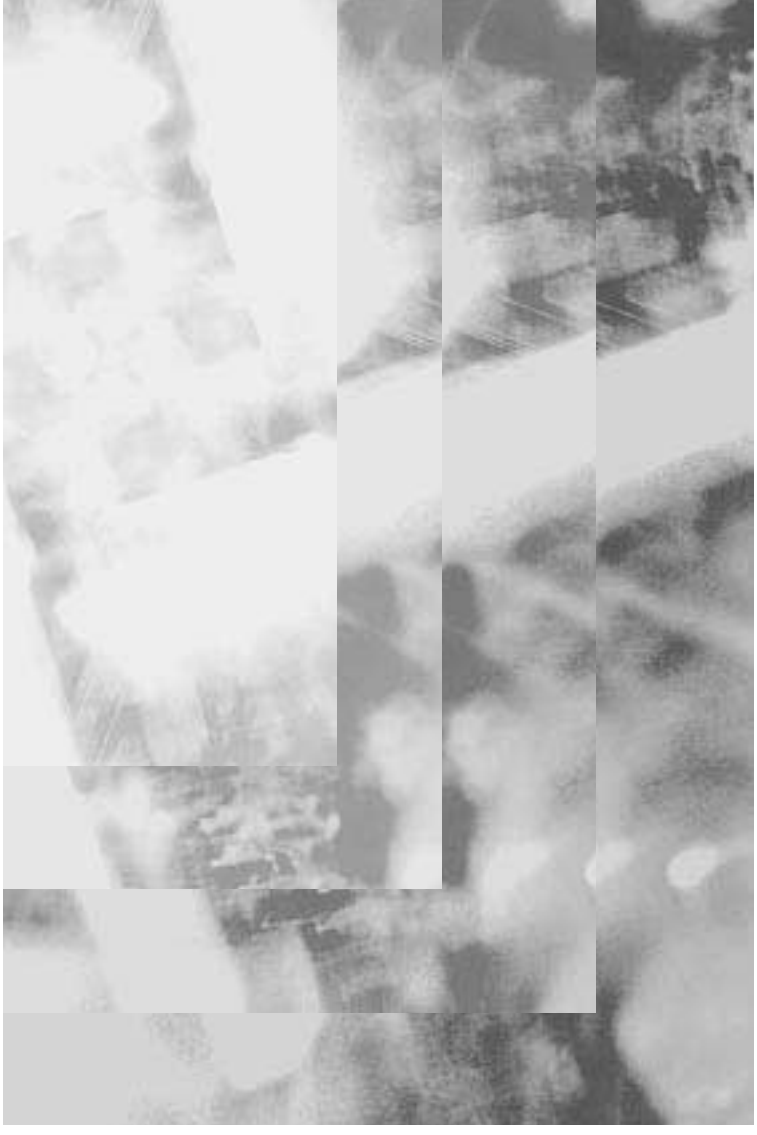


E-MAIL VIRUS PROTECTION HANDBOOK

SYNGRESS®

KEY	SERIAL NUMBER
001	9TM1L2ADSE
002	XPS1697TC4
003	CLNKK98FV7
004	DC5EPL4RL6
005	Z74DQ81524
006	PJ62NT41NB
007	4W2VANZX44
008	V8DF743RTD
009	65Q2M94ZTS
010	SM654PSMRN

PUBLISHED BY
Syngress Publishing, Inc.
800 Hingham Street
Rockland, MA 02370

E-mail Virus Protection Handbook

Printed in the United States of America

1 2 3 4 5 6 7 8 9 0

ISBN: 1-928994-23-7

Copy edit by: Eileen Kramer
Technical edit by: James Stanger
Index by: Rober Saigh
Project Editor: Katharine Glennon

Proofreading by: Adrienne Rebello
Technical Review by: Stace Cunningham
Page Layout and Art by: Shannon Tozier
Co-Publisher: Richard Kristof

Distributed by Publishers Group West

Acknowledgments

We would like to acknowledge the following people for their kindness and support in making this book possible.

Richard Kristof, Duncan Anderson, Jennifer Gould, Robert Woodruff, Kevin Murray, Dale Leatherwood, Shelley Everett, Laurie Hedrick, Rhonda Harmon, Lisa Lavallee, and Robert Sanregret of Global Knowledge, for their generous access to the IT industry's best courses, instructors and training facilities.

Ralph Troupe and the team at Rt. 1 Solutions for their invaluable insight into the challenges of designing, deploying and supporting world-class enterprise networks.

Karen Cross, Kim Wylie, Harry Kirchner, John Hays, Bill Richter, Kevin Votel, Brittin Clark, Sarah Schaffer, Luke Kreinberg, Ellen Lafferty and Sarah MacLachlan of Publishers Group West for sharing their incredible marketing experience and expertise.

Peter Hoenigsberg, Mary Ging, Caroline Hird, Simon Beale, Julia Oldknow, Kelly Burrows, Jonathan Bunkell, Catherine Anderson, Peet Kruger, Pia Rasmussen, Denelise L'Ecluse, Rosanna Ramacciotti, Marek Lewinson, Marc Appels, Paul Chrystal, Femi Otesanya, and Tracey Alcock of Harcourt International for making certain that our vision remains worldwide in scope.

Special thanks to the professionals at Osborne with whom we are proud to publish the best-selling Global Knowledge Certification Press series.

From Global Knowledge

At Global Knowledge we strive to support the multiplicity of learning styles required by our students to achieve success as technical professionals. As the world's largest IT training company, Global Knowledge is uniquely positioned to offer these books. The expertise gained each year from providing instructor-led training to hundreds of thousands of students worldwide has been captured in book form to enhance your learning experience. We hope that the quality of these books demonstrates our commitment to your lifelong learning success. Whether you choose to learn through the written word, computer based training, Web delivery, or instructor-led training, Global Knowledge is committed to providing you with the very best in each of these categories. For those of you who know Global Knowledge, or those of you who have just found us for the first time, our goal is to be your lifelong competency partner.

Thank your for the opportunity to serve you. We look forward to serving your needs again in the future.

Warmest regards,

Duncan Anderson
President and Chief Executive Officer, Global Knowledge

Contributors

Philip Baczewski is the Associate Director of Academic Computing Services at the University of North Texas Computing Center. He serves as project manager for university student Internet services, and works with client server implementations of IMAP, IMSP, SMTP, and LDAP protocols. Philip also provides technical consultation support in the areas of mainframe and UNIX programming, data management, electronic mail, and Internet services. Philip holds his Doctorate in Musical Arts, Composition from the University of North Texas.

Brian Bagnall is a Sun Certified Java Programmer and Developer. His current project is designing and programming a distributed computing effort for Distco.com. Brian would like to say thanks to Deck Reyes for his help with the material. He would also like to thank his family for their support. Contact Brian at bbagnall@escape.ca.

Chris O. Broomes (MCSE, MCP+I, MCT, CCNA) has over seven years of networking experience. He started his career as a consultant at Temple University, and has worked with organizations such as Morgan, Lewis & Bockius, Temple University Dental School, and Dynamic Technologies, Inc. Currently, Chris works in Philadelphia as a Network Administrator at EXE Technologies, Inc., a global provider of business-to-business e-fulfillment solutions.

