MAPSec's evaluation gave insights into MAPSec's performance and the network's vulnerabilities. In the following, we detail each toolkit.

Cellular Network Vulnerability Assessment Toolkit

In this part of the chapter, we present an overview of CAT and its many features. CAT is implemented using the Java programming language. It is made up of a number of subsystems (as shown in Figure 12.8). The *knowledge base* contains the cellular network knowledge obtained from SDL specifications. SDL specifications contain simple flowchart-like diagrams. The flowcharts are converted into data in the *knowledge base*. The *integrated data structure* is similar to that of the knowledge base; it holds intermediate attack graph results.

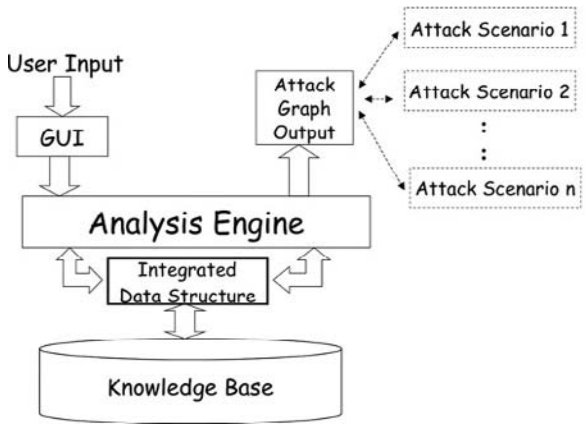


Figure 12.8: Architecture of CAT.

The GUI subsystem takes user input in the form of seeds and goals. The analysis engine contains algorithms (forward and midpoint) incorporated with cascading effect detection rules. It explores the possibility of the user input *seed* leading to the cascading effect of the user input *goal*, using the knowledge base, and outputs the cascading attack in the form of attack graphs.

Using these attack graphs, realistic attack scenarios may be derived. Attack scenarios explain the effect of the attack on the subscriber in a realistic setting. Each attack graph may have multiple interpretations and give rise to multiple scenarios. Each scenario gives a different perspective on how the attack may affect the subscriber.

- *Cascading effect detection rules.* These rules were defined to extract cascading effects from the SDL specifications contained in the knowledge base. They are incorporated into the algorithms in the analysis engine. These rules define what constitutes propagation of corruption from a signaling message to a block, and vice versa, and propagation of corruption within a service node. For example, when a service node receives a signaling message with a corrupt data item and stores the data item, it constitutes propagation of corruption from a signaling message to a block. Note that these rules are high level.
- *Attack graph.* The CAT attack graph may be defined as a state transition showing the paths through a system, starting with the conditions of the attack, followed by attack action, and ending with its cascading effects.

Figure 12.9 presents the CAT attack graph output, which was built using user input of *ISDN Bearer Capability* as a seed and *Bearer Service* as goal. The attack graph constitutes nodes and edges. Nodes represent states in the network with respect to the attack, and *edges* represent network state transitions. For description purposes, each node has been given a node label followed by an alphabet, and the attack graph has been divided into layers.

Nodes may be broadly classified as *conditions, actions,* and *goals,* with the conditions of the attack occurring at the lowest layer and the final cascading effect at the highest layer. In the following, we detail each node type.

- *Condition nodes.* Nodes at the lowest layer typically correspond to the conditions that must exist for the attack to occur. These condition nodes directly follow from the taxonomy. They are an adversary's physical access, target service node, and vulnerability exploited. For example, the adversary has access to links connecting to the GMSC service node, that is, Level II physical access; this is represented as Node A in the attack graph. Likewise, the adversary corrupts data item ISDN Bearer Capability in the IAM message arriving at the GMSC. Hence the target of the attack is the GMSC and is represented by Node B. Similarly, the adversary exploits vulnerabilities in a message (IAM), and this is represented by Node D in the attack graph.

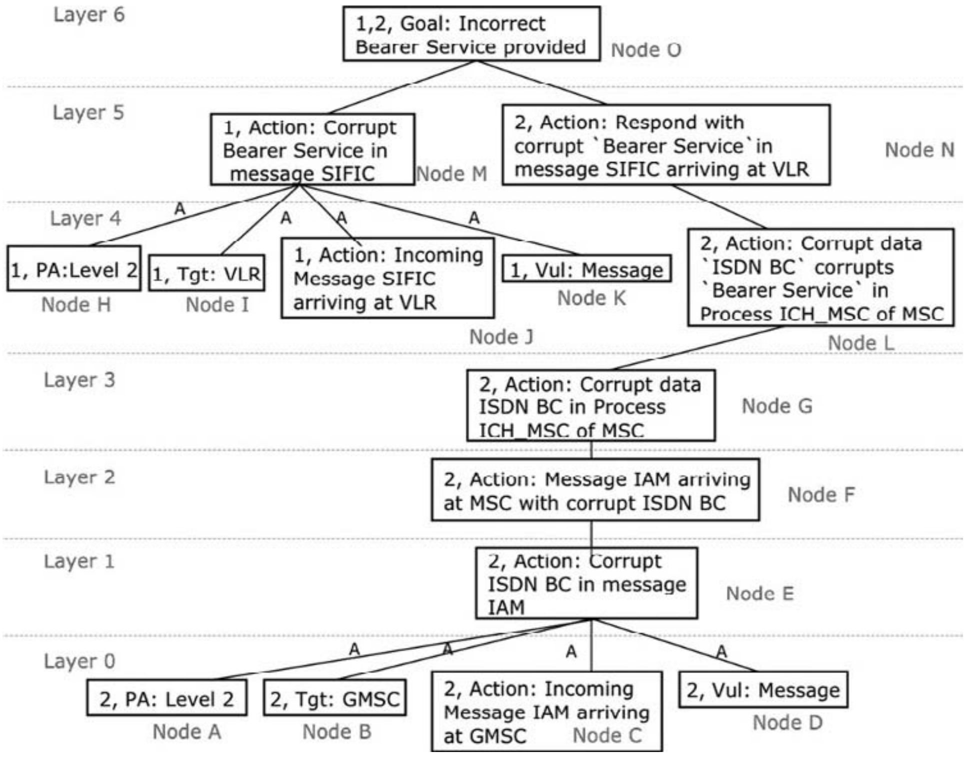

Layer 6
1,2, Goal: Incorrect Bearer Service provided
Node O
Layer 5
1, Action: Corrupt Bearer Service in message SIFIC
Node M
2, Action: Respond with corrupt `Bearer Service` in message SIFIC arriving at VLR
Node N
Layer 4
A
A
A
A
1, PA:Level 2
Node H
1, Tgt: VLR
Node I
1, Action: Incoming Message SIFIC arriving at VLR
Node J
1, Vul: Message
Node K
2, Action: Corrupt data `ISDN BC` corrupts `Bearer Service` in Process ICH_MSC of MSC
Node L
Layer 3
2, Action: Corrupt data ISDN BC in Process ICH_MSC of MSC
Node G
Layer 2
2, Action: Message IAM arriving at MSC with corrupt ISDN BC
Node F
Layer 1
2, Action: Corrupt ISDN BC in message IAM
Node E
Layer 0
A
A
A
A
2, PA: Level 2
Node A
2, Tgt: GMSC
Node B
2, Action: Incoming Message IAM arriving at GMSC
Node C
2, Vul: Message
Node D

Figure 12.9: CAT attack graph output.

- The CAT attack graphs show all the possible conditions for an attack to happen, that is, we see not only the corruption due to the seed ISDN Bearer Capability in signaling message IAM arriving at the GMSC but also other possibilities such as the corruption of goal Bearer Service in signaling message SIFIC, represented by Node M.
- *Action nodes*. Nodes at higher layers are actions that typically correspond to effects of the attack propagating through the network. Effects typically include propagation of corruption between service nodes, such as from MSC to VLR (Node N), propagation of corruption within service nodes such as ISDN Bearer Capability corrupting Bearer Service (Node L), and so on. Actions may further be classified as adversary actions, normal network operations, or normal subscriber activities. Adversary actions include insertion, corruption, or deletion of data, signaling messages, or service logic represented by Node E. Normal network operations include sending (Node N) and receiving signaling messages (Node E). Subscriber activity may include updating personal data or initiating service.
- *Goal nodes*. Goal nodes typically occur at the highest layer of the attack graph. They indicate corruption of the goal items due to the direct corruption of seeds by the adversary (Node A).

- *Edges*. In our graph, edges represent network transitions due to both normal network actions and adversary actions. Edges help show the global network view of adversary action. This is the uniqueness of our attack graph. Transitions due to adversary action are indicated by an edge marked by the letter A (edges connecting Layer 0 and Layer 1). By inclusion of normal network transitions in addition to the transitions caused by the adversary, our attack graph shows not only the adversary's activity but also the *global network view of the adversary's action*. This is a unique feature of the attack graph.
- *Trees*. In the graph, trees are distinguished by the tree numbers assigned to its nodes. For example, all the nodes marked with number 2 belong to Tree 2 of the graph. Some nodes in the graph belong to multiple trees. Tree numbers are used to distinguish between AND and OR nodes in the graph. Nodes at a particular layer with the same tree number(s) are AND nodes. For example, at Layer 4, Nodes H, I, J, and K are AND nodes; they all must occur for Node M at Layer 5 to occur. Multiple tree numbers on a node are called OR nodes. The OR node may be arrived at using alternate ways. For example, Node O at Layer 6 is an OR node, the network state indicated by Node O may be arrived at from Node M or Node N.

Each attack tree shows the attack effects due to corruption of a seed at a specific network location (such as signaling message or process in a block). For example, Tree 1 shows the attack due to the corruption of the seed Bearer Service at the VLR. Tree 2 shows the propagation of the seed ISDN Bearer Capability in the signaling message `IAM`. These trees show that the vulnerability of the cellular network is not limited to one place but can be realized due to the corruption of data in many network locations.

In constructing the attack graph, CAT assumes that an adversary has all the necessary conditions for launching the attack. The CAT attack graph format is well suited to cellular networks because data propagates through the network in various forms during the normal operation of a network; thus an attack that corrupts a data item manifests itself as the corruption of a different data item in a different part of the network after some network operations take place.

- Attack scenario derivation. The CAT attack graph is in cellular network semantics, and realistic attack scenarios may be derived to understand the implications of the attack graph. Here we detail the principles involved in the derivation of realistic attack scenarios.
  1. *End-user effect*. Goal node(s) are used to infer the end effect of the attack on the subscriber. According to the goal node in Figure 12.9, the SIFIC message to the VLR has incorrect goal item Bearer Service. The SIFIC message is used to inform the VLR the calling party's preferences such as voice channel requirements and request the VLR to set up the call based on the calling party and receiving party preferences.

If the calling party's preferences (such as Bearer Service) are incorrect, the call setup by the VLR is incompatible with the calling party, and the communication is ineffective (garbled speech). From the goal node, it can be inferred that Alice, the receiver of the call, is unable to communicate effectively with Bob, the caller, because Alice can only hear garbled speech from Bob's side.

2. *Origin of attack.* Nodes at Layer 0 indicate the origin of the attack, and hence the location of the attack may be inferred. The speech attack may originate at the signaling messages IAM, or the VLR service node.
3. *Attack propagation and side effects.* Nodes at all other layers show the propagation of corruption across the various service nodes in the network. From other layers in Figure 12.9, it can be inferred that the seed is the ISDN Bearer Capability and the attack spreads from the MSC to the VLR.

- Attack scenario. Using these guidelines, an attack scenario may be derived as follows. Trudy, the adversary, corrupts the ISDN Bearer Capability of Bob, the victim, at the IAM message arriving at the GMSC. The GMSC propagates this corruption to the MSC, which computes, and hence corrupts, the Bearer Service. The corrupt Bearer Service is passed on to the VLR, which sets up the call between Bob, the caller, and Alice, the receiver. Bob and Alice cannot communicate effectively because Alice is unable to understand Bob.

Though CAT has detected several cascading attacks, its output to a great extent depends on SDL's ability to capture data dependencies. SDL is limited in its expression capability in the sense that it does not always accurately capture the relationship between data items, and in many cases, SDL does even specify the relationship. Without these details CAT may miss some cascading effects due to loss of data relationships. CAT's output to a minor extent also depends on user input in the sense that to accurately capture all the cascading effect of a seed, the user's input must comprise all the seeds that can occur in the cascading effect; otherwise the exact cascading effect is not captured. To alleviate CAT's inadequacies, aCAT was developed.

Advanced Cellular Network Vulnerability Assessment Toolkit

In this part of the chapter, we present aCAT, an extension of CAT with enhanced features. These enhanced features include (1) incorporating expert knowledge to compensate for the lacking caused by SDL's inexplicit assumptions; expert knowledge added to the knowledge base with the SDL specifications; (2) defining a network dependency model that accurately captures the dependencies in the cellular network; the network dependency model is used to format the data in knowledge base, thereby clarifying the nature of the network dependency; and (3) defining infection propagation rules that define fine-grained rules to detect cascading attacks; these infection propagation rules are incorporated into the analysis engine, which comprises the forward, reverse, and combinatory algorithms. aCAT is also improved in terms of its user input requirements. It requires as input either seeds or goals, whereas CAT required both seeds and goals.

In principle, cascading attacks are the result of propagation of corruption between network components (such as signaling messages, caches, local variables, and service logic) due to dependencies that exist between these network components. Hence, to uncover these attacks, the network dependency model and infection propagation (IP) rules were defined. In the following, we detail the network dependency model and infection propagation model using Figure 12.10.

- *Network dependency model.* The network dependency model accurately defines fine-grained dependencies between the various network components. Given that service nodes comprise agents and data sources (from the abstract model), the dependencies are defined as follows. In interagent dependency, agents communicate with each other using agent invocations (as shown by 6 in Figure 12.10) containing data items. Thus, agents are related to each other through data items. Likewise, in agent to data source dependency, agents communicate with data sources using `Read` and `Write` operations containing data items. Therefore, agents and data items are related to each other through data items. Within agents, derivative dependencies define relationships between data items. Here data items are used as input to derive data items using derivation operations such as AND, OR operations. Therefore, data items are related to each other through derivation operation. For further details on the network dependency model, refer to Kotapati et al. [45].

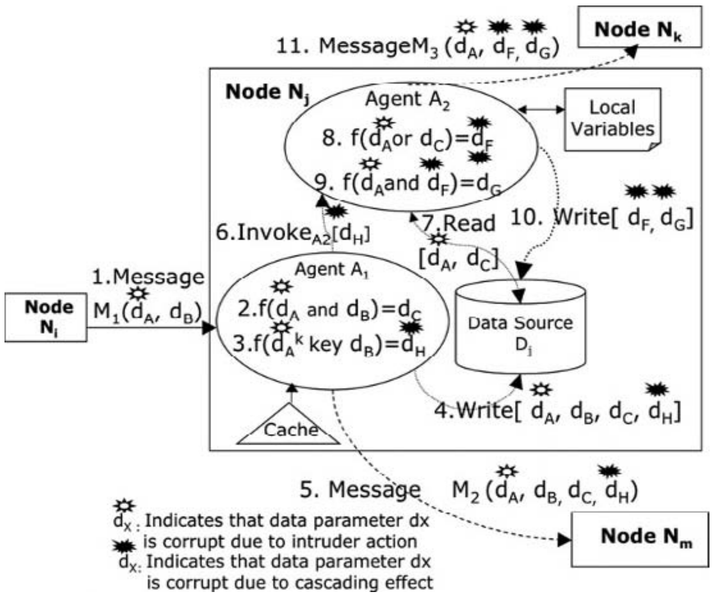


Figure 12.10: Network dependency model.

- *Infection propagation (IP) rules.* These are fine-grained rules to detect cascading effects. They are incorporated into the algorithms in the analysis engine. An example of the IP rule is that an output data item in the AND dependency is corrupt only if both the input data items are corrupt (as shown by 9 in Figure 12.10). Likewise, an output data item in the OR dependency is corrupt if a single input data item is corrupt (as shown by 8 in Figure 12.10). Similarly, corruption propagates between agents when the data item used to invoke the agent is corrupt, and the same data item is used as an input in the derivative dependency whose output may be corrupt (as shown by 6, 8 in Figure 12.10). Accordingly, corruption propagates from an agent to a data source if the data item written to the data source is corrupt (as shown by 4 in Figure 12.10). Finally, corruption propagates between service nodes if a data item used in the signaling message between the service nodes is corrupt, and the corrupt data item is used to derive corrupt output items or the corrupt data item is stored in the data source (as shown by 1, 3 or 1, 4 in Figure 12.10) [46].
  With such a fine-grained dependency model and infection propagation rules, aCAT was very successful in identifying cascading attacks in several of the key services offered by the cellular network, and it was found that aCAT can indeed identify a better set of cascading effects in comparison to CAT. aCAT has also detected several interesting and unforeseen cascading attacks that are subtle and difficult to identify by other means. These newly identified cascading attacks include the alerting attack, power-off/power-on attack, mixed identity attack, call redirection attack, and missed calls attack.
- Alerting attack. In the following we detail aCAT's output, a cascading attack called the alerting attack, shown in Figure 12.11. From goal nodes (Node A at Level 5, and Node C at Level 4) in the alerting attack, it can be inferred that the Page message has incorrect data item page type. The Page message is used to inform subscribers of the arrival of incoming calls, and "page type" indicates the type of call. "Page type" must be compatible with the subscriber's mobile station or else the subscriber is not alerted. From the goal node it may be inferred that Alice, a subscriber of the system, is not alerted on the arrival of an incoming call and hence does not receive incoming calls. This attack is subtle to detect because network administrators find that the network processes the incoming call correctly and that the subscriber is alerted correctly. They might not find that this alerting pattern is incompatible with the mobile station itself.

Also, nodes at Level 0 indicate the origin of the attack as signaling messages `SRI`, `PRN`, the service nodes VLR, or the HLR. From the other levels it may be inferred that the seed is the *alerting pattern* that the adversary corrupts in the `SRI` message and the attack spreads from the HLR to the VLR and from the VLR to the MSC. For more details on these attacks, refer to Kotapati et al. [47].

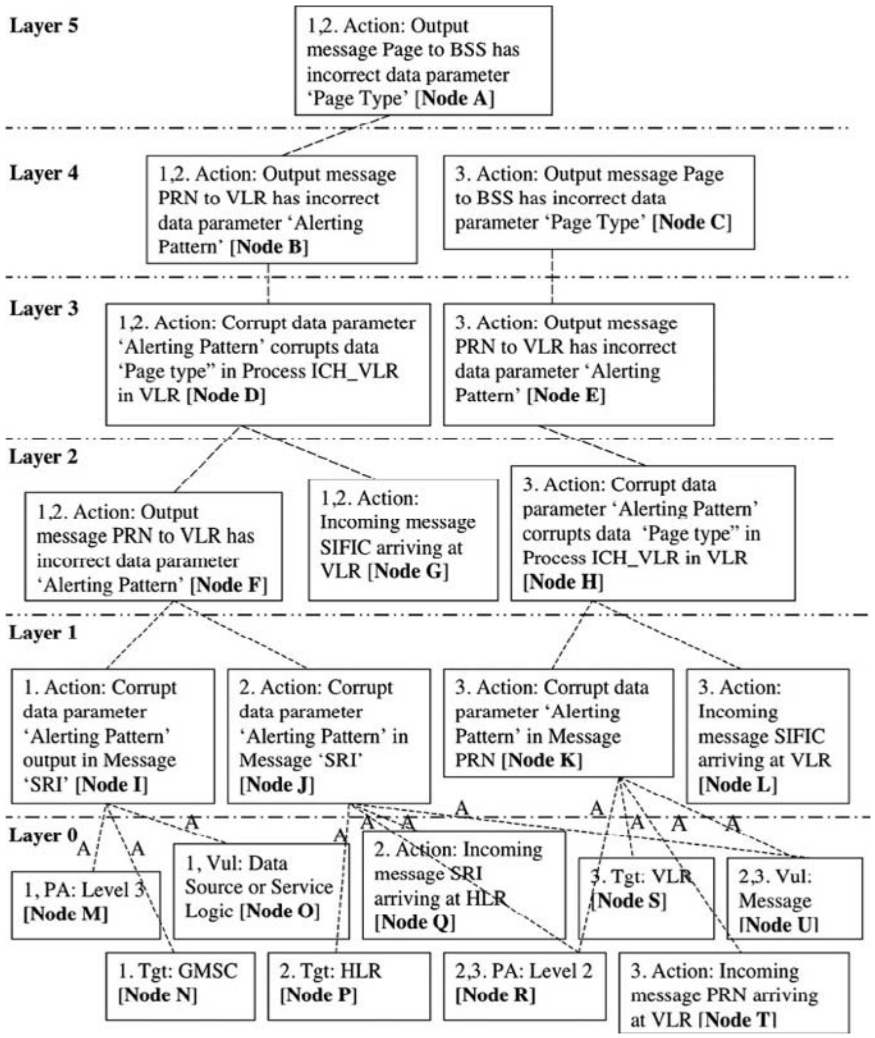

Layer 5
1,2. Action: Output message Page to BSS has incorrect data parameter 'Page Type' [Node A]
Layer 4
1,2. Action: Output message PRN to VLR has incorrect data parameter 'Alerting Pattern' [Node B]
3. Action: Output message Page to BSS has incorrect data parameter 'Page Type' [Node C]
Layer 3
1,2. Action: Corrupt data parameter 'Alerting Pattern' corrupts data 'Page type" in Process ICH_VLR in VLR [Node D]
3. Action: Output message PRN to VLR has incorrect data parameter 'Alerting Pattern' [Node E]
Layer 2
1,2. Action: Output message PRN to VLR has incorrect data parameter 'Alerting Pattern' [Node F]
1,2. Action: Incoming message SIFIC arriving at VLR [Node G]
3. Action: Corrupt data parameter 'Alerting Pattern' corrupts data 'Page type" in Process ICH_VLR in VLR [Node H]
Layer 1
1. Action: Corrupt data parameter 'Alerting Pattern' output in Message 'SRI' [Node I]
2. Action: Corrupt data parameter 'Alerting Pattern' in Message 'SRI' [Node J]
3. Action: Corrupt data parameter 'Alerting Pattern' in Message PRN [Node K]
3. Action: Incoming message SIFIC arriving at VLR [Node L]
Layer 0
A
1, PA: Level 3 [Node M]
1, Vul: Data Source or Service Logic [Node O]
2. Action: Incoming message SRI arriving at HLR [Node Q]
3. Tgt: VLR [Node S]
2,3. Vul: Message [Node U]
1. Tgt: GMSC [Node N]
2. Tgt: HLR [Node P]
2,3. PA: Level 2 [Node R]
3. Action: Incoming message PRN arriving at VLR [Node T]

Figure 12.11: Attack graph for alerting attack.

Cellular Network Vulnerability Assessment Toolkit for Evaluation

In this part of the chapter, we present eCAT, an extension to aCAT. eCAT was developed to evaluate new security protocols before their deployment. Though the design goals and threat model of these security protocols are common knowledge, eCAT was designed to find (1) the effective protection coverage of these security protocols in terms of percentage of attacks prevented; (2) the other kinds of security schemes required to tackle the attacks that can evade the security protocol under observation; and (3) the most vulnerable network areas (also called *hotspots*) [48].

eCAT computes security protocol coverage using attack graphs generated by aCAT and Boolean probabilities in a process called *attack graph marking* and quantifies the coverage using *coverage measurement formulas* (CMF). Attack graph marking also identifies network hotspots and exposes if the security protocol being evaluated protects these hotspots. eCAT was also used to evaluate MAPSec, as it is a relatively new protocol, and evaluation results would aid network operators.

- *Boolean probabilities*. Boolean probabilities are used in attack graphs to distinguish between nodes eliminated (denoted by 0, or shaded node in attack graph) and nodes existing (denoted by 1, or unshaded node in attack graph) due to the security protocol under evaluation. By computing Boolean probabilities for each node in the attack graph, eCAT can extract the attack effects that may be eliminated by the security protocol under evaluation.
- *Attack graph marking*. To mark attack graphs, user input of Boolean probabilities must be provided for Layer 0 nodes. For example, if the security protocol under evaluation is MAPSec, then because MAPSec provides security on links between nodes, it eliminates Level 2 physical access. For example, consider the attack graph generated by eCAT shown in Figure 12.12. Here, Node 5 is set to 0, while all other nodes are set to 1. eCAT uses the input from Layer 0 nodes to compute the Boolean probabilities for the rest of the nodes starting from Layer 1 and moving upward. For example, the Boolean probability of the AND node (Node 18) is the product of all the nodes in the previous layer with the same tree number. Because Node 5 has the same tree number as Node 18, and Node 5's Boolean probability is 0, Node 18's Boolean probability is also 0. This process of marking attack graphs is continued until Boolean probability of all the nodes is computed till the topmost layer.