Patrick T. Lane (MCSE, MCP+I, MCT, CIW Foundations, CIW Server Administrator, CIW Internetworking Professional, and CompTIA Network+ and i-Net+) is a Content Architect for ProsoftTraining.com who assisted in the creation of the Certified Internet Webmaster (CIW) program. He holds a Master's degree in Education. Lane began working with computers in 1984, and has developed curriculum and trained students across the computer industry since 1994. He is the author of more than 20 technical courses, the director of the CIW Foundations and CIW Internetworking Professional series, and a member of the CompTIA Network+ Advisory Committee. Lane's work has been published in six languages, and he has been a featured speaker at Internet World.

Michael Marfino is the IS Operations Manager for EDS in Las Vegas, Nevada. He earned a Bachelor's of Science degree in Management Information Systems from Canisius College in Buffalo, N.Y. He has over a decade of technical industry experience, working in hardware/software support, e-mail administration, system administration, network administration, and IT management. His tenure includes positions at MCI Worldcom and Softbank.

Eriq Oliver Neale is a full-time computing technology professional, part-time author and teacher, and occasional musician. He has worked in the computer support industry for over 13 years, and has been on the anti-virus bandwagon since before Michelangelo hit the national media. His recommendations for practicing "safe hex" have been presented in numerous articles and seminars. Eriq lives in the North Texas area with his wife and their two dogs, seven cats, and a school of Mollies that are reproducing faster than believed possible. Eriq has been known to teach the occasional class in web development and attend major league baseball games when not otherwise occupied.

Ryan Russell (CCNA, CCNP) has been been employed in the networking field for over ten years, including more than five years working with Cisco equipment. He has held IT positions ranging from help desk support to network design, providing him with a good perspective on the challenges that face a network manager. Recently, Ryan has been doing mostly information security work involving network security and firewalls. He has completed his CCNP, and holds a Bachelor's of Science degree in computer science.

Henk-Evert Sonder (CCNA) has about 15 years of experience as an Information and Communication Technologies (ICT) professional, building and maintaining ICT infrastructures. In recent years he has specialized in integrating ICT infrastructures with business applications and the security that comes with it. His mission is to raise the level of companies security awareness about their networks. According to Henk, "So many people talk about the security threats coming from the Internet, but they can forget that the threats from within are equally dangerous." Currently he works as a senior consultant for a large Dutch ICT solutions provider. His own company, IT Selective, helps retailers get e-connected.

Technical Editor

James Stanger (Ph.D., MCSE, MCT, CIW Security Professional) is a writer and systems analyst currently living in Washington State, where he works for ProsoftTraining.com's research and development department. He also consults for companies such as Axent, IBM, DigitalThink, and Evinci concerning attack detection and analysis. In addition to Windows 2000 and Linux security issues, his areas of expertise include e-mail and DNS server security, firewall and proxy server deployment, and securing Web servers in enterprise environments. He is currently an acting member of the Linux Professional Institute (LPI), Linux+, and Server+ advisory boards, and leads development concerning the Certified Internet Webmaster security certification. A prolific author, he has written titles concerning network security auditing, advanced systems administration, network monitoring with SNMP, I-Net+ certification, Samba, and articles concerning William Blake, the nineteenth-century British Romantic poet and artist. When not writing or consulting, he enjoys bridge and cliff jumping, preferably into large, deep bodies of water.

Technical Reviewer

Stace Cunningham (CCNA, MCSE, CLSE, COS/2E, CLSI, COS/2I, CLSA, MCPS, A+) is a Systems Engineer with SDC Consulting located in Biloxi, MS. SDC Consulting specializes in the design, engineering, and installation of networks. Stace is also certified as an IBM Certified LAN Server Engineer, IBM Certified OS/2 Engineer, IBM Certified LAN Server Administrator, IBM Certified LAN Server Instructor, IBM Certified OS/2 Instructor. Stace has participated as a Technical Contributor for the IIS 3.0 exam, SMS 1.2 exam, Proxy Server 1.0 exam, Exchange Server 5.0 and 5.5 exams, Proxy Server 2.0 exam, IIS 4.0 exam, IEAK exam, and the revised Windows 95 exam.

In addition, he has coauthored or technical edited about 30 books published by Microsoft Press, Osborne/McGraw-Hill, and Syngress Media as well as contributed to publications from The SANS Institute and Internet Security Advisor magazine.

His wife Martha and daughter Marissa are very supportive of the time he spends with his computers, routers, and firewalls in the "lab" of their house. Without their love and support he would not be able to accomplish the goals he has set for himself.

Contents

Introduction

One of the lessons I learned early in life is to never confess the stupid things that I have done in public—unless there's a good punch line at the end of the story. Well, there is really no punch line at the end of the story I am about to tell you, but I am going to tell it anyway, because it helps introduce some of the key issues and concepts involved when securing e-mail clients and servers.

In 1994, I was browsing the Web with my trusty version of Netscape Navigator (version 1.0—yes, the one that ran just great on a Windows 3.11 machine that screamed along on top of an ultra-fast 486 processor). While browsing, I found a Web page that was selling a really nifty Telnet client. This piece of software had everything: I could use Kermit, Xmodem, and Zmodem to transfer files, and it even allowed automatic redial in case of a dropped connection. I just *had* to have it, and I had to have it right away; there was no waiting for it to arrive via "snail mail." I wanted to download it immediately.

Things being the way they were in 1994, the site's Web page invited me to either call their 800 number, or e-mail my Visa information for quicker processing. I'm something of a night owl, and it was about 2:30 a.m., and no one was manning the phones at the time. Rather than wait, I naïvely decided to use my Eudora e-mail client and send my Visa card number and expiration date to the site.

Two things happened as a result of this choice: I received an e-mail message response right away, complete with an access code that allowed me to download the software. With my new purchase, I was able to use Telnet as no one had ever used it before. That was the good part. The second thing happened two days after I began Telnetting my way across the world: I received a phone call from my Visa card company, asking me if I had authorized the use of this card for $250.00 in telephone charges, and around $375.00 for shoes. I hadn't. Someone

was using my Visa card to make telephone calls to Hawaii and purchase really expensive Nike's.

Before I had a chance to say anything to the Visa customer service representative (my profound response to her was a long "uuuhhh…"), she informed me that my charges were nearly identical to several others, all of which belonged to users who had sent e-mail messages to a certain site on the Internet. I remember the way she said the words "e-mail" and "Internet," because she said them as if she had never seen nor heard the words before. I told her that yes, I had visited the site on the Internet, and that I had sent an e-mail message containing my Visa information. I also told her that I had not made any purchases on the card lately. She quickly reversed the charges, cancelled the card, and issued me a new one. As I hung up the phone, I remember feeling both grateful and frightened: I had just been the victim of an Internet hacker who had obtained my Visa information via e-mail, presumably by "sniffing" it as it passed across the Internet, or by breaking into the site itself.

Now, alas, you have probably lost all confidence in me, the technical editor for this book. You may feel just like a person who is about to embark on a three-day journey through the great woods of the Pacific Northwest with no one else but a thin, nervous Forest Service guide who has poison ivy rashes all over his face. After all, I have helped write this book, and yet I have fallen victim to a hacker. Some expert I must be, right? Well, in some ways, I don't blame you if you feel a bit nervous about this book, at least at first. I still sometimes ask myself what was I thinking when I clicked the Send button. How could I be so foolish? What was I thinking? How could I be so lucky that my credit card company contacted me about this incident, rather than the other way around? Do you have any idea about the kind of runaround I would get in trying to reverse these illicit charges if it was only my idea?

And that's just the beginning of the questions I asked myself on the day I found out I had been "hacked." Trust me: Most of the remaining questions I ask myself are pretty harsh. After all, sending important information without first encrypting it is, to put it bluntly, pretty silly. But one thing that helps me regain some sort of self-confidence is the knowledge that I learn quickly from my mistakes.

Nowadays, I congratulate myself by knowing exactly how I got hacked, and, even more important, how I can use today's cutting-edge technologies to help keep anything like this from ever happening again. I now understand how an e-mail message is passed from the end user's

client machine through e-mail servers across the Internet. I have, in essence, empowered myself with knowledge concerning how e-mail messages are sent, processed, and received. I didn't learn these things as a direct result of getting hacked. Still, it has been very helpful for me to think back to that incident as I subsequently learned about arcane bits of knowledge relevant to e-mail (the Simple Mail Transfer Protocol (SMTP), the Domain Name System (DNS), packet sniffing applications, and encryption, etc.).

As I think back to that incident, I consider another question that is really quite intriguing: What was it that made me almost immediately go back to my computer, fire up my e-mail client, and keep sending e-mail messages? After all, I had been hacked. Yet, as silly as I felt, I still needed to communicate via e-mail. The sheer speed, convenience, and usefulness of the medium made it far too important and compelling to stop using it.

End-users, power users, and systems administrators all use e-mail every day, in spite of the security problems found in current e-mail technologies. This book explains how to implement specific security measures for e-mail clients and servers that make communication via e-mail both secure and convenient. In this book, you will learn about the problems associated with e-mail, including specific attacks that malicious users, sometimes called hackers, can wage against e-mail servers. First, you will learn about how these attacks are waged, and why. Once you understand the hacker's perspective, you can then begin to approach your e-mail client and server software from a more informed perspective.

This book will show you how to encrypt e-mail messages using the freeware Pretty Good Privacy (PGP) application, one of the most successful software packages ever. You will also learn about problems associated with Web-based e-mail, and how to solve some of them by using more secure options. Later chapters discuss how to install and configure the latest anti-virus applications, and also how to install "personal firewall" software, which is designed to isolate your computer's operating system so that it is not as susceptible to attacks waged by malicious users.

Once this book has thoroughly discussed how to secure e-mail clients, it then turns to the server side. Remember, once you click the Send button, you then involve two types of e-mail servers: The first type is designed to send e-mail messages across the Internet. The second type is designed to store e-mail messages, then allow you to log in remotely in order to read and download them. In the second section,

you will learn how to harden the operating system so that it can properly house an e-mail server. You will then learn about how to protect your system against malicious code by invoking third-party software, which is designed to scan e-mail messages (and attachments) for malicious content.

This book is unique because it discusses the latest methods for securing both the e-mail client and the e-mail server from the most common threats. These threats include "sniffing" attacks that illicitly obtain e-mail message information, denial of service attacks, that attempt to crash e-mail clients and servers, and authentication-based attacks, that attempt to defeat the user names and passwords that we use every day to secure our systems. Time will not eliminate these threats. In fact, it is likely that these will become even more serious. As e-mail becomes even more central to business practice, you will find this book very handy as a desktop reference for installing the latest e-mail security software. Even after the software discussed in this book becomes outdated, you will find that the concepts and principles enacted in this book will remain timely and useful. This is the book that I wish I had back in 1994. With this book, I would have been able to use my nifty Telnet client with full peace of mind, because I would have waited until the proper technologies were available in order to send my confidential e-mail message.

The authors we have assembled for this book are all authorities in network security. They are a diverse group. Some of the authors are experts in creating public key encryption solutions and knowing how to harden an operating system so that it can safely house an e-mail server. Others are experienced software coders who have deep knowledge of just what malicious code can do. Some of the authors presented in this book are seasoned IT professionals, while others have had extensive contact with the very hackers that are currently lurking the Internet, looking for unwitting victims who have not yet bought and read this book (here's hoping you have bought this book, and have not checked it out from the library!).

As diverse as this group is, all have one thing in common: Each is sincere in the wish to teach you how to secure your system. Each has learned through extensive study and experience about the industry best practices to follow when deploying software solutions. What is more, each of these authors has taken the time to share insights. I hope you enjoy this book. I have enjoyed editing it, as well as contributing a chapter or two. After you have read this book, you will be able to encrypt your e-mails, scan for malicious code on both the client

and the server side, and thoroughly understand what happens when you click the Send button, or double-click an attachment.

So, as you read the Case Studies, all of which are provided as real-world examples from real-world companies, and as you thumb through the details provided in this book, consider that you are now able to take advantage of the shared wisdom of many different authors. It is even possible that some of them have made a few mistakes along the way, just so that you can benefit from the lessons they learned.

Chapter 1

Understanding the Threats: E-mail Viruses, Trojans, Mail Bombers, Worms, and Illicit Servers

Solutions in this chapter:

- **Sending and Receiving E-mail**
- **Understanding E-mail Attacks**
- **Identifying the Impact of a Sniffing Attack**
- **Protecting E-mail Clients and Servers**
- **Encrypting E-mail**

Introduction

E-mail is the essential killer application of the Internet. Although Web-based commerce, business to business (B2B) transactions, and Application Service Providers (ASPs) have become the latest trends, each of these technologies is dependent upon the e-mail client/server relationship. E-mail has become the "telephone" of Internet-based economy; without e-mail, a business today is as stranded as a business of 50 years ago that lost its telephone connection. Consider that 52 percent of Fortune 500 companies have standardized to Microsoft's Exchange Server for its business solutions (see http://serverwatch.internet.com/reviews/mail-exchange2000_1.html). Increasingly, e-mail has become the preferred means of conducting business transactions. For example, the United States Congress has passed the Electronic Signatures in Global and National Commerce Act. Effective October 2000, e-mail signatures will have the same weight as pen-and-paper signatures, which will enable businesses to close multi-billion dollar deals with properly authenticated e-mail messages. Considering these two facts alone, you can see that e-mail has become critical in the global economy. Unfortunately, now that businesses have become reliant upon e-mail servers, it is possible for e-mail software to become killer applications in an entirely different sense—if they're down, they can kill your business.

There is no clear process defined to help systems administrators, management, and end-users secure their e-mail. This is not to say that no solutions exist; there are many (perhaps even too many) in the marketplace—thus, the need for this book. In this introductory chapter, you will learn how e-mail servers work, and about the scope of vulnerabilities and attacks common to e-mail clients and servers. This chapter also provides a summary of the content of the book. First, you will get a brief overview of how e-mail works, and then learn about historical and recent attacks. Although some of these attacks, such as the Robert Morris Internet Worm and the Melissa virus, happened some time ago, much can still be learned from them. Chief among the lessons to learn is that systems administrators need to address system bugs introduced by software manufacturers. The second lesson is that both systems administrators and end-users need to become more aware of the default settings on their clients and servers. This chapter will also discuss the nature of viruses, Trojan horses, worms, and illicit servers.

This book is designed to provide real-world solutions to real-world problems. You will learn how to secure both client and server software from known attacks, and how to take a proactive stance against possible new attacks. From learning about encrypting e-mail messages with Pretty Good Privacy (PGP) to using anti-virus and personal firewall software, to

actually securing your operating system from attack, this book is designed to provide a comprehensive solution. Before you learn more about how to scan e-mail attachments and encrypt transmissions, you should first learn about some of the basics.