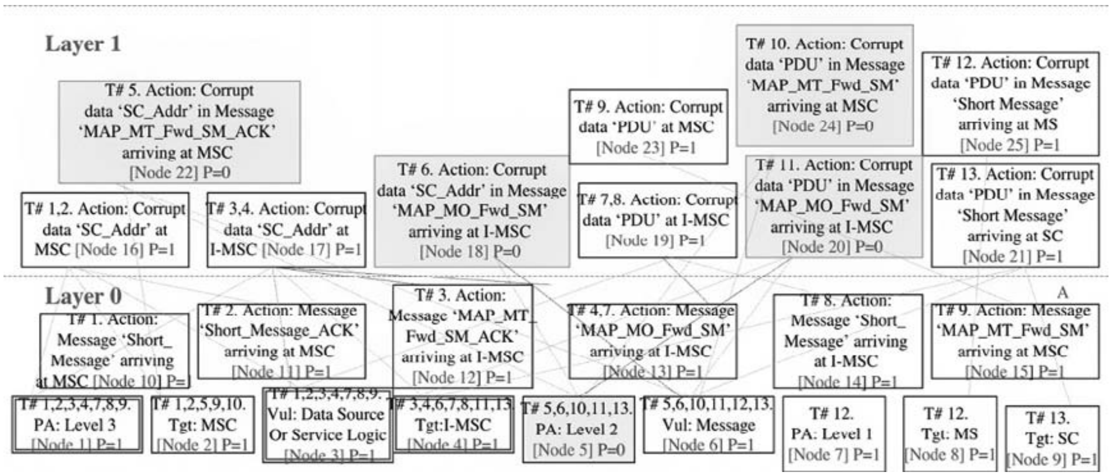

Layer 1
T# 5. Action: Corrupt data 'SC_Addr' in Message 'MAP_MT_Fwd_SM_ACK' arriving at MSC [Node 22] P=0
T# 9. Action: Corrupt data 'PDU' at MSC [Node 23] P=1
T# 10. Action: Corrupt data 'PDU' in Message 'MAP_MT_Fwd_SM' arriving at MSC [Node 24] P=0
T# 12. Action: Corrupt data 'PDU' in Message 'Short Message' arriving at MS [Node 25] P=1
T# 1,2. Action: Corrupt data 'SC_Addr' at MSC [Node 16] P=1
T# 3,4. Action: Corrupt data 'SC_Addr' at I-MSC [Node 17] P=1
T# 6. Action: Corrupt data 'SC_Addr' in Message 'MAP_MO_Fwd_SM' arriving at I-MSC [Node 18] P=0
T# 7,8. Action: Corrupt data 'PDU' at I-MSC [Node 19] P=1
T# 11. Action: Corrupt data 'PDU' in Message 'MAP_MO_Fwd_SM' arriving at I-MSC [Node 20] P=0
T# 13. Action: Corrupt data 'PDU' in Message 'Short Message' arriving at SC [Node 21] P=1
Layer 0
T# 1. Action: Message 'Short_Message' arriving at MSC [Node 10] P=1
T# 2. Action: Message 'Short_Message_ACK' arriving at MSC [Node 11] P=1
T# 3. Action: Message 'MAP_MT_Fwd_SM_ACK' arriving at I-MSC [Node 12] P=1
T# 4,7. Action: Message 'MAP_MO_Fwd_SM' arriving at I-MSC [Node 13] P=1
T# 8. Action: Message 'Short_Message' arriving at I-MSC [Node 14] P=1
T# 9. Action: Message 'MAP_MT_Fwd_SM' arriving at MSC [Node 15] P=1
A
T# 1,2,3,4,7,8,9. PA: Level 3 [Node 1] P=1
T# 1,2,5,9,10. Tgt: MSC [Node 2] P=1
T# 1,2,3,4,7,8,9. Vul: Data Source Or Service Logic [Node 3] P=1
T# 3,4,6,7,8,11,13. Tgt:I-MSC [Node 4] P=1
T# 5,6,10,11,13. PA: Level 2 [Node 5] P=0
T# 5,6,10,11,12,13. Vul: Message [Node 6] P=1
T# 12. PA: Level 1 [Node 7] P=1
T# 12. Tgt: MS [Node 8] P=1
T# 13. Tgt: SC [Node 9] P=1

Figure 12.12: A marked attack graph generated by eCAT.

- *Hotspots*. Graph marking also marks the network hotspots in the attack graph. With respect to the attack graph, hotspots are the Layer 0 nodes with the highest tree number count. For example, in Figure 12.12, Node 3 and Node 4 are the hotspots. A high tree number count indicates an increased attractiveness of the network location to adversaries. This is because by breaking into the network location indicated by the hotspot node, the adversary has a higher likelihood of success and can cause the greatest amount of damage.
  Extensive testing of eCAT on several of the network services using MAPSec has revealed hotspots to be "Data Sources and Service Logic." This is because a corrupt data source or service logic may be used by many different services and hence cause many varied cascading effects, spawning a large number of attacks (indicated by multiple trees in attack graphs). Thus attacks that occur due to exploiting service logic and data source vulnerabilities constitute a major portion of the networkwide vulnerabilities and so a major problem. In other words, by exploiting service logic and data sources, the likelihood of attack success is very high. Therefore data source and service logic protection mechanisms must be deployed. It must be noted that MAPSec protects neither service logic nor data sources; rather, it protects MAP messages.
- *Coverage measurement formulas*. The CMF comprises the following set of three formulas to capture the coverage of security protocols: (1) *effective coverage*, to capture the average effective number of attacks eliminated by the security protocol; the higher the value of Effective Coverage the greater the protection the security protocol; (2) *deployment coverage*, to capture the coverage of protocol deployments; and (3) *attack coverage*, to capture the attack coverage provided by the security protocol; the higher this value, the greater is the security solution's efficacy in eliminating a large number of attacks on the network.

Extensive use of CMF on several of the network services has revealed that MAPSec has an average networkwide attack coverage of 33%. This may be attributed to the fact that message corruption has a low spawning effect. Typically a single message corruption causes a single attack, since messages are typically used by a single service. Hence MAPSec is a solution to a small portion of the total network vulnerabilities.

Finally, in evaluating MAPSec using eCAT, it was observed that though MAPSec is 100% effective in preventing MAP message attacks, it cannot prevent a successfully launched attack from cascading. For MAPSec to be truly successful, every leg of the MAP message transport must be secured using MAPSec. However, the overhead for deploying MAPSec can be high, in terms of both processing load and monetary investment. Also, as MAP messages travel through third-party networks en route to their destinations, the risk level of attacks without MAPSec is very high. Hence, MAPSec is vital to protect MAP messages.

In conclusion, because MAPSec can protect against only 33% of attacks, it alone is insufficient to protect the network. A complete protection scheme for the network must include data source and service logic protection.

6. Discussion

Next to the Internet, the cellular network is the most highly used communication network. It is also the most vulnerable, with inadequate security measures making it a most attractive target to adversaries that want to cause communication outages during emergencies. As the cellular network is moving in the direction of the Internet, becoming an amalgamation of several types of diverse networks, more attention must be paid to securing these networks. A push from government agencies requiring mandatory security standards for operating cellular networks would be just the momentum needed to securing these networks.

Of all the attacks discussed in this chapter, cascading attacks have the most potential to stealthily cause major network misoperation. At present there is no standardized scheme to protect from such attacks. EndSec is a good solution for protecting from cascading attacks, since it requires every data item to be signed by the source service node. Because service nodes are unlikely to corrupt data items and they are to be accounted for by their signatures, the possibility of cascading attacks is greatly reduced. EndSec has the added advantage of providing end-to-end security for all types of signaling messages. Hence, standardizing EndSec and mandating its deployment would be a good step toward securing the network.

Both Internet and PSTN connectivity are the open gateways that adversaries can use to gain access and attack the network. Because the PSTN's security is not going to be improved, at least its gateway to the core network must be adequately secured. Likewise, since neither the Internet's design nor security will be changed to suit the cellular network, at least its gateways to the core network must be adequately secured.

Finally, because the cellular network is an amalgamation of many diverse networks, it has too many vulnerable points. Hence, the future design of the network must be planned to reduce the number of vulnerable network points and reduce the number of service nodes that participate in servicing the subscriber, thereby reducing the number of points from which an adversary may attack.

References

[1] 3GPP, architectural requirements, Technical Standard 3G TS 23.221 V6.3.0, 3G Partnership Project, May 2004.

[2] Murakami K, Haase O, Shin J, LaPorta TF. Mobility management alternatives for migration to mobile internet session-based services. IEEE Journal on Selected Areas in Communications (J-SAC), special issue on Mobile Internet 2004;22:834–848.

[3] 3GPP, 3G security, Security threats and requirements, Technical Standard 3G TS 21.133 V3.1.0, 3G Partnership Project, December 1999.
[4] 3GPP, network architecture, Technical Standard 3G TS 23.002 V3.3.0, 3G Partnership Project, May 2000.
[5] Eberspacher V. GSM Switching, Services and Protocols. John Wiley & Sons; 1999.
[6] 3GPP, Basic call handling - technical realization, Technical Standard 3GPP TS 23.018 V3.4.0, 3G Partnership Project, April 1999.
[7] 3GPP, A guide to 3rd generation security, Technical Standard 3GPP TR 33.900 V1.2.0, 3G Partnership Project, January 2001.
[8] ibid.
[9] Chatras B, Vernhes C. Mobile application part design principles. Proceedings of XIII International Switching Symposium, vol. 1, June 1990:35–42.
[10] Audestad JA. The mobile application part (map) of GSM, technical report. Telektronikk, March 2004.
[11] 3GPP, Mobile Application part (MAP) specification, Technical Standard 3GPP TS 29.002 V3.4.0, 3G Partnership Project, April 1999.
[12] Boman K, Horn G, Howard P, Niemi V. Umts security. Electronics Communications Engineering Journal 14:(5) (October 2002) 191–204. Special issue security for mobility.
[13] 3GPP, A guide to 3rd generation security, Technical Standard 3GPP TR 33.900 V1.2.0, 3G Partnership Project, January 2001.
[14] Kotapati K, Liu P, LaPorta TF. Dependency relation-based vulnerability analysis of 3G networks: can it identify unforeseen cascading attacks? Special Issue of Springer Telecommunications Systems on Security, Privacy and Trust for Beyond 3G Networks, March 2007.
[15] Kotapati K, Liu P, LaPorta TF. Evaluating MAPSec by marking attack graphs. ACM/Kluwer Journal of Wireless Networks Journal (WINET) 2008.
[16] Kotapati K, Liu P, LaPorta TF. EndSec: An end-to-end message security protocol for cellular networks IEEE Workshop on Security, Privacy and Authentication in Wireless Networks (SPAWN 2008) in IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WOWMOM), June 2008.
[17] Kotapati K, Liu P, LaPorta TF. Evaluating MAPSec by marking attack graphs. ACM/Kluwer Journal of Wireless Networks Journal (WINET) 2008.
[18] ibid.
[19] Moore D, Paxson V, Savage S, Shannon C, Staniford S, Weaver N. Inside the slammer worm. IEEE Security and Privacy 2003;1(4):33–39.
[20] Enck W, Traynor P, McDaniel P, LaPorta TF. Exploiting open functionality in sms-capable cellular networks. In: CCS '05: Proceedings of the 12th ACM Conference on Computer and Communications Security, ACM Press; 2005.
[21] ibid.
[22] Traynor P, Enck W, McDaniel P, LaPorta TF. Mitigating attacks on open functionality in sms-capable cellular networks In: MobiCom '06: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking, ACM Press; 2006. p. 182–193.
[23] Traynor P, McDaniel P, LaPorta TF. On attack causality in internet-connected cellular networks, USENIX Security Symposium (SECURITY). August 2007.
[24] Traynor P, Enck W, McDaniel P, LaPorta TF. Mitigating attacks on open functionality in SMS-capable cellular networks. In: MobiCom '06: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking, ACM Press; 2006. p. 182–193.
[25] ibid.

[26] Traynor P, McDaniel P, LaPorta TF. On attack causality in internet-connected cellular networks, USENIX Security Symposium (SECURITY). August 2007.
[27] Moore T, Kosloff T, Keller J, Manes G, Shenoi S. Signaling system 7 (SS7) network security. Proceedings of the IEEE 45th Midwest Symposium on Circuits and Systems, August 2002.
[28] Lorenz G, Moore T, Manes G, Hale J, Shenoi S. Securing SS7 telecommunications networks. Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, June 2001.
[29] Moore T, Kosloff T, Keller J, Manes G, Shenoi S. Signaling system 7 (SS7) network security. In: Proceedings of the IEEE 45th Midwest Symposium on Circuits and Systems, August 2002.
[30] Kotapati K, Liu P, Sun Y, LaPorta TF. A taxonomy of cyber attacks on 3G networks. In: Proceedings IEEE International Conference on Intelligence and Security Informatics, ISI, Lecture Notes in Computer Science. Springer-Verlag, May 2005. p. 631–633.
[31] ibid.
[32] Kotapati K. Assessing Security of Mobile Telecommunication Networks. Ph. D dissertation. Penn State University; 2008.
[33] Switch, 5ESS Switch. www.alleged.com/telephone/5ESS/.
[34] Telcoman, Central Offices. www.thecentraloffice.com/.
[35] Prevelakis V, Spinellis D. The Athens affair, IEEE Spectrum. (July 2007).
[36] Hannu H. Signaling Compression (SigComp) Requirements & Assumptions, RFC 3322 (Informational); January 2003.
[37] Stallings W. Cryptography and Network Security: Principles and Practice. Prentice Hall; 2000.
[38] Kotapati K. Assessing Security of Mobile Telecommunication Networks. Ph. D dissertation. Penn State University; 2008.
[39] Kotapati K, Liu P, LaPorta TF. CAT - a practical graph & SDL based toolkit for vulnerability assessment of 3G networks. In: Proceedings of the 21st IFIP TC-11 International Information Security Conference, Security and Privacy in Dynamic Environments, SEC 2006, May 2006.
[40] Kotapati K, Liu P, LaPorta TF. Dependency relation-based vulnerability analysis of 3G networks: can it identify unforeseen cascading attacks? Special Issue of Springer Telecommunications Systems on Security, Privacy and Trust for Beyond 3G Networks, March 2007.
[41] Kotapati K, Liu P, LaPorta TF. Evaluating MAPSec by marking attack graphs. ACM/Kluwer Journal of Wireless Networks Journal (WINET) 2008:12.
[42] 3GPP2 3GPP, Third Generation Partnership Project. www.3gpp.org/, 2006.
[43] Telcoman, Central Offices. www.thecentraloffice.com/.
[44] Ellsberger J, Hogrefe D, Sarma A. SDL. Formal Object-oriented Language for Communicating Systems. Prentice Hall; 1997.
[45] Kotapati K, Liu P, LaPorta TF. Dependency relation-based vulnerability analysis of 3G networks: can it identify unforeseen cascading attacks? Special Issue of Springer Telecommunications Systems on Security, Privacy and Trust for Beyond 3G Networks, March 2007.
[46] Kotapati K, Liu P, LaPorta TF. Dependency relation-based vulnerability analysis of 3G networks: can it identify unforeseen cascading attacks? Special Issue of Springer Telecommunications Systems on Security, Privacy and Trust for Beyond 3G Networks, March 2007.
[47] Kotapati K, Liu P, LaPorta TF. Dependency relation-based vulnerability analysis of 3G networks: can it identify unforeseen cascading attacks? Special Issue of Springer Telecommunications Systems on Security, Privacy and Trust for Beyond 3G Networks, March 2007.
[48] Kotapati K, Liu P, LaPorta TF. Evaluating MAPSec by marking attack graphs. ACM/Kluwer Journal of Wireless Networks Journal (WINET) 2008:12.