Essential Concepts

It is helpful to define terms clearly before proceeding. This section provides a guide to many terms used throughout this book.

Servers, Services, and Clients

A *server* is a full-fledged machine and operating system, such as an Intel system that is running the Red Hat 6.2 Linux operating system, or a Sparc system that is running Solaris 8. A *service* is a process that runs by itself and accepts network requests; it then processes the requests. In the UNIX/Linux world, a service is called a *daemon*. Examples of services include those that accept Web (HTTP, or Hypertext Transfer Protocol), e-mail, and File Transfer Protocol (FTP) requests. A client is any application or system that requests services from a server. Whenever you use your e-mail client software (such as Microsoft Outlook), this piece of software is acting as a client to an e-mail server. An entire machine can become a client as well. For example, when your machine uses the Domain Name System (DNS) to resolve human readable names to IP addresses when surfing the Internet, it is acting as a client to a remote DNS server.

Authentication and Access Control

Authentication is the practice of proving the identity of a person or machine. Generally, authentication is achieved by proving that you know some unique information, such as a user name and a password. It is also possible to authenticate via something you may have, such as a key, an ATM card, or a smart card, which is like a credit card, except that it has a specialized, programmable computer chip that holds information. It is also possible to authenticate based on fingerprints, retinal eye scans, and voice prints.

Regardless of method, it is vital that your servers authenticate using industry-accepted means. Once a user or system is authenticated, most operating systems invoke some form of access control. Any network operating system (NOS) contains a sophisticated series of applications and processes that enforce uniform authentication throughout the system. Do not confuse authentication with access control. Just because you get authenticated by a server at work does not mean you are allowed access to every

computer in your company. Rather, your computers maintain databases, called *access control lists*. These lists are components of complex subsystems that are meant to ensure proper access control, usually based on individual users and/or groups of users. Hackers usually focus their activities on trying to defeat these authentication and access control methods.

Now that you understand how authentication and access control works, let's review a few more terms.

Hackers and Attack Types

You are probably reading this book because you are:

1. Interested in protecting your system against intrusions from unauthorized users.
2. Tasked with defending your system against attacks that can crash it.
3. A fledgling hacker who wishes to learn more about how to crash or break into systems.

To many, a hacker is simply a bad guy who breaks into systems or tries to crash them so that they cannot function as intended. However, many in the security industry make a distinction between *white hat* hackers, who are benign and helpful types, and *black hat* hackers, who actually cross the line into criminal behavior, such as breaking into systems unsolicited, or simply crashing them. Others define themselves as *grey hat* hackers, in that they are not criminal, but do not consider themselves tainted (as a strict white hat would) by associating with black hats. Some security professionals refer to white hat hackers as *hackers*, and to black hat hackers as *crackers*. Another hacker term, *script kiddie*, describes those who use previously-written scripts from people who are more adept. As you might suspect, script kiddie is a derisive term.

Many professionals who are simply very talented users proudly refer to themselves as hackers, not because they break into systems, but because they have been able to learn a great deal of information over the years. These professionals are often offended by the negative connotation that the word hacker now has. So, when does a hacker become a cracker? When does a cracker become a benign hacker? Well, it all depends upon the perspective of the people involved. Nevertheless, this book will use the terms hacker, cracker, and malicious user interchangeably.

What Do Hackers Do?

Truly talented hackers know a great deal about the following:

1. Programming languages, such as C, C++, Java, Perl, JavaScript, and VBScript.
2. How operating systems work. A serious security professional or hacker understands not only how to click the right spot on an interface, but also understands what happens under the hood when that interface is clicked.
3. The history of local-area-network (LAN)- and Internet-based services, such as the Network File System (NFS), Web servers, Server Message Block (SMB, which is what allows Microsoft systems to share file and printing services), and of course e-mail servers.
4. Many hackers attack the protocols used in networks. The Internet uses Transmission Control Protocol/Internet Protocol (TCP/IP), which is a fast, efficient, and powerful transport and addressing method. This protocol is in fact an entire suite of protocols. Some of these include Telnet, DNS, the File Transfer Protocol (FTP), and all protocols associated with e-mail servers, which include the Simple Mail Transfer Protocol (SMTP), Post Office Protocol 3 (POP3), and the Internet Messaging Application Protocol (IMAP).
5. How applications interact with each other. Today's operating systems contain components that allow applications to "talk" to each other efficiently. For example, using Microsoft's Component Object Model (COM) and other technologies, one application, such as Word, can send commands to others on the local machine, or even on remote machines. Hackers understand these subtle relationships, and craft applications to take advantage of them.

A talented hacker can quickly create powerful scripts in order to exploit a system.

Attack Types

Don't make the mistake of thinking that hackers simply attack systems. Many different types of attacks exist. Some require more knowledge than others, and it is often necessary to conduct one type of attack before conducting another. Below is a list of the common attacks waged against all network-addressable servers:

- **Scanning** Most of the time, hackers do not know the nature of the network they wish to compromise or attack. By using TCP/IP programs such as ping, traceroute, and netstat, a hacker can learn about the physical makeup (topology) of a network. Once a hacker knows more about the machines, it is possible to attack or compromise them.

- **Denial of service (DoS)** This type of attack usually results in a crashed server. As a result, the server is no longer capable of offering services. Thus, the attack denies these services to the public. Many of the attacks waged against e-mail servers have been denial of service attacks. However, do not confuse a DoS attack with other attacks that try to gather information or obtain authentication information.
- **Sniffing and/or man-in-the-middle** This attack captures information as it flows between a client and a server. Usually, a hacker attempts to capture TCP/IP transmissions, because they may contain information such as user names, passwords, or the actual contents of an e-mail message. A sniffing attack is often classified as a man-in-the-middle attack, because in order to capture packets from a user, the machine capturing packets must lie in between the two systems that are communicating (a man-in-the-middle attack can also be waged on one of the two systems).
- **Hijacking and/or man-in-the-middle** Another form of a man-in-the-middle attack is where a malicious third party is able to actually take over a connection as it is being made between two users. Suppose that a malicious user wants to gain access to machine A, which is beginning a connection with machine B. First, the malicious user creates a denial of service attack against machine B; once the hacker knocks machine B off of the network, he or she can then assume that machine's identity and collect information from machine A.
- **Physical** Thus far, you have learned about attacks that are waged from one remote system to another. It is also possible to walk up to the machine and log in. For example, how many times do you or your work-mates simply walk away from a machine after having logged in? A wily hacker may be waiting just outside your cubicle to take over your system and assume your identity. Other, more sophisticated, attacks involve using specialized floppy disks and other tools meant to defeat authentication.
- **System bug/back door** No operating system, daemon, or client is perfect. Hackers usually maintain large databases of software that have problems that lead to system compromise. A system bug attack takes advantage of such attacks. A back door attack involves taking advantage of an undocumented subroutine or (if you are lucky) a password left behind by the creator of the application. Most back doors remain unknown. However, when they are discovered, they can lead to serious compromises.

- **Social engineering** The motto of a good social engineer is: Why do all the work when you can get someone else to do it for you? *Social engineering* is computer-speak for the practice of conning someone into divulging too much information. Many social engineers are good at impersonating systems administrators. Another example of social engineering is the temporary agency that is, in reality, a group of highly skilled hackers who infiltrate companies in order to conduct industrial espionage.