This page intentionally left blank

CHAPTER 13

Radio Frequency Identification Security

Chunming Rong, Erdal Cayirci, Gansen Zhao, Laing Yan
University of Stavanger

Radio frequency identification (RFID) systems use RFID tags to annotate and identify objects. When objects are processed, an RFID reader is used to read information from the tags attached to the objects. The information will then be used with the data stored in the back-end databases to support the handling of business transactions.

1. Radio Frequency Identification Introduction

Generally, an RFID system consists of three basic components: RFID tags, RFID readers, and a back-end database.

- *RFID tags or RFID transponders*. These are the data carriers attached to objects. A typical RFID tag contains information about the attached object, such as an identifier (ID) of the object and other related properties of the object that may help to identify and describe it.
- *The RFID reader or the RFID transceiver*. These devices can read information from tags and may write information into tags if the tags are rewritable.
- *Back-end database*. This is the data repository responsible for the management of data related to the tags and business transactions, such as ID, object properties, reading locations, reading time, and so on.

RFID System Architecture

RFID systems' architecture is illustrated in Figure 13.1. Tags are attached to or embedded in objects to identify or annotate them. An RFID reader will send out signals to a tag for requesting information stored on the tag. The tag responds to the request by sending back the appropriate information. With the data from the back-end database, applications can then use the information from the tag to proceed with the business transaction related to the object.

Next we describe the RFID tags, RFID reader, and back-end database in detail.

Network and System Security. DOI:10.1016/B978-1-59749-535-6.00013-0

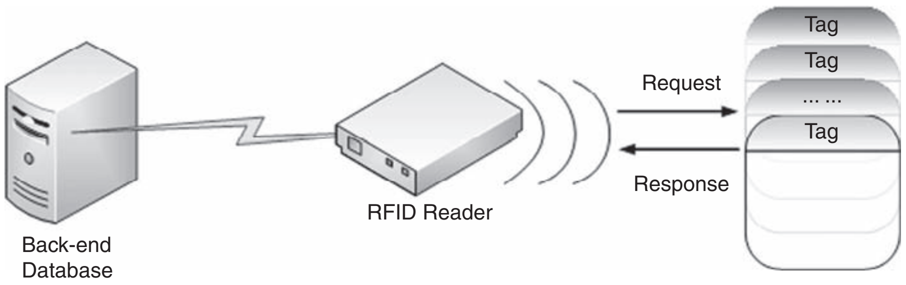


Figure 13.1: RFID system architecture.

Tags

In RFID systems, objects are identified or described by information on RFID tags attached to the objects. An RFID tag basically consists of a microchip that is used for data storage and computation and a coupling element for communicating with the RFID reader via radio frequency communication, such as an antenna. Some tags may also have an on-board battery to supply a limited amount of power.

RFID tags can respond to radio frequencies sent out by RFID readers. On receiving the radio signals from a RFID reader, an RFID tag will either send back the requested data stored on the tag or write the data into the tag, if the tag is rewritable. Because radio signals are used, RFID tags do not require line of sight to connect with the reader and precise positioning, as barcodes do. Tags may also generate a certain amount of electronic power from the radio signals they receive, to power the computation and transmission of data.

RFID tags can be classified based on four main criteria: power source, type of memory, computational power, and functionality.

A basic and important classification criterion of RFID tags is to classify tags based on power source. Tags can be categorized into three classes: active, semiactive, and passive RFID tags.

Active RFID tags have on-board power sources, such as batteries. Active RFID tags can proactively send radio signals to an RFID reader and possibly to other tags [1] as well. Compared with tags without on-board power, active tags have longer transmission range and are more reliable. Active tags can work in the absence of an RFID reader. However, the on-board power supply also increases the costs of active tags.

Semiactive RFID tags also have on-board power sources for powering their microchips, but they use RFID readers' energy field for actually transmitting their data [2] when responding to the incoming transmissions. Semiactive tags have the middle transmission range and cost.

Passive RFID tags do not have internal power sources and cannot initiate any communications. Passive RFID tags generate power from radio signals sent out by an RFID reader in the course

Table 13.1: Tags Classified by Power Source

Power Source	Active Tags	Semiactive Tags	Passive Tags
On-board power supply	Yes	Yes	No
Transmission range	Long	Medium	Short
Communication pattern	Proactive	Passive	Passive
Cost	Expensive	Medium	Inexpensive

of communication. Thus passive RFID tags can only work in the presence of an RFID reader. Passive tags have the shortest transmission range and the cheapest cost. The differences of active tags, semiactive tags and passive tags are shown in Table 13.1.

RFID tags may have read-only memory, write-once/read-many memory, or fully rewritable memory. RFID tags can be classified into three categories according to the type of memory that a tag uses: read-only tags, write-once/read-many tags, and fully rewritable tags. The information on read-only tags cannot be changed in the life-cycle of the tags. Write-once/read-many tags can be initialized with application-specific information. The information on fully rewritable tags can be rewritten many times by an RFID reader.

According to the computational power, RFID tags can be classified into three categories: basic tags, symmetric-key tags, and public-key tags. Basic tags do not have the ability to perform cryptography computation. Symmetric-key tags and public-key tags have the ability to perform symmetric-key and public-key cryptography computation, respectively.

RFID tags can also be classified according to their functionality. The MIT Auto-ID Center defined five classes of tags according to their functionality in 2003 [3]: Class 0, Class 1, Class 2, Class 3, and Class 4 tags. Every class has different functions and different requirements for tag memory and power resource. Class 0 tags are passive and do not contain any memory. They only announce their presence and offer electronic article surveillance (EAS) functionality. Class 1 tags are typically passive. They have read-only or write-once/read-many memory and can only offer identification functionality. Class 2 tags are mostly semiactive and active. They have fully rewritable memory and can offer data-logging functionality. Class 3 tags are semiactive and active tags. They contain on-board environmental sensors that can record temperature, acceleration, motion, or radiation and require fully rewritable memory. Class 4 tags are active tags and have fully rewritable memory. They can establish ad hoc wireless networks with other tags because they are equipped with wireless networking components.

RFID Readers

An RFID reader (transceiver) is a device used to read information from and possibly also write information into RFID tags. An RFID reader is normally connected to a back-end database for sending information to that database for further processing.

An RFID reader consists of two key functional modules: a high-frequency (HF) interface and a control unit. The HF interface can perform three functions: generating the transmission power to activate the tags, modulating the signals for sending requests to RFID tags, and receiving and demodulating signals received from tags. The control unit of an RFID reader has also three basic functions: controlling the communication between the RFID reader and RFID tags, encoding and decoding signals, and communicating with the back-end server for sending information to the back-end database or executing the commands from the back-end server. The control unit can perform more functions in the case of complex RFID systems, such as executing anticollision algorithms in the cause of communicating with multitags, encrypting requests sent by the RFID reader and decrypting responses received from tags, and performing the authentication between RFID readers and RFID tags [4].

RFID readers can provide high-speed tag scanning. Hundreds of objects can be dealt with by a single reader within a second; thus it is scalable enough for applications such as supply chain management, where a large number of objects need to be dealt with frequently. RFID readers need only to be placed at every entrance and exit. When products enter or leave the designated area by passing through an entrance or exit, the RFID readers can instantly identify the products and send the necessary information to the back-end database for further processing.

Back-End Database

The back-end database is in the back-end server that manages the information related to the tags in an RFID system. Every object's information can be stored as a record in the database, and the information on the tag attached to the object can serve as a pointer to the record.

The connection between an RFID reader and a back-end database can be assumed as secure, no matter via wireless link or TCP/IP, because constraints for readers are not very tight and security solutions such as SSL/TLS can be implemented for them [5].

RFID Standards

Currently, as different frequencies are used for RFID systems in various countries and many standards are adopted for different kinds of application, there is no agreement on a universal standard that's accepted by all parties. Several kinds of RFID standards [6] are being used today. These standards include contactless smart cards, item mana-gement tags, RFID systems for animal identification, and EPC tags. These standards specify the physical layer and the link layer characteristics of RFID systems but do not cover the upper layers.

Contactless smart cards can be classified into three types according to the communication ranges. The ISO standards for them are ISO 10536, ISO 14443, and ISO 15693. ISO 10536 sets the standard for close-coupling smart cards, for which the communication range is about

0–1 cm. ISO 14443 sets the standard for proximity-coupling smart cards, which have a communication range of about 0–10 cm. ISO 15693 specifies vicinity-coupling smart cards, which have a communication range of about 0–1 m. The proximity-coupling and vicinity-coupling smart cards have already been implemented with some cryptography algorithms such as 128-bit AES, triple DES, and SHA-1 and challenge-response authentication mechanisms to improve system security [7].

Item management tag standards include ISO 15961, ISO 15962, ISO 15963, and ISO 18000 series [8]. ISO 15961 defines the host interrogator, tag functional commands, and other syntax features of item management. ISO 15962 defines the data syntax of item management, and ISO 15963 is “Unique Identification of RF tag and Registration Authority to manage the uniqueness.” For the ISO 18000 standards series, part 1 describes the reference architecture and parameters definition; parts 2, 3, 4, 5, 6, and 7 define the parameters for air interface communications below 135 kHz, at 13.56 MHz, at 2.45 GHz, at 860 MHz, at 960 MHz, and at 433 MHz, respectively.

Standards for RFID systems for animal identification include ISO 11784, ISO 11785, and ISO 14223 [9]. ISO 11784 and ISO 11784 define the code structure and technical concepts for radio frequency identification of animals. ISO 14223 includes three parts: air interface, code and command structure, and applications. These kinds of tags use low frequency for communication and have limited protection for animal tracking [10].

The EPC standard was created by the MIT Auto-ID, which is an association of more than 100 companies and university labs. The EPC system is currently operated by EPCglobal [11]. A typical EPC network has four parts [12]: the electronic product code, the identification system that includes RFID tags and RFID readers, the Savant middleware, and the object naming service (ONS). The first- and second-generation EPC tags cannot support strong cryptography to protect the security of the RFID systems due to the limitation of computational resources, but both of them can provide a kill command to protect the privacy of the consumer [13].

EPC tag encoding includes a Header field followed by one or more Value fields. The Header field defines the overall length and format of the Value fields. There are two kinds of EPC format: EPC 64-bit format and EPC 96-bit format. In the most recent version [14], the 64-bit format was removed from the standard. As shown in Table 13.2, both the formats include four fields: a header (8 bits), an EPC manager number (28 bits), an object class (24 bits), and a serial number (36 bits). The header and the EPC manager number are assigned by EPCglobal [15], and the object class and the serial number are assigned by EPC

Table 13.2: EPC Basic Format

Header	EPC Manager Number	Object Class	Serial Number

manager owner. The EPC header identifies the length, type, structure version, and generation of the EPC. The EPC manager number is the entity responsible for maintaining the subsequent partitions of the EPC. The object class identifies a class of objects. Serial number identifies the instance.

RFID Applications

Recently more and more companies and organizations have begun to use RFID tags rather than traditional barcode because RFID systems have many advantages over traditional barcode systems. First, the information stored in the RFID tags can be read by RFID readers without line of sight, whereas barcodes can only be scanned within the line of sight. Second, the distance between a tag and a reader is longer compared with the barcode system. For example, an RFID reader can read information from a tag at a distance as long as 300 feet, whereas the read range for a barcode is typically no more than 15 feet. Third, RFID readers can scan hundreds of tags in seconds. Fourth, since today most RFID tags are produced using silicon technology, more functions can be added to them, such as large memory for more information storage and calculation ability to support various kinds of encryption and decryption algorithms, so privacy can be better protected and the tags cannot be easily cloned by attackers. In addition, the information stored in the barcode cannot be changed after being imprinted on the barcode, whereas for the RFID tags with rewritable memory the information can be updated when needed.

With these characteristics and advantages, RFID has been widely adopted and deployed in various areas. Currently, RFID can be used in passports, transportation payments, product tracking, lap scoring, animal identification, inventory systems, RFID mandates, promotion tracking, human implants, libraries, schools and universities, museums, and social retailing. These myriad applications of RFID can be classified into seven classes according to the purpose of identifying items [16]. These classes are asset management, tracking, authenticity verification, matching, process control, access control, and automated payment. Table 13.3 lists the identification purposes of various application types.

Table 13.3: RFID Application Purpose

Application Type	Identification Purpose
Asset management	Determine item presence
Tracking	Determine item location
Authenticity verification	Determine item source
Matching	Ensure affiliated items are not separated
Process control	Correlate item information for decision making
Access control	Person authentication
Automated payment	Conduct financial transaction

Asset management involves determining the presence of tagged items and helping manage item inventory. One possible application of asset management is electronic article surveillance (EAS). For example, every good in a supermarket is attached to an EAS tag, which will be deactivated if it is properly checked out. Then RFID readers at the supermarket exits can detect unpaid goods automatically when they pass through.

Tracking is used to identify the location of tagged items. If the readers are fixed, a single reader can cover only one area. To effectively track the items, a group of readers is needed, together with a central system to deal with the information from different readers.

Authenticity verification methods are used to verify the source of tagged items. For example, by adding a cryptography-based digital signature in the tag, the system can prevent tag replication to make sure that the good is labeled with the source information.

Matching is used to ensure that affiliated items are not separated. Samples for matching applications include mothers and their newborn babies to match each other in the hospital and for airline passengers to match their checked luggage and so prevent theft.

Access control is used for person authentication. Buildings may use contactless RFID card systems to identify authorized people. Only those authorized people with the correct RFID card can authenticate themselves to the reader to open a door and enter a building. Using a car key with RFID tags, a car owner can open his own car automatically, another example of RFID's application to access control.

Process control involves decision making by correlating tagged item information. For example, RFID readers in different parts of an assembly line can read the information on the products, which can be used to help production managers make suitable decisions.

Automated payment is used to conduct financial transactions. The applications include payment for toll expressways and at gas stations. These applications can improve the speed of payment to hasten the processing of these transactions.

2. RFID Challenges

RFID systems have been widely deployed in some areas. Perhaps this happened before the expectations of RFID researchers and RFID services providers were satisfied. There are many limitations of the RFID technology that restrain the deployment of RFID applications, such as the lack of universal standardization of RFID in the industry and the concerns about security and privacy problems that may affect the privacy and security of individuals and organizations. The security and privacy issues pose a huge challenge for RFID applications. Here we briefly summarize some of the challenges facing RFID systems.

Counterfeiting

As described earlier in the chapter, RFID tags can be classified into three categories based on the equipped computation power: basic tags, symmetric-key tags, and public-key tags. Symmetric-key and public-key tags can implement cryptography protocols for authentication with private key, and public keys, respectively. Basic tags are not capable of performing cryptography computation. Although they lack the capability to perform cryptography computation, they are most widely used for applications such as supply chain management and travel systems. With the widespread application of fully writable or even reprogrammable basic tags, counterfeiters can easily forge basic tags in real-world applications, and these counterfeit tags can be used in multiple places at the same time, which can cause confusion.

The counterfeiting of tags can be categorized into two areas based on the technique used for tampering with tag data: modifying tag data and adding data to a blank tag. In real-world applications, we face counterfeit threats such as the following [17]:

- The attacker can modify valid tags to make them invalid or modify invalid tags to make them valid.
- The attacker can modify a high-priced object's tag as a low-priced object or modify a low-priced object's tag as a high-priced object.
- The attacker can modify an object's tag to be the same as the tags attached to other objects.
- The attacker can create an additional tag for personal reasons by reading the data from an authorized tag and adding this data to a blank tag in real-world applications, such as in a passport or a shipment of goods.

Sniffing

Another main issue of concern in deploying RFID systems is the sniffing problem. It occurs when third parties use a malicious and unauthorized RFID reader to read the information on RFID tags within their transmission range. Unfortunately, most RFID tags are indiscriminate in their responses to reading requests transmitted by RFID readers and do not have access control functions to provide any protection against an unauthorized reader. Once an RFID tag enters a sufficiently powered reader's field, it receives the reader's requests via radio frequency. As long as the request is well formed, the tag will reply to the request with the corresponding information on the tag. Then the holder of the unauthenticated reader may use this information for other purposes.

Tracking

With multiple RFID readers integrated into one system, the movements of objects can be tracked by fixed RFID readers [18]. For example, once a specific tag can be associated with a particular person or object, when the tag enters a reader's field the reader can obtain the specific identifier of the tag, and the presence of the tag within the range of a specific reader implies specific location information related to the attached person or object. With location information coming from multiple RFID readers, an attacker can follow movements of people or objects. Tracking can also be performed without decrypting the encrypted messages coming from RFID readers [19]. Generally, the more messages the attacker describes, the more location or privacy information can be obtained from the messages.

One way to track is to generate maps of RFID tags with mobile robots [20]. A sensor model is introduced to compute the likelihood of tag detections, given the relative pose of the tag with respect to the robot. In this model a highly accurate FastSLAM algorithm is used to learn the geometrical structure of the environment around the robots, which are equipped with a laser range scanner; then it uses the recursive Bayesian filtering scheme to estimate the posterior locations of the RFID tags, which can be used to localize robots and people in the environment with the geometrical structure of the environment learned by the FastSLAM algorithm.

There is another method to detect the motion of passive RFID tags that are within a detecting antenna's field. Response rate at the reader is used to study the impact of four cases of tag movements that can provide prompt and accurate detection and the influence of the environment. The idea of multiple tags/readers is introduced to improve performance. The movement-detection algorithms can be improved and integrated into the RFID monitoring system to localize the position of the tags. The method does not require any modification of communication protocols or the addition of hardware.

In real-world applications, there exists the following tracking threat:

- The attacker can track the potential victim by monitoring the movement of the person and performing some illegal actions against the potential victim [21].

Denial of Service

Denial of service (DoS) takes place when RFID readers or back-end servers cannot provide excepted services. DoS attacks are easy to accomplish and difficult to guard against [22]. The following are nine DoS threats.

- *Killing tags to make them disabled to disrupt readers' normal operations*. EPCglobal had proposed that a tag have a "kill" command to destroy it and protect consumer privacy. If an attacker knows the password of a tag, it can "kill" the tag easily in real-world applications. Now Class-0, Class-1 Generation-1, and Class-1 Generation-2 tags are all equipped with the kill command.

- *Carry a blocker tag that can disrupt the communication between an RFID reader and RFID tags.* A blocker tag is an inexpensive, passive RFID device that can simulate many basic RFID tags at one time and render specific zones private or public. An RFID reader can only communicate with a single RFID tag at any specific time. If more than one tag responds to a request coming from the reader at the same time, "collision" happens. In this case, the reader cannot receive the information sent by the tags, which makes the system unavailable to authorized uses.
- *Carry a special absorbent tag that can be tuned to the same radio frequencies used by legitimate tags.* The absorbent tag can absorb the energy or power generated by radio frequency signals sent by the reader, and the resulting reduction in the reader's energy may make the reader unavailable to communicate with other tags.
- *Remove, physically destroy, or erase the information on tags attached to or embedded in objects.* The reader will not communicate with the dilapidated tags in a normal way.
- *Shield the RFID tags from scrutiny using a Faraday cage.* A Faraday cage is a container made of a metal enclosure that can prevent reading radio signals from the readers [23].
- *Carry a device that can actively broadcast more powerful return radio signals or noises than the signals responded to by the tags so as to block or disrupt the communication of any nearby RFID readers and make the system unavailable to authorized users.* The power of the broadcast is so high that it could cause severe blockage or disruption of all nearby RFID systems, even those in legitimate applications where privacy is not a concern [24].
- *Perform a traditional Internet DoS attack and prevent the back-end servers from gathering EPC numbers from the readers.* The servers do not receive enough information from the readers and cannot provide the additional services from the server.
- *Perform a traditional Internet DoS attack against the object-naming service (ONS).* This can deny the service.
- *Send URL queries to a database and make the database busy with these queries.* The database may then deny access to authorized users.

Other Issues

Besides the four basic types of attack—counterfeiting, sniffing, tracking, and denial of service—in real-world applications, there also exists some other threats of RFID systems.

Spoofing

Spoofing attacks take place when an attacker successfully poses as an authorized user of a system [25]. Spoofing attacks are different from counterfeiting and sniffing attacks, though they are all falsification types of attack. Counterfeiting takes place when an attacker forges the RFID tags that can be scanned by authorized readers. Sniffing takes place when an attacker forges authorized readers that can scan the authorized tags to obtain useful information. But the forging object of spoofing is an authorized user of a system. There exist the following spoofing threats in real-world applications [26]:

- *The attacker can pose as an authorized EPC global Information Service Object Naming Service (ONS) user.* If the attacker successfully poses as an authorized ONS user, he can send queries to the ONS to gather EPC numbers. Then, from the EPC numbers, the attacker may easily obtain the location, identification, or other privacy information.
- *The attacker can pose as an authorized database user in an RFID system.* The database stores the complete information from the objects, such as manufacturer, product name, read time, read location, and other privacy information. If the attacker successfully poses as an authorized database user and an authorized user of ONS, he can send queries to the ONS for obtaining the EPC number of one object, then get the complete information on the object by mapping the EPC number to the information stored in the database.
- *The attacker can also pose as an ONS server.* If the attacker's pose is successful, he can easily use the ONS server to gather EPC numbers, respond to invalid requests, deny normal service, and even change the data or write malicious data to the system.

Repudiation

Repudiation takes place when a user denies doing an action or no proof exists to prove that the action has been implemented [27]. There are two kinds of repudiation threats.

- The sender or the receiver denies performing the send and receive actions. A nonrepudiation protocol can be used to resolve this problem.
- The owner of the EPC number or the back-end server denies that it has the information from the objects to which the tags are attached.

Insert Attacks

Insert attacks take place when an attacker inserts some system commands to the RFID system where data is normally expected [28]. In real-world applications, the following attack exists:

- A system command rather than valid data is carried by a tag in its data storage memory.

Replay Attacks

Replay attacks take place when an attacker intercepts the communication signals between an RFID reader and an RFID tag and records the tag's response. Then the RFID tag's response can be reused if the attacker detects that the reader sends requests to the other tags for querying [29]. The following two threats exist:

- The attacker can record the communications between proximity cards and a building access reader and play it back to access the building.
- The attacker can record the response that an RFID card in a car gives to an automated highway toll collection system, and the response can be used when the car of the attacker wants to pass the automated toll station.

Physical Attacks

Physical attacks are very strong attacks that physically obtain tags and have unauthorized physical operations on the tags. But it is fortunate that physical attacks cannot be implemented in public or on a widespread scale, except for Transient Electromagnetic Pulse Emanation Standard (TEMPEST) attacks. The following physical attacks exist [30,31]:

- *Probe attacks.* The attacker can use a probe directly attached to the circuit to obtain or change the information on tags.
- *Material removal.* The attacker can use a knife or other tools to remove the tags attached to objects.
- *Energy attacks.* The attacks can be either of the contact or contactless variety. It is required that contactless energy attacks be close enough to the system.
- *Radiation imprinting.* The attacker can use an X-ray band or other radial bands to destroy the data unit of a tag.
- *Circuit disruption.* The attacker can use strong electromagnetic interference to disrupt tag circuits.
- *Clock glitch.* The attacker can lengthen or shorten the clock pulses to a clocked circuit and destroy normal operations.

Viruses

Viruses are old attacks that threaten the security of all information systems, including RFID systems. RFID viruses always target the back-end database in the server, perhaps destroying and revealing the data or information stored in the database. The following virus threats exist:

- An RFID virus destroys and reveals the data or information stored in the database.
- An RFID virus disturbs or even stops the normal services provided by the server.
- An RFID virus threatens the security of the communications between RFID readers and RFID tags or between back-end database and RFID readers.

Social Issues

Due to the security challenges in RFID, many people do not trust RFID technologies and fear that they could allow attackers to purloin their privacy information.

Weis [31] presents two main arguments. These arguments make some people choose not to rely on RFID technology and regard RFID tags as the "mark of the beast." However, security issues cannot prevent the success of RFID technology.