Overview of E-mail Clients and Servers

When you click on a button to receive an e-mail message, the message that you read is the product of a rather involved process. This process involves at least two protocols, any number of servers, and software that exists on both the client and the server side. Suppose that you want to send an e-mail to a friend. You generate the message using client software, such as Microsoft Outlook, Netscape Messenger, or Eudora Pro. Once you click the Send button, the message is sent to a server, which then often has to communicate with several other servers before your message is finally delivered to a central server, where the message waits. Your friend then must log in to this central server and download the message to read it.

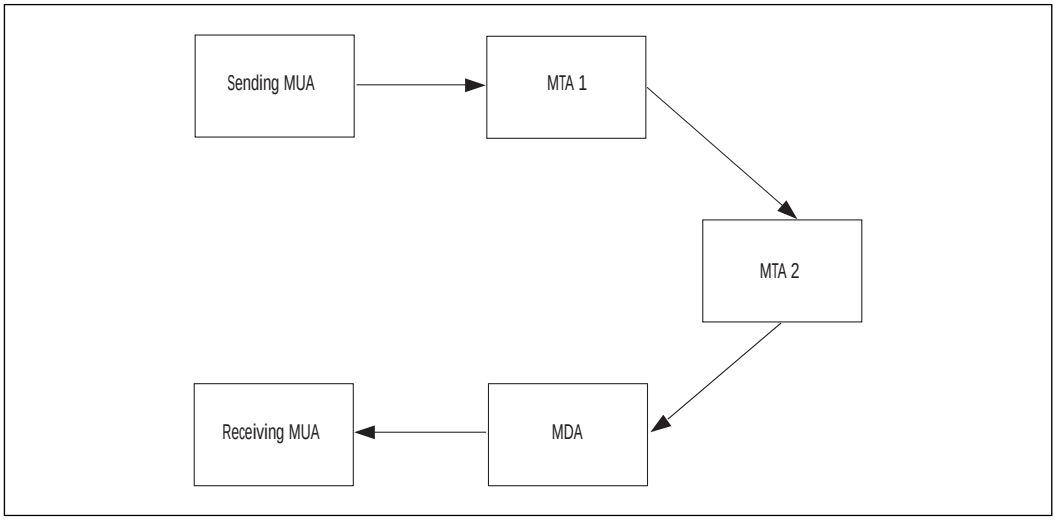
Understanding a Mail User Agent and a Mail Transfer Agent

When you create an e-mail message, the client software you use is called a *Mail User Agent (MUA)*. When you send your message, you send it to a server called a *Mail Transfer Agent (MTA)*. As you might suspect, an MTA is responsible for transferring your message to a single server or collection of additional MTA servers, where it is finally delivered. The server that holds the message so that it can be read is called a *Mail Delivery Agent (MDA)*. You should note that an MDA and an MTA can reside on the same server, or on separate servers. Your friend can then use his or her MUA to communicate with the MDA to download your message. Figure 1.1 shows how a sending MUA communicates with an MTA (MTA 1), which then communicates with another MTA. The message is then delivered to an MDA, where the receiving MUA downloads the message.

Each of these agents must cooperate in order for your message to get through. One of the ways that they cooperate is that they use different protocols. In regards to the Internet, the MTA uses a protocol called the Simple Mail Transfer Protocol (SMTP), which does nothing more than

deliver messages from one server to another. When you click the Send button, your client software (i.e., your MUA) communicates directly with an SMTP server.

Figure 1.1 Tracing an e-mail message.



NOTE

All systems that are connected to a network (such as the Internet) must have open ports, which are openings to your system that allow information to pass in and out of your system. Many times these ports must remain open. However, there are times when you should close them. You will learn how to close ports in Chapter 8.

An MTA using SMTP on the Internet uses TCP port 25. Once an MTA receives a message, its sole purpose is to deliver it to the e-mail address you have specified. If the MTA is lucky, it only needs to find a user defined locally (i.e., on itself). If the user is in fact defined locally, then the MTA simply places the e-mail in the inbox designated for the recipient. If the user is not defined locally, then the MTA has more work to do. It will contact other servers in its search for the proper destination server. This search involves using the Domain Name System to find the correct domain name. If, for example, your friend's e-mail address is james@syngress.com, then the MTA will find the syngress.com domain name, then search for the e-mail server that is designated for this DNS domain.

NOTE

An MTA finds the correct domain name by consulting a special DNS entry called a mail exchanger (MX) record. This record defines the authoritative e-mail server for this domain. Using an MX record allows an e-mail message to be addressed to james@syngress.com, instead of james@mailserver.syngress.com. This is because an MX record ensures that any message sent to the syngress.com domain automatically gets sent to the machine named mailserver.syngress.com. This feature of DNS greatly simplifies e-mail addresses, and is in use everywhere.

The Mail Delivery Agent

Once an MTA delivers the e-mail you have sent to your friend, it resides in a *drop directory*. The recipient, James, then has at least two options:

1. He can log on to the server and access the message. Whether he logs on locally or remotely, he can use an MUA to read the message.
2. He can use his own e-mail client and log on remotely using either the POP3 or IMAP protocol.

The Post Office Protocol 3 is the third version of a protocol that allows you to quickly log into a central server, download messages, and read them. This protocol listens for authentication requests on TCP port 110. With this protocol, you must first authenticate using a user name and a password, and then download the messages. After the recipient downloads the message you sent, his MUA will tell the server to delete it, unless he configures it to leave messages on the server.

The Internet Message Access Protocol (IMAP) is a more sophisticated protocol. Like POP3, it requires a user to authenticate with a user name and password. Unlike POP3, an IMAP server does not require that you first download your e-mail messages before you read them. After logging in, the recipient can simply read the messages, rearrange them onto directories that exist on the MDA server's hard drive, or delete them. He will never have to download the messages to his own hard drive if he doesn't want to. An IMAP server usually listens on TCP port 143.

When Are Security Problems Introduced?

Because this is a book on security, you may be wondering when, during this process, security problems are introduced. The answer is that they are usually introduced by the MUA. There are several reasons for this:

- MUA software, such as Netscape Messenger, is designed for convenience rather than security.
- The software is often upgraded, quickly produced, and is not meant to conceal information.
- The applications are often used by naïve end-users who use default settings.
- When the MUA logs in to the MDA POP3 or IMAP server, authentication information is often sent in clear text format. In other words, the password information is not encrypted, and can be sniffed off the Internet by malicious users.
- Users will often double-click an e-mail attachment without knowing its origin. If this attachment contains malicious code, a chain reaction will occur, which usually involves having the MUA send unsolicited messages to other MUAs. The result is an ever-increasing stream of traffic that can bog down the sending servers (the MTAs), as well as the MDA.

It is possible for problems to be introduced at the MTA level, as well as at the MDA level. To learn more about these problems, let's take a look at some of the older attacks and the specific weaknesses of the servers we use every day.

History of E-mail Attacks

It may be tempting to think that attacks on e-mail clients and servers are recent events. The Melissa, BubbleBoy, and Life Stages attacks were all waged in the last year, for example. Each of these attacks is essentially the same. They take advantage of the sophisticated relationship between an e-mail client and the rest of the operating system. By simply double-clicking on an attachment, an unwitting user can infect their own system, then begin a process where additional users are sent malicious files. The process continues from there. It would certainly seem that such attacks are closely associated with the world's embrace of the Internet. However, e-mail servers have been the target of some of the oldest attacks on record.