The first argument is that RFID tags are regarded as the best replacement for current credit cards and all other ways of paying for goods and services. But RFID tags can also serve as identification. The replacement of current ways of paying by RFID tag requires that people accept RFID tags instead of credit cards, and they cannot sell or buy anything without RFID tags.

There is a second argument [33]: "Since RFID tags are also used as identification, they should be implanted to avoid losing the ID or switching it with someone. Current research has shown that the ideal location for the implant is indeed the forehead or the hand, since they are easy to access and unlike most other body parts they do not contain much fluid, which interferes with the reading of the chip."

Comparison of All Challenges

Previously in this chapter we introduced some of the challenges that RFID systems are facing. Every challenge or attack can have a different method or attack goal, and the consequences of the RFID system after an attack may also be different. In this part of the chapter, we briefly analyze the challenges according to attack methods, attack goals, and the consequences of RFID systems after attacks (see Table 13.4).

The first four challenges are the four basic challenges in RFID systems that correspond to the four basic use cases. *Counterfeiting* happens when counterfeiters forge RFID tags by copying the information from a valid tag or adding some well-formed format information to a new tag in the RFID system. *Sniffing* happens when an unauthorized reader reads the information from a tag, and the information may be utilized by attackers. *Tracking* happens when an attacker who holds some readers unlawfully monitors the movements of objects attached by an RFID tag that can be read by those readers. *Denial of service* happens when the components of RFID systems deny the RFID service.

The last seven challenges or attacks can always happen in RFID systems (see Table 13.4). *Spoofing* happens when an attacker poses as an authorized user of an RFID system on which the attacker can perform invalid operations. *Repudiation* happens when a user or component of an RFID system denies the action it performed and there is no proof that the user did

Table 13.4: Comparison of All Challenges or Attacks in RFID Systems

Challenge or Attack	Attack Method	Attack Goal	Direct Consequence
Counterfeiting	Forge tags	Tag	Invalid tags
Sniffing	Forge readers	Reader	Reveals information
Tracking	Monitor the movement of objects	Objects of an RFID system	Tracks the movement of object
Denial of service	RF jamming, kill normal command, physically destroy, and so on	Reader, back-end database or server	Denies normal services
Spoofing	Pose as an authorized user	User	Invalid operations by invalid user
Repudiation	Deny action or no proof that the action was implemented	Tag, reader, back-end database or server	Deniable actions
Insert attacks	Insert invalid command	Tag	Invalid operations by invalid commands
Replay attacks	Reuse the response of tags	Communication between RFID tags and readers	Invalid identification
Physical attacks	Physical operations on tag	Tag	Disrupts or destroys communication between RFID tags and readers
Virus	Insert invalid data	Back-end database or server	Destroys data or service of system
Social issues	Social attitude	Psychology of potential user	Restricts the widespread application

perform the action. *Insert attacks* happen when an attacker inserts some invalid system commands into the tags and some operations may be implemented by the invalid command. *Replay attacks* happen when an attacker intercepts the response of the tag and reuses the response for another communication. *Physical attacks* happen when an attacker does some physical operations on RFID tags and these attacks disrupt communications between the RFID readers and tags. A *virus* is the security challenge of all information systems; it can disrupt the operations of RFID systems or reveal the information in those systems. *Social issues* involve users' psychological attitudes that can influence the users' adoption of RFID technologies for real-world applications.

3. RFID Protections

According to their computational power, RFID tags can be classified into three categories: basic tags, symmetric-key tags, and public-key tags. In the next part of the chapter, we introduce some protection approaches for these three kinds of RFID tags.

Basic RFID System

Prices have been one of the biggest factors to be considered when we're making decisions on RFID deployments. Basic tags are available for the cheapest price, compared with symmetric-key tags and public-key tags. Due to the limited computation resources built into a basic tag, basic tags are not capable of performing cryptography computations. This imposes a huge challenge to implement protections on basic tags; cryptography has been one of the most important and effective methods to implement protection mechanisms. Recently several approaches have been proposed to tackle this issue.

Most of the approaches to security protection for basic tags focus on protecting consumer privacy. A usual method is by tag killing, proposed by EPCglobal. In this approach, when the reader wants to kill a tag, it sends a kill message to the tag to permanently deactivate it. Together with the kill message, a 32-bit tag-specific PIN code is also sent to the object tag, to avoid killing other tags. On receiving this kill message, a tag will deactivate itself, after which the tag will become inoperative. Generally, tags are killed when the tagged items are checked out in shops or supermarkets. This is very similar to removing the tags from the tagged items when they are purchased. It is an efficient method of protecting the privacy of consumers, since a killed tag can no longer send out information.

The disadvantage of this approach is that it will reduce the post-purchase benefits of RFID tags. In some cases, RFID tags need to be operative only temporarily. For example, RFID tags used in libraries and museums for tagging books and other items need to work at all times and should not be killed or be removed from the tagged items. In these cases, instead of being killed or removed, tags can be made temporarily inactive. When a tag needs to be reawoken, an RFID reader can send a wake message to the tag with a 32-bit tag-specific PIN code, which is sent to avoid waking up other tags. This also results in the management of PIN codes for tags, which brings some inconvenience.

Another approach to protecting privacy is tag relabeling, which was first proposed by Sarma et al. [34]. In this scheme, to protect consumers' privacy, identifiers of RFID tags are effaced when tagged items are checked out, but the information on the tags will be kept for later use. Inoue and Yasuuran [35] proposed that consumers can store the identifiers of the tags and give each tag a new identifier. When needed, people can reactivate the tags with the new identifiers. This approach allows users to manage tagged items throughout the items' life cycle. A third approach is to allocate each tag a new random number at each checkout; thus attackers cannot rely on the identifiers to collect information about customers [36]. This method does not solve the problem of tracking [37]. To prevent tracking, random numbers need to be refreshed frequently, which will increase the burden on consumers. Juels proposed a system called the *minimalist* system [38], in which every tag has a list of pseudonyms, and for every reader query, the tag will respond with a different pseudonym from the list and return to the beginning of the list when this list is exhausted. It is assumed that only authorized readers know all these tag pseudonyms. Unauthorized readers that do not know these pseudonyms cannot identify the tags correctly. To prevent unauthorized readers getting the pseudonyms list by frequent query, the tags will response to an RFID reader's request with a relatively low rate, which is called *pseudonym throttling*. Pseudonym throttling is useful, but it cannot provide a high level of privacy for consumers, because with the tag's small memory, the number of pseudonyms in the list is limited. To tackle this problem, the protocol allows an authorized RFID reader to refresh a tag's pseudonyms list.

Juels and Pappu [39] proposed to protect consumers' privacy by using tagged banknotes. The proposed scheme used public-key cryptography to protect the serial numbers of tagged banknotes. The serial number of a tagged banknote is encrypted using a public key to generate a ciphertext, which is saved in the memory of the tag. On receiving a request of the serial number, the tag will respond with this ciphertext. Only law enforcement agencies know the related private key and can decrypt this ciphertext to recover the banknote's serial number. To prevent tracking of banknotes, the ciphertext will be reencrypted periodically. To avoid the ciphertext of a banknote being reencrypted by an attacker, the tagged banknote can use an optical write–access key. A reader that wants to reencrypt this ciphertext needs to scan the write–access key first. In this system only one key pair, a public key and a private key, is used. But this is not enough for the general RFID system. Using multiple key pairs will impair the privacy of RFID systems, since if the reader wants to reencrypt the ciphertext, it needs to know the corresponding public key of this tag.

So, a universal reencryption algorithm has been introduced [40]. In this approach, an RFID reader can reencrypt the ciphertext without knowing the corresponding public key of a tag. The disadvantage of this approach is that attackers can substitute the ciphertext with a new ciphertext, so the integrity of the ciphertext cannot be protected. By signing the ciphertext with a digital signature, this problem can be solved [41], since only the authenticated reader can access the ciphertext.

Floerkemeier et al. [42] introduced another approach to protect consumer privacy by using a specially designed protocol. In their approach, they first designed the communication protocol between RFID tags and RFID readers. This protocol requires an RFID reader to provide information about the purpose and the collection type for the query. In addition, a privacy-enforcing device called a *watchdog tag* is used in the system. This watchdog tag is a kind of sophisticated RFID tag that is equipped with a battery, a small screen, and a long-range communication channel. A watchdog tag can be integrated into a PDA or a cell phone and can decode the messages from an RFID reader and display them on the screen for the user to read. With a watchdog tag, a user can know not only the information from the RFID readers in the vicinity of the tag but also the ID, the query purpose, and the collection type of the requests sent by the RFID readers. With this information, the user is able to identify the unwanted communications between tags and an RFID reader, making this method useful for users to avoid the reader ID spoofing attack.

Rieback and colleagues [43] proposed another privacy-enforcing device called RFID Guardian, which is also a battery-powered RFID tag that can be integrated into a PDA or a cell phone to protect user privacy. RFID Guardian is actually a user privacy protection platform in RFID systems. It can also work as an RFID reader to request information from RFID tags, or it can work like a tag to communicate with a reader. RFID Guardian has four different security properties: auditing, key management, access control, and authentication. It can audit RFID readers in its vicinity and record information about the RFID readers, such as commands, related parameters, and data, and provide these kinds of information to the user. Using this information, the user can sufficiently identify illegal scanning. In some cases, a user might not know or could forget the tags in his vicinity. With the help of RFID Guardian, the user can detect all the tags within radio range. Then the user can deactivate the tags according to his choice.

For RFID tags that use cryptography methods to provide security, one important issue is key management. RFID Guardian can perform two-way RFID communications and can generate random values. These features are very useful for key exchange and key refresh. Using the features of coordination of security primitives, context awareness, and tag-reader mediation, RFID Guardian can provide access control for RFID systems [44]. Also, using two-way RFID communication and standard challenge-response algorithms, RFID Guardian can provide off-tag authentication for RFID readers.

Another approach for privacy protecting is proposed by Juels and colleagues [45]. In this approach, an inexpensive, passive RFID tag is used as the blocker tag. Since this blocker tag can simulate many RFID tags at the same time, it is very difficult for an RFID reader to identify the real tag carried by the user. The blocker tag can both simulate all the possible RFID tags and simulate only a select set of the tags, making it convenient for the user to manage the RFID tags. For example, the user can tell the blocker tag to block only the tags that belong to a certain company. Another advantage of this approach is that if the user wants to reuse these RFID tags, unlike the "killed" tags that need to be activated by the user, the user need only remove the blocker tag. Since the blocker tag can shield the serial numbers of the tags from being read by RFID readers, it can also be used by attackers to disrupt proper operation of an RFID system. A thief can also use the blocker tag to shield the tags attached to the commodities in shops and take them out without being detected.

RFID System Using Symmetric-Key Cryptography

Symmetric-key cryptography, also called *secret-key cryptography* or *single-key cryptography,* uses a single key to perform both encryption and decryption. Due to the limited amount of resources available on an RFID chip, most available symmetric-key cryptographs are too costly to be implemented on an RFID chip. For example, a typical implementation of Advanced Encryption Standard (AES) needs about 2000–3000 gates. This is not appropriate for low-cost RFID tags. It is only possible to implement AES in high-end RFID tags. A successful case of implementing a 128-bit AES on high-end RFID tags has been reported [46].

Using the Symmetric Key to Provide Authentication and Privacy

Symmetric-key cryptography can be applied to prevent tag cloning in RFID systems using a challenge and response protocol. For example, if a tag shares a secret key K with a reader and the tag wants to authenticate itself to the reader, it will first send its identity to the reader. The reader will then generate a nonce N and send it to the tag. The tag will use this nonce and the secret K to generate a hash code $H = h(K,N)$ and send this hash code to the reader. The reader can also generate a hash code $H' = h(K,N)$ and compare these two codes to verify this tag. Using this scheme, it is difficult for an attacker to clone the tags without knowing the secret keys.

Different kinds of symmetric-key cryptography protocol-based RFID tags have been used recently in daily life. For example, an RFID device that uses this symmetric-key challenge-response protocol, called a digital signature transponder, has been introduced by Texas Instruments. This transponder can be built into cars to prevent car theft and can be implemented into wireless payment devices used in filling stations.

One issue of RFID systems that use symmetric-key cryptography is key management. To authenticate itself to an RFID reader, each tag in the system should share a different secret key with the reader, and the reader needs to keep all the keys of these tags. When a tag wants to authenticate itself to an RFID reader, the reader needs to know the secret key shared between them. If the reader does not know the identification of the tag in advance, it cannot determine which key can be used to authenticate this tag. If the tag sends its identification to the reader before the authentication for the reader to search the secret key, the privacy of the tag cannot be protected, since other readers can also obtain the identification of this tag.

To tackle this problem, one simple method is *key searching*. The reader will search all the secret keys in its memory to find the right key for the tag before authentication. There are some protocols proposed for the key search for RFID tags. One general kind of key search scheme [47] has been proposed. In this approach, the tag first generates a random nonce N and hashes this N using its secret key K to generate the hash code. Then it sends both this hash code and N to the reader. Using this nonce N, the reader will generate the hash code with all the secret keys and compare them with the received hash code from the tag. If there is a match, it means it found the right key. In this scheme, since the nonce N is generated randomly every time, the privacy of the tag can be protected.