The MTA and the Robert Morris Internet Worm

In 1988, a graduate student named Robert Morris created a software program that took advantage of a popular MTA server named Sendmail. Sendmail is arguably the most popular MTA on UNIX and Linux servers (it is covered in detail in Chapter 10). Back in 1989, it was the only MTA capable of routing e-mail messages across the Internet. The particular version of Sendmail popular in 1989 was subject to a bug where it would run on the system and forward any request given to it. Morris created code that took advantage of the open nature of Sendmail. The code was designed to first attack a little-documented Sendmail debugging feature that allowed the server to execute commands directly on the system.

Morris' program was specifically designed to:

- Run itself automatically on the local system.
- Use the local system to query for additional target systems that also had the Sendmail debugging feature. For example, it would use applications such as traceroute and netstat to discover other machines on the network.
- Cause a daemon called *finger* to crash. The finger daemon is designed to inform a person about the users currently logged on to a system. Morris's worm caused this daemon to crash by sending it too much information. As a result, the finger daemon's memory space, called a *buffer*, overflowed itself and overwrote memory that was actually allocated to another system. This problem is called a *buffer overflow*. As a result, the worm was able to crash the daemon and then use memory left behind to execute itself.
- Change its name before moving to another system.
- Propagate itself automatically to other systems. Often, this was accomplished by exploiting system trusts, which allow trusted systems to log on without first authenticating.
- Log on to other servers, then execute itself to spread to another system.
- Execute itself repeatedly on the system, thereby drawing on system resources until the system crashed.

Thus, the code could move from server to server without human intervention. The code also worked quickly, running multiple copies of itself on one system. The result was a series of system crashes that invaded between four to six thousand servers in less than 24 hours. Almost two thirds of the known Internet was brought down in one night.

MDA Attacks

In Chapter 2, you will learn how Web-based e-mail servers such as HotMail have fallen prey to attacks. Most of these attacks involve code that is designed to thwart authentication. Sometimes, the attacks focus on code meant to dupe unsuspecting users into thinking that they are logging in, when in fact they are actually sending their passwords to a malicious user. Other attacks are more global. These involve scripts that completely defeat the authentication process and allow a hacker to log in to any account without a password.

Once a hacker has logged in, he or she can:

1. Assume the identity of a valid user and send bogus e-mail messages to unsuspecting users.
2. Obtain the passwords of the rightful user. This practice may not seem to be very fruitful, but consider this: Many people use the same password for multiple purposes; a person's e-mail password may also be his or her bank card PIN, home security password, or network login password.
3. Manipulate e-mail messages that are waiting to be read. In addition to simply deleting such messages, a malicious user can actually alter incoming messages so that they contain bogus information.

Analyzing Famous Attacks

The following is a brief discussion of additional attacks. As you read about them, notice that although they no longer involve Sendmail and the finger daemon, they still take advantage of internal and external system trusts:

Melissa Perhaps the most famous e-mail attack, the Melissa virus was released in February of 1999. Melissa was the first popularly known e-mail virus that spread from user to user via e-mail. The chief reason for its success was that it was able to take advantage of Microsoft Outlook's address book. It read the address book and sent infected e-mail to the first 50 people listed on the address book. Because the infected e-mails appeared to originate from friends, many people double-clicked the attachment, which allowed the virus to spread at a rapid rate. Now that it has been out for some time, different versions of Melissa have appeared. These mutations have essentially the same effect, although they have slightly different names. Melissa's creator attacked Microsoft technology, so the virus was not able to use the MUAs residing on Macintosh, UNIX, or Linux systems. Melissa succeeded in crashing the e-mail servers for several major sites, including military installations and Internet service providers (ISPs) such as America Online.

BubbleBoy Like Melissa, this attack targets Microsoft-specific MUAs, specifically Microsoft Outlook and Outlook Express. When activated, it will send itself to all names in your personal address book. All messages sent from infected machines have the following line in the Subject field: "BubbleBoy is back!" One of the chief differences between this virus and others is that it does not require direct user intervention to spread. Whereas Melissa required a naïve user to double-click on an attachment, BubbleBoy activates when the Preview Pane option is activated in Microsoft Outlook or Outlook Express. The virus is specific to Microsoft Windows 98 and 2000 that have Internet Explorer 5 installed on them. Furthermore, the Window Scripting Host option must be enabled in Internet Explorer (a default selection). This requirement may seem to be a limitation, but considering the ubiquitous nature of Windows, you can quickly get an idea of how quickly this virus can spread. Mutations of BubbleBoy have appeared since it was originally introduced to the Internet in November of 1999. Some of these mutations can have destructive effects.

Love Letter This worm was released from a computer in the Philippines. It targets MUAs that are designed to run Visual Basic scripts (again, Microsoft Outlook and Outlook Express). The attachment, which reads "LOVE-LETTER-FOR-YOU.TXT.vbs," contains malicious script that has your MUA (usually Microsoft Outlook or Outlook Express) automatically send copies of itself to all of the contacts it finds in your address book. Not only does this particular worm alter various files (such as .jpg, .mp3, .wav, .doc, .gif, and .htm), but it also attempts to download a binary called WIN-BUGSFIX.EXE, which attempts to collect password information from the host. This worm also spreads via Internet Relay Chat (IRC) programs. The indirect result of this virus was that many corporate MTAs and MDAs crashed because they couldn't handle all the traffic.

Life Stages Introduced in June of 2000, this worm spreads primarily through e-mail, although it can also spread through IRC and ICQ ("I Seek You," a chat program provided by Mirabilis, at www.mirabilis.com). This virus is characterized by an e-mail message apparently sent by a friend that contains a message such as "Life Stages," "Jokes," or "Funny." One of the unique elements of this worm is that it is able to change itself to avoid detection. When a worm or virus can alter itself, it is said to be *polymorphic*. Although this worm requires some user intervention, it is not as sneaky as BubbleBoy; a user must double-click on an attachment before it spreads to all users listed in your address book.

Case Study

In June of 2000, a medium-sized company (just over 200 employees) was attacked by a variant of the Love Letter virus. The attack was immediately noticed around 8:10 a.m., when the majority of people in the company had logged in and checked their e-mail. Most of the users who fell prey to the attack were new to the company and had not yet been trained how to open attachments safely. In fact, several of the users double-clicked on the attachments several times, because nothing visible occurred. The end-users expected an image or a movie, and so they just kept clicking on the mouse. The result of this attack was that the e-mail server had to be restarted, and about fifteen employees had to update their anti-virus definitions. Furthermore, the systems administrator promptly circulated an e-mail reminding users about being careful about opening e-mail attachments and updating their antivirus software.

Learning from Past Attacks

Clearly, there is much to learn from all of these attacks. One of the first lessons is that the Internet is still very much prone to a similar attack. The Life Stages, BubbleBoy, and Melissa programs demonstrate how vulnerable e-mail clients and servers are to illicit code. Without third-party software and custom configuration, your software is extremely vulnerable. Second, the Morris worm was able to spread because many systems blindly trusted each other to do the right thing. If your system trusts others blindly, then you are vulnerable. Most servers that allow clients to log in and pass on e-mails without first conducting a scan are far too trusting.

Third, the internal software components of each server also blindly trusted each other. One illicit application sent from one server to the next was able to cause a massive amount of damage. This fundamental pattern has not changed. Likewise, most server components still blindly trust each other, which means that one compromised element of the operating system can then cause a malicious application to spread throughout the system and crash it. When it comes to e-mail servers, the domino theory applies today: If one server or client falls to a virus, chances are that many others will, as well.

Fourth, applications that make things simple can cause problems. Any e-mail application that automatically opens attachments, provides preview panes, and allows information to pass unchecked back and forth between applications is helping to contribute to security breaches and attacks.

Fifth, these attacks all suggest that unchecked system bugs can help cause problems. Although it is impossible to eliminate all system bugs from all of your software, you should make every effort to keep your sys-

tems current. Such proactive steps will save you countless headaches in the future.

Finally, poor programming practice and application design helps contribute to e-mail attacks. When checking your software, remember that to one person, a particular feature of an application or server may appear as a bug or security flaw. Always consider the ramifications of various features of the software that you use.

Viruses

Now that you have a good understanding of the behavior of e-mail server attacks, it is necessary to further define some of the terms used in this chapter. A *virus* is any binary file that meets the following criteria:

1. It requires direct human intervention in order to spread. Unlike a worm, which spreads automatically, a virus requires a user to download and double-click a binary file, or transfer it using an infected medium, such as a floppy disk.
2. It has a payload, which can be destructive behavior (deleting or altering files), or annoying messages left on the screen, or both.
3. A virus spreads quickly to all documents in an operating system. A virus never spreads itself to other systems automatically.

Although many others exist, *macro viruses* are by far the most common. Word processors and spreadsheets, such as Microsoft Word and Excel, allow users to create powerful, convenient mini-applications that reside within the word processor. These macros are meant to simplify life by cutting down on repetitive tasks.

The problem with macros is that many end-users allow macros to run without first establishing controls over what they can do. The macro facilities in office suites, such as MS Office, are almost always powerful enough to launch applications, delete files, and begin a sequence of events that can seriously damage the system. A malicious user can take advantage of powerful macro facilities. In fact, the Melissa virus is a macro virus. Many others exist that are not as ambitious, but which are still powerful.

Worms

The chief difference between a worm and a virus is that a *worm* spreads to other systems. Furthermore, a worm is able to spread with little or no user intervention. Remember, in order for a virus to spread, a user must first install it by copying a file or inserting a floppy disk. A worm can spread

itself upon activation. By simply double-clicking a file, the worm can be activated, and deliver its payload (if any), then spread by taking advantage of system settings, macros, and applications (called application programming interfaces, or APIs) that reside on a system.

Whereas a virus is generally designed to spread throughout an entire machine, a worm is designed to propagate itself to all systems on a network. There are four factors that allow a worm to spread rapidly:

1. Networks that use one operating system. For example, an exclusively Microsoft or Novell network stands a greater risk of rapid infection than a heterogeneous network that uses UNIX, Novell, and Microsoft servers.
2. Networks that standardize to one MUA, such as Microsoft Outlook. Just as networks that have one operating system are vulnerable, a company that uses one MUA is liable to experience an event where a virus is propagated quickly. Also, because Outlook is so popular, hackers are more familiar with it. Therefore, a hacker can create an application that exploits it.
3. Operating systems, such as those vended by Microsoft, that provide interpreters and models, such as the Component Object Model (COM), which make it easy to create powerful applications in just a few steps.
4. Networks that use TCP/IP. Although TCP/IP is a powerful, efficient protocol, it was not designed with security in mind. Although the next version of IP, called IPv6, improves security, this version of IP has not been implemented widely. The current version of IP, called IPv4 allows a malicious user to imitate (i.e., spoof) the origin of an IP address. As a result, it can be very difficult to find the true attacker in case of an incident.

Types of Worms

Below is a brief discussion of the three major types of worms:

1. **True worms** Requires no human intervention to spread. This type of worm is rare, because it requires great skill on the part of the programmer, and will function only on a homogeneous network. A true worm is also rare because it uses the programming language of the e-mail server itself. For example, to create a worm for the Netscape Enterprise e-mail server, you would have to write the application using the language that Netscape Enterprise Server uses.

2. **Protocol worms** Any worm that uses a transport protocol, such as TCP/IP, to spread. The Robert Morris worm, for example, used elements of TCP/IP, including finger and Sendmail (which uses SMTP), to spread itself. This type of worm can also spread without any direct human intervention.
3. **Hybrid worms** A worm that requires a low level of user intervention to spread, but also acts like a virus. A simple click on a malicious attachment does not mean that this user is ready to copy or transmit an application. However, a click still represents user intervention. Most of the worms discussed in this chapter, such as BubbleBoy, Melissa, and Life Stages are hybrid worms, because they behave like viruses in that they deliver a payload. However, they also exhibit worm-like behavior, because they are able to spread automatically from system to system.

Trojans

A Trojan horse, or *Trojan*, is nothing more than an application that purports to do one thing, but in fact does another. Trojans are named after the mythic Trojan horse in Homer's *Iliad*. In the legend, the Greeks created a wooden horse, then gave it to the citizens of Troy as a peace offering. However, before the horse was presented, Greek soldiers hid inside it. The horse was brought inside the city gates, and when the city was asleep, the Greek soldiers emerged and were able to conquer Troy. Similarly, a Trojan looks like a benign or useful program, but contains a payload. For example, a Trojan can:

- Launch an application that defeats standard authentication procedures.
- Delete files.
- Format the hard drive.
- Launch legitimate applications with the intent of defeating security.

Many Trojans have a payload. A common payload is to delete a file, many files, or even an entire partition. Perhaps the most common payload is an illicit server.

Illicit Servers

An illicit server is nothing more than a simple service or daemon that defeats a server's authentication mechanisms. A *valid* server, such as an

e-mail or Web server, always has authentication mechanisms that allow only certain users. Illicit servers have the following characteristics:

1. They open up an ephemeral TCP or UDP port (over 1024).
2. They attempt to hide any trace of their existence. They do not show up in a task bar or in a task list.
3. Most of the time, an illicit server is a very small binary that is easy to conceal as a hidden file, or it is one small file in the midst of several others.

Using such a server, a malicious user can compromise your e-mail server. Examples of illicit servers include:

NetBus and NetBus Professional Although many professionals consider NetBus Professional to be perfectly legitimate, each of these applications can be used to gain unauthorized control of a system. NetBus has a client and a server. Usually, a hacker will engage in social engineering or other means in order to get the server installed on the victim's system.

Back Orifice and Back Orifice 2000 More ambitious than NetBus, these illicit servers allow you to open FTP and HTTP connections on any port you specify. Using these servers, a malicious user can read the entire hard drive of any Windows system, as well as upload, download, and delete files. Back Orifice 2000 even allows a malicious user to specify a password, encrypt transmissions, and even destroy the server to avoid detection. Like NetBus, Back Orifice uses a client and a server. Figure 1.2 shows the client.