The problem with this approach is that if there are a large number of tags, the key searching will be very costly. To reduce the cost of the key searching, a modification of this scheme was proposed [48]; in Haehnel et al. [20], in this approach, a scheme called *tree of secret* is used. Every tag is assigned to a leaf in the tree, and every node in the tree has its secret key. This way the key search cost for each tag can be reduced, but it will add some overlap to the sets of keys for each tag.

Another approach to reduce the cost of key searching is for the RFID tags and RFID reader to keep synchronization with each other. In this kind of approach, every tag will maintain a counter for the reader query times. For each reader's query, the tag should respond with a different value. The reader will also maintain counters for all the tags' responses and maintain a table of all the possible response values. Then, if the reader and tags can keep synchronization, the reader can know the approximate current counter number of the tags. When the reader receives a response from a tag, it can search the table and quickly identify this tag.

Other Symmetric-Key Cryptography-Based Approaches

In addition to the basic symmetric-key challenge-response protocol, some symmetric-key cryptography-based approaches have been proposed recently to protect the security and privacy of RFID systems.

One approach is called YA-TRAP: Yet Another T rivial RFID Authentication Protocol, proposed by Tsudik [49]. In this approach, a technique for the inexpensive untraceable identification of RFID tags is introduced. Here *untraceable* means it is computationally difficult to gather the information about the identity of RFID tags from the interaction with them. In YA-TRAP, for the purpose of authentication, only minimal communication between the reader and tags is needed, and the computational burden on the back-end server is very small.

The back-end server in the system is assumed to be secure and maintains all tag information. Each tag should be initialized with three values: K_i, T_0, and T_{max}. K_i is both the identifier and the cryptographic key for this tag. The size of K_i depends on the number of tags and the secure authentication requirement; in practice, 160 bits is enough. T_0 is the initial timestamp of this tag. The value of T_0 of each tag does not need to vary. This means that a group of tags can have the same T_0. T_{max} is the maximum value of T_0, and a group of tags can also have the same T_{max} value. In addition, each tag has a seeded pseudorandom number generator.

YA-TRAP works as follows: First, each tag should store a timestamp T_t in its memory. When an RFID reader wants to interrogate a RFID tag, it will send the current timestamp T_r to this tag. Receiving T_r, the tag will compare T_r with the timestamp value it stores and with T_{max}. If $T_r < T_t$ or $T_r > T_{max}$, this tag will respond to the reader with a random value generated by the seeded pseudo random number generator. Otherwise, the tag will replace T_t with T_r and calculate $H_r = HMACK_i(T_t)$, and then send H_r to the reader. Then the reader will send T_r and H_r to the back-end server. The server will look up its database to find whether this tag is a valid tag. If it's not, the server will send a tag-error message to the reader. If this is a valid tag, the server will send the meta-ID of this tag or the valid message to the reader, according to different application requirements. Since the purpose of this protocol is to minimize the interaction between the reader and tags and minimize the computation burden of the back-end server, it has some vulnerability. One of them is that the adversary can launch a DoS attack to the tag. For example, the attack can send a timestamp $t < T_{max}$, but this t is wildly inaccurate with the current time. In this case, the tag will update its timestamp with the wrong time and the legal reader cannot get access to this tag.

In Jechlitschek [33], another approach called *deterministic hash locks* [50] was proposed. In this scheme, the security of RFID systems is based on the one-way hash function. During initialization, every tag in the system will be given a meta-ID, which is the hash code of a random key. This meta-ID will be stored in the tag's memory. Both the meta-ID and the random key are also stored in the back-end server. After initialization, all the tags will enter the locked state. When they stay in the locked state, tags will respond only with the meta-ID when interrogated by an RFID reader. When a legitimate reader wants to unlock a tag, as shown in Figure 13.2, it will first send a request to the tag. After receiving the meta-ID from the tag, the reader will send this meta-ID to the back-end server. The back-end server will search in its database using this meta-ID to get the random key. Then it will send this key to the reader, and the reader will send it to the tag. Using this random key, the tag will hash this key and compare the hash code with its meta-ID. The tag will unlock itself and send its actual identification to the reader if these two values match. Then the tag will return to the locked state to prevent hijacking of illegal readers. Since the illegal reader cannot contact the back-end server to get the random key, it cannot get the actual identification of the tag.

One problem with deterministic hash locks is that when the tag is queried, it will respond with its meta-ID. Since the meta-ID of the tag is a static one and cannot change, the tag can be tracked easily. To solve this problem, Weis, Sarma, Rivest, and Engels proposed the Randomized Hash Locks protocol to prevent tracking of the tag. In this protocol, each tag is equipped with not only the one-way hash function but also a random number generator.

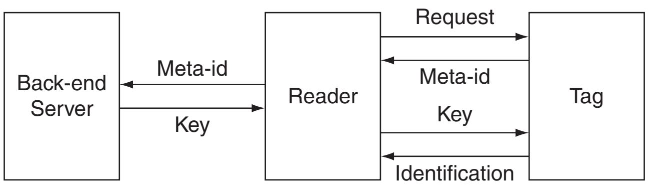


Figure 13.2: Tag unlock.

When the tag is requested by a reader, it will use the random number generator to generate a random number and will hash this random number together with its identification. The tag will respond with this hash code and the random number to the reader. After receiving this response from the tag, the reader will get all identifications of the tags from the back-end server. Using these identifications, the reader will perform brute-force search by hashing the identification of each tag together with the random number and compare the hash code. If there is a match, the reader can know the identification of the tag. In this approach, the tag response to the reader is not dependent on the request of the reader, which means that the tag is vulnerable to replay attack. To avoid this, Juels and Weis proposed a protocol called Improved Randomized Hash-Locks [51].

RFID System Using Public-Key Cryptography

Symmetric-key cryptography can provide security for RFID systems, but it is more suitable to be implemented in a closed environment. If the shared secret keys between them are leaked, it will impose a big problem for the security of RFID systems. Public-key cryptography is more suitable for open systems, since both RFID readers and tags can use their public keys to protect the security of RFID systems. In addition, using public-key cryptography can not only prevent leakage of any information to the eavesdropper attack during communication between reader and tags, it also can provide digital signatures for both the readers and tags for authentication. In the public-key cryptography system, an RFID reader does not need to keep all the secret keys for each tag and does not need to search the appropriate key for each tag as it does in a symmetric-key cryptography system. This will reduce the system burden for key management. Although public-key cryptography has some advantages over symmetric-key cryptography, it is commonly accepted that public-key cryptography is computationally more expensive than symmetric-key cryptography. Because of the limitations of memory and computational power of the ordinary RFID tags, it is difficult for the public-key cryptography to be implemented in RFID systems. In recent years, some research shows that some kinds of public key-based cryptographies such as elliptic curve cryptography and hyperelliptic curve cryptography are feasible to be implemented in high-end RFID tags [52].

Authentication with Public-Key Cryptography

Basically, there are two different kinds of RFID tag authentication methods using public-key cryptography: one is online authentication and the other is offline authentication [53].

For the authentication of RFID tags in an online situation, the reader is connected with a database server. The database server stores a large number of challenge-response pairs for each tag, making it difficult for the attacker to test all the challenge-response pairs during a limited time period. During the challenge-response pairs enrollment phase, the physical uncloneable function part of RFID systems will be challenged by a Certification Authority with a variety of challenges, and accordingly it will generate responses for these challenges. The physical uncloneable function is embodied in a physical object and can give responses to the given challenges [54]. Then these generated challenge-response pairs will be stored in the database server.

In the authentication phase, when a reader wants to authenticate a tag, first the reader will send a request to the tag for its identification. After getting the ID of the tag, the reader will search the database server to get a challenge-response pair for this ID and send the challenge to the tag. After receiving the challenge from the reader, the tag will challenge its physical uncloneable function to get a response for this challenge and then send this response to the reader. The reader will compare this received response with the response stored in the database server. If the difference between these two responses is less than a certain predetermined threshold, the tag can pass the authentication. Then the database server will remove this challenge-response pair for this ID.

One paper [55] details how the authentication of RFID tags works in an offline situation using public key cryptography. To provide offline authentication for the tags, a PUF-Certificate-Identify-based identification scheme is proposed. In this method, a standard identification scheme and a standard signature scheme are used. Then the security of RFID systems depends on the security of the PUF, the standard identification scheme, and the standard signature scheme. For the standard identification scheme, an elliptic curve discrete log based on Okamoto's Identification protocol [56] is used. This elliptic curve discrete log protocol is feasible to be implemented in the RFID tags.

Identity-Based Cryptography Used in RFID Networks

An identity-based cryptographic scheme is a kind of public-key-based approach that was first proposed by Shamir [57] in 1984. To use identity-based cryptography in RFID systems, since both the RFID tags and the reader have their identities, it is convenient for them to use their own identities to generate their public keys. An RFID system based on identity-based cryptography should be set up with the help of a PKG. When the reader and tags enter the system, each of them is allocated a unique identity stored in their memory. The process of key generation and distribution in the RFID system that uses identity-based cryptography is shown in Figure 13.3 and is outlined here.

1. PKG generates a "master" public key PU_{pkg} and a related "master" private key PR_{pkg} and saves them in its memory.
2. The RFID reader authenticates itself to the PKG with its identity ID_{re}.
3. If the reader can pass the authentication, PKG generates a unique private key PR_{re} for the reader and sends this private key together with PU_{pkg} to reader.

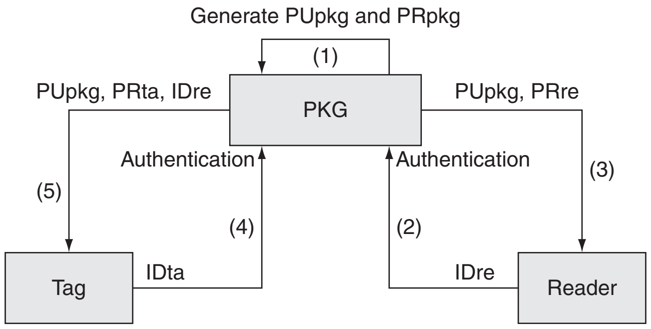


Figure 13.3: Key generation and distribution.

4. When an RFID tag enters the system, it authenticates itself to the PKG with its identity ID_{ta}.
5. If the tag can pass the authentication, PKG generates a unique private key PR_{ta} for the tag and sends PR_{ta} together with PU_{pkg} and the identity of the reader ID_{re} to the tag.

After this process, the reader can know its private key PR_{re} and can use PU_{pkg} and its identity to generate its public key. Every tag entered into the system can know its own private key and can generate a public key of its own and a public key of the reader.

If an RFID tag is required to transmit messages to the reader in security, since the tag can generate the reader's public key PU_{re}, it can use this key PU_{re} to encrypt the message and transmit this encrypted message to the reader. As shown in Figure 13.4, after receiving the message from the tag, the reader can use its private key PR_{re} to decrypt the message. Since only the reader can know its private key PR_{re}, the security of the message can be protected.

Figure 13.5 illustrates the scheme for the reader to create its digital signature and verify it. First, the reader will use the message and the hash function to generate a hash code, and then it uses its private key PR_{re} to encrypt this hash code to generate the digital signature and attach it to the original message and send both the digital signature and message to the tag. After receiving them, the RFID tag can use the public key of the reader PU_{re} to decrypt the digital signature to recover the hash code. By comparing this hash code with the hash code generated from the message, the RFID tag can verify the digital signature.

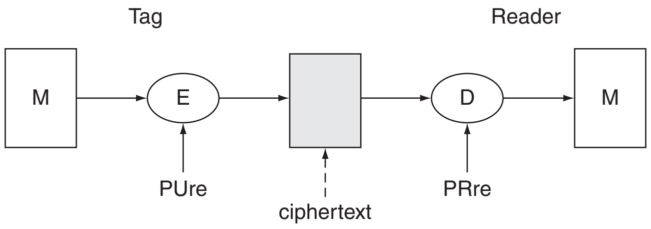


Figure 13.4: Message encryption.

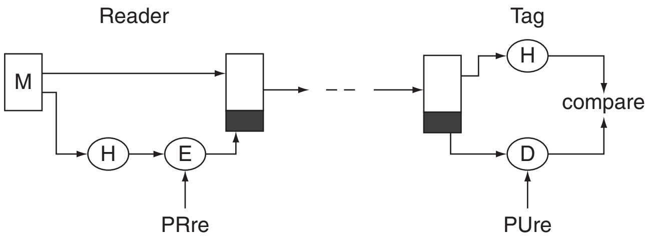


Figure 13.5: Digital signature from a reader.

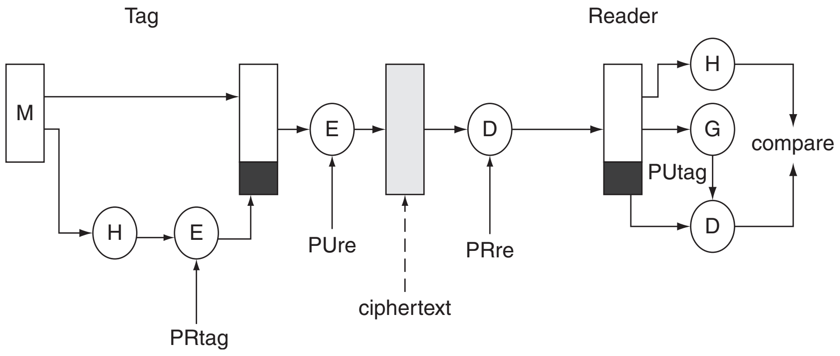


Figure 13.6: Digital signature from a tag.

Figure 13.6 illustrates the scheme for the RFID tag to create its digital signature and verify it. In RFID systems, the reader cannot know the identity of the tag before reading it from the tag. The reader cannot generate the public key of the tag, so the general protocol used in identity-based networks cannot be used here. In our approach, first, the tag will use its identity and its private key PR_{ta} to generate a digital signature. When the tag needs to authenticate itself to the reader, it will add this digital signature to its identity, encrypt it with the public key of the reader PU_{re}, and send to the reader; only the reader can decrypt this ciphertext and get the identity of the tag and the digital signature. Using the tag identity, the reader can generate the tag's public key PU_{ta}. Then the reader can use this public key to verify the digital signature.

As mentioned, the most important problem for the symmetric-key approach in RFID systems is the key management. The RFID tags need a great deal of memory to store all the secret keys related with each tag in the system for message decryption. Also, if the RFID reader receives a message from a tag, it cannot know which tag this message is from and therefore cannot know which key it can use to decrypt the message. The reader needs to search all the keys until it finds the right one. In RFID systems using identity-based cryptography, every tag can use the public key of the reader to generate the ciphertext that can be decrypted using the reader's private key, so the reader does not need to know the key of the tags; all it needs to keep is its own private key.

In some RFID applications such as epassports and visas, tag authentication is required. However, the symmetric-key approach cannot provide digital signatures for RFID tags to authenticate them to RFID readers. By using an identity-based scheme, the tags can generate digital signatures using their private keys and store them in the tags. When they need to authenticate themselves to RFID readers, they can transmit these digital signatures to the reader, and the reader can verify them using the tags' public keys.

In identity-based cryptography RFID systems, since the identity of the tags and reader can be used to generate public keys, the PKG does not need to keep the key directory, so it can reduce the resource requirements. Another advantage of using identity-based cryptography in RFID systems is that the reader does not need to know the public keys of the tags in advance. If the reader wants to verify the digital signature of an RFID tag, it can read the identity of the tag and use the public key generated from the identity to verify the digital signature.

An inherent weakness of identity-based cryptography is the key escrow problem. But in RFID systems that use identity-based cryptography, because all the devices can be within one company or organization, the PKG can be highly trusted and protected, and the chance of key escrow can be reduced.

Another problem of identity-based cryptography is revocation. For example, people always use their public information such as their names or home addresses to generate their public key. If their private keys are compromised by an attacker, since their public information cannot be changed easily, this will make it difficult to regenerate their new public keys. In contrast, in RFID systems the identity of the tag is used to generate the public key. If the private key of one tag has been compromised, the system can allocate a new identity to the tag and use this new identity to effortlessly create a new private key to the tag.

References

[1] Weis SA. Security and Privacy in Radio-Frequency Identification Devices.
[2] Langheinrich M. RFID and Privacy.
[3] Auto-ID Center, Draft Protocol Specification for a Class 0 Radio Frequency Identification Tag, February 2003.
[4] Finkenzeller K. RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification.
[5] Peris-Lopez P, Hernandez-Castro JC, Estevez-Tapiador J, Ribagorda A. RFID systems: A survey on security threats and proposed solutions. In: 11th IFIP International Conference on Personal Wireless Communications – PWC06, Vol. 4217 of Lecture Notes in Computer Science. Springer-Verlag; 2006. p. 159–70.
[6] RFID Handbook. 2nd ed. J. Wiley & Sons.
[7] Phillips T, Karygiannis T, Huhn R. Security standards for the RFID market. IEEE Security & Privacy (November/December 2005); 85–9.
[8] RFID Handbook. 2nd ed. J. Wiley & Sons.
[9] RFID Handbook. 2nd ed. J. Wiley & Sons.

[10] Phillips T, Karygiannis T, Huhn R. Security standards for the RFID market. IEEE Security & Privacy (November/December 2005); 85–9.
[11] EPCglobal. www.epcglobalinc.org/, June 2005.
[12] Peris-Lopez P, Hernandez-Castro JC, Estevez-Tapiador J, Ribagorda A. RFID systems: a survey on security threats and proposed solutions. In: 11th IFIP International Conference on Personal Wireless Communications – PWC06, Vol. 4217 of Lecture Notes in Computer Science. Springer-Verlag; 2006. p. 159–70.
[13] Phillips T, Karygiannis T, Huhn R. Security standards for the RFID market. IEEE Security & Privacy 2005;85–9.
[14] EPCglobal Tag Data Standards. Version 1.3.
[15] EPCglobal. www.epcglobalinc.org/, June 2005.
[16] Guidelines for Securing Radio Frequency Identification (RFID) Systems, Recommendations of the National Institute of Standards and Technology, NIST Special Publication 800–98.
[17] Thompson DR, Chaudhry N, Thompson CW. RFID Security Threat Model.
[18] Weis S, Sarma S, Rivest R, Engels D. Security and privacy aspects of low-cost radio frequency identification systems. In: Stephan W, Hutter D, Muller G, Ullmann M, editors. International Conference on Security in Pervasive computing-SPC 2003, vol. 2802. Springer-Verlag; 2003. p. 454–69.
[19] Peris-Lopez P, Hernandez-Castro JC, Estevez-Tapiador J, Ribagorda A. RFID systems: a survey on security threats and proposed solutions. In: 11th IFIP International Conference on Personal Wireless Communications – PWC06, Vol. 4217 of Lecture Notes in Computer Science. Springer-Verlag; 2006. p. 159–70.
[20] Haehnel D, Burgard W, Fox D, Fishkin K, Philipose M. Mapping and localization with WID technology, International Conference on Robotics & Automation 2004.
[21] Thompson DR, Chaudhry N, Thompson CW. RFID Security Threat Model.
[22] Thompson DR, Chaudhry N, Thompson CW. RFID Security Threat Model.
[23] Juels A, Rivest RL, Syzdlo M. The blocker tag: selective blocking of RFID tags for consumer privacy. In: Atluri V, editor. 8th ACM Conference on Computer and Communications Security. 2003. p. 103–11.
[24] Juels A, Rivest RL, Syzdlo M. The blocker tag: selective blocking of RFID tags for consumer privacy. In: Atluri V, editor. 8th ACM Conference on Computer and Communications Security. 2003. p. 103–11.
[25] Thompson DR, Chaudhry N, Thompson CW. RFID Security Threat Model.
[26] Thompson DR, Chaudhry N, Thompson CW. RFID Security Threat Model.
[27] Thompson DR, Chaudhry N, Thompson CW. RFID Security Threat Model.
[28] Thornton F, Haines B, Das AM, Bhargava H, Campbell A, Kleinschmidt J. RFID Security.
[29] Jechlitschek C. A Survey Paper on Radio Frequency Identification (RFID) Trends.
[30] Weingart SH. Physical Security Devices for Computer Subsystems: A Survey of Attacks and Defenses.
[31] Weis SA. Security and Privacy in Radio-Frequency Identification Devices.
[32] Jechlitschek C. A Survey Paper on Radio Frequency Identification (RFID) Trends.
[33] ibid.
[34] Sarma SE, Weis SA, Engels DW. RFID systems security and privacy implications, Technical Report, MITAUTOID-WH-014, AutoID Center, MIT, 2002.
[35] Inoue S, Yasuura H. RFID privacy using user-controllable uniqueness. In: RFID Privacy Workshop. MIT, November 2003.
[36] Good N, Han J, Miles E, Molnar D, Mulligan D, Quilter L, et al. Radio frequency ID and privacy with information goods. In: Workshop on Privacy in the Electronic Society (WPES). 2004.
[37] Good N, Han J, Miles E, Molnar D, Mulligan D, Quilter L, et al. Radio frequency ID and privacy with information goods. In: Workshop on Privacy in the Electronic Society (WPES). 2004.

[38] Juels A. Minimalist cryptography for low-cost RFID tags. In: Blundo C, Cimato S, editors. The Fourth International Conference on Security in Communication Networks – SCN 2004, Vol. 3352 of Lecture Notes in Computer Science. Springer-Verlag; 2004. p. 149–64.

[39] Juels A, Pappu R. Squealing euros: privacy protection in RFID-enabled banknotes. In: Wright R, editor. Financial Cryptography '03 vol. 2742. Springer-Verlag; 2003. p. 103–21.

[40] Golle P, Jakobsson M, Juels A, Syverson P. Universal re-encryption for mixnets. In: Okamoto T, editor. RSA Conference-Cryptographers' Track (CT-RSA), vol. 2964. 2004. p. 163–78.

[41] Ateniese G, Camenisch J, de Madeiros B. Untraceable RFID tags via insubvertible encryption. In: 12th ACM Conference on Computer and Communication Security. 2005.

[42] Floerkemeier C, Schneider R, Langheinrich M. Scanning with a Purpose Supporting the Fair Information Principles in RFID Protocols. 2004.

[43] Rieback MR, Crispo B, Tanenbaum A. RFID Guardian: a battery-powered mobile device for RFID privacy management. In: Boyd C, Gonz'alez Nieto JM, editors. Australasian Conference on Information Security and Privacy – ACISP. 2005, Vol. 3574 of Lecture Notes in Computer Science. Springer-Verlag; 2005. p. 184–94.

[44] Rieback MR, Crispo B, Tanenbaum A. RFID guardian: a battery-powered mobile device for RFID privacy management. In: Boyd C, Gonz'alez Nieto JM, editors. Australasian Conference on Information Security and Privacy – ACISP 2005, Vol. 3574 of Lecture Notes in Computer Science. Springer-Verlag; 2005. p. 184–94.

[45] Juels A, Rivest RL, Syzdlo M. The blocker tag: selective blocking of RFID tags for consumer privacy. In: Atluri V, editor. 8th ACM Conference on Computer and Communications Security. 2003. p. 103–11.

[46] Feldhofer M, Dominikus S, Wolkerstorfer J. Strong authentication for RFID systems using the AES algorithm. In: Joye M, Quisquater JJ, editors. Workshop on Cryptographic Hardware and Embedded Systems CHES 04, Vol. 3156 of Lecture Notes in Computer Science. Springer-Verlag; 2004. p. 357–70.

[47] Weis S, Sarma S, Rivest R, Engels D. Security and privacy aspects of low-cost radio frequency identification systems. In: Stephan W, Hutter D, Muller G, Ullmann M, editors. International Conference on Security in Pervasive computing-SPC, vol. 2802. Springer-Verlag; 2003. p. 454–69.

[48] Molnar D, Wagner D. Privacy and security in library RFID: issues, practices, and architectures. In: Pfitzmann B, McDaniel P, editors. ACM Conference on Communications and Computer Security. ACM Press; 2004. p. 210–9.

[49] Tsudik G. YA-TRAP: Yet another trivial RFID authentication protocol. In: Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06). 2006. p. 640–3.

[50] Weis S, Sarma S, Rivest R, Engels D. Security and privacy aspects of low-cost radio frequency identification systems. In: Stephan W, Hutter D, Muller G, Ullmann M, editors. International Conference on Security in Pervasive computing-SPC, vol. 2802. Springer-Verlag; 2003. p. 454–69.

[51] Juels A, Weis S. Defining strong privacy for RFID. Pervasive Computing and Communications Workshops. 2007.

[52] Tuyls P, Batina L. RFID tags for anticounterfeiting. In: Pointcheval D, editor. Topics in Cryptology-CT-RSA. Springer-Verlag; 2006.

[53] Batina L, Guajardo J, Kerins T, Mentens N, Tuyls P, Verbauwhede I. Public-key cryptography for RFID-tags. Printed handout of Workshop on RFID Security," RFIDSec06, 2006; 61–76.

[54] Tuyls P, Batina L. RFID tags for anticounterfeiting. In: Pointcheval D, editor. Topics in Cryptology-CT-RSA. Springer-Verlag; 2006.

[55] Tuyls P, Batina L. RFID tags for anticounterfeiting. In: Pointcheval D, editor. Topics in Cryptology-CT-RSA. Springer-Verlag; 2006.

[56] Okamoto T. Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell EF, editor. Advances in Cryptology | CRYPTO'92, Vol. 740 of LNCS. Springer-Verlag; 1992. p. 31–53.
[57] Shamir A. Identity-based cryptosystems and signature scheme, Advances in Cryptology. Proceedings of CRYPTO 84, LNCS, 1984. p. 47–53.

Index