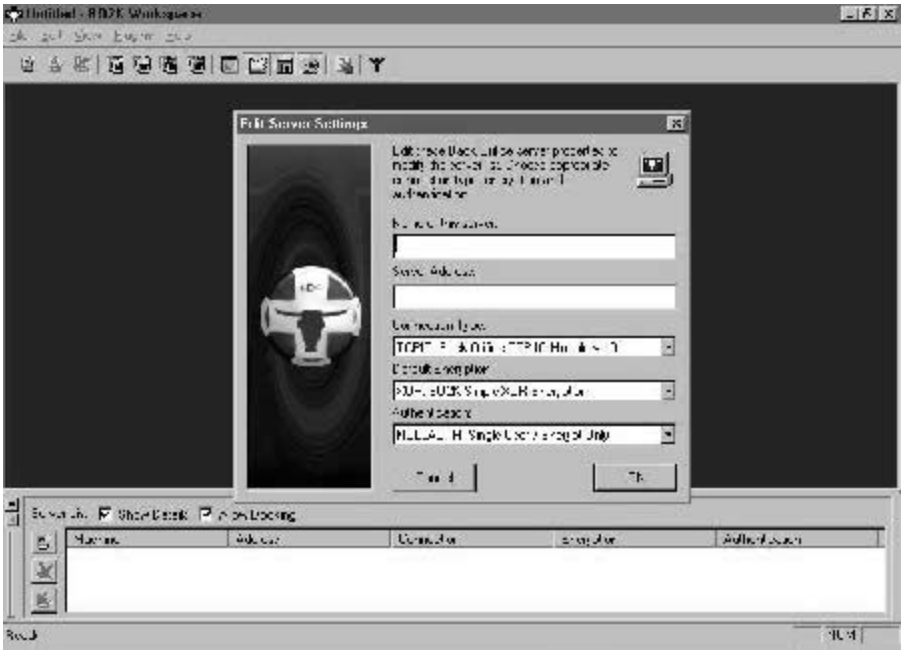
Netcat Although a legitimate tool, it is possible for a malicious user to use this application to create an illicit server.

Many other illicit servers exist, most of which you will never hear about; after all, why would a hacker give up trade secrets? Usually, a hacker will trojanize these servers in an attempt to trick end-users into installing them. Such social engineering practices are common. One of the more infamous examples of social engineering is where a hacker took a version of the Whack-A-Mole game and linked it to NetBus. Then, the hacker began sending this game to various people, who then played it and unwittingly installed the NetBus server on their systems.

Differentiating between Trojans and Illicit Servers

Do not use the terms Trojan and illicit server interchangeably. An illicit server is often presented to users in trojanized form, but an illicit server is not necessarily a Trojan. For example, unless you disguise NetBus as another application, it is simply an illicit server.

Figure 1.2 The Back Orifice client.



E-mail Bombing

Another form of attack involves sending hundreds, if not thousands, of large e-mail messages to an account on a server. Due to the large volume of e-mail messages (not to mention their size), the victim account will remain unusable until the systems administrator removes all of the messages, or creates another account.

Many easy-to-use applications exist that are meant to enable the most untalented user to send an e-mail bomb. You will learn how to thwart such attacks in subsequent chapters.

Sniffing Attacks

TCP/IP is an inherently insecure protocol, because it does not encrypt transmissions by default. Therefore, it is possible for a malicious user to use a protocol analyzer (also called a *packet sniffer*) to capture and then view packets. Applications such as Sniffer Basic and TCPdump are specially designed to place a Network Interface Card (NIC) into *promiscuous mode*. Once in promiscuous mode, a NIC can then capture any packets that are passing through your particular portion of the network.

Most network sniffers are able to capture all information sent across the network. Information can include such things as user names, passwords, and the contents of an e-mail message.

NOTE

In order for a malicious user to capture e-mail traffic, he or she must be between the two servers that are communicating. Any ISP, for example, is in an ideal position to sniff traffic. However, due to the nature of most networks, any traffic passing from one computer to another can be sniffed. If the president of the company logs on to his e-mail server using a standard POP3 or IMAP account, this password—a well as any e-mail message—is sent in the clear. As a result, any user with a sniffer can capture the password and read the company president's e-mail messages.

Carnivore

One of the more notorious examples of e-mail sniffing is the Carnivore application. Developed by the United States Federal Bureau of Investigation (FBI), this application is designed to capture and process large amounts of e-mail. All an agent has to do is place a machine with Carnivore enabled on the hub or a router of an ISP, and then read all e-mail messages sent to it.

NOTE

A *router* is a specialized machine responsible for ensuring that different IP networks can communicate with each other. A *hub* is a simple device that allows machines on the same network to communicate with each other.

Using Carnivore, the FBI can read a user's incoming and outgoing mail, learn about the people the user is communicating with, and gain access to passwords and other information. The FBI is supposed to obtain a search warrant that identifies only specific users. Needless to say, this application is quite controversial, and has raised questions concerning privacy.

Recently, a company named NetworkICE has created its own version of Carnivore. Called Altivore, this application does much the same thing as

Carnivore, but is freely available at the www.networkice.com Web site. Now, anyone has the ability to capture and read e-mail transmissions. What's more, Altivore can run on almost any standard PC, whereas Carnivore requires a dedicated system. Considering that this software is readily available to any user, it is very possible that your private e-mail is not so private after all.

Spamming and Security

Many older MTA servers allow any user or system to connect to them and send e-mail anonymously. Whenever an e-mail server allows a user to send e-mail anonymously, it is said to allow *relaying*. Servers that allow relaying allow users to specify any user name and any DNS domain in an e-mail message. For example, should you find an e-mail server that allows relaying, you could, with just a few commands, create a fairly convincing e-mail message from bill.gates@microsoft.com, william.shakespeare@bard.com, or keisersoze@usualsuspects.com.

While this practice may seem amusing, bulk e-mail applications can send thousands, if not millions, of junk e-mail messages called *spam*. Although most MTA servers that currently ship do not have relaying turned on, you should check your system. Not only is spam e-mail annoying, it wastes time, valuable network bandwidth, and slows down the Internet.

The Mail Abuse Prevention System (MAPS) is one of several organizations that have organized to prevent spamming. You can read more about MAPS at their Web site (www.mail-abuse.org). Their chief goal is to conduct scans of e-mail servers across the Internet and then inform systems administrators that their servers currently allow e-mails to be sent anonymously.

MAPS then informs the offending systems administrator. If no action is taken, then MAPS will blacklist your e-mail server so that it cannot communicate with the rest of the Internet. Additional anti-spam organizations include:

- The Coalition Against Unsolicited Commercial E-mail (www.cauce.org)
- The Forum for Responsible and Ethical E-mail (www.spamfree.org)

Common Authoring Languages

Table 1.1 provides an overview of the languages often used when authoring applications designed to exploit e-mail servers. None of these languages is better or worse than the other. Some are best suited for certain practices.

Table 1.1 Languages Used To Create Malicious Code

Language	Description
C	The most popular language among hackers. Linux, for example, ships with a free compiler that allows programmers to create and compile code easily. C is an older language, but remains popular because it is efficient in regards to networking.
C++	A newer language, C++ is also more complex to learn. However, more and more applications are being written in this language as the knowledge base grows.
Java	Java applets and applications are increasingly becoming popular in e-mail-based exploits.
Visual Basic	Visual Basic is a Microsoft-specific language. It is especially popular among those who wish to exploit Windows systems running Microsoft Outlook and Outlook Express.
JavaScript	JavaScript is an interpreted language, which means that it does not need to be compiled. This script is usually inserted into HTML pages. It can also be used on servers. However, JavaScript embedded into HTML pages is by far the most popular way to create an exploit. JavaScript is best suited to creating fake, pop-up authentication windows and applications meant to dupe unwitting users to reveal their passwords. JavaScript only remotely resembles Java. Do not confuse the two, as JavaScript is not anywhere near as complex or capable.
VBScript	VBScript is, like JavaScript, an interpreted language that is ideal for inserting into HTML pages. When placed within an HTML page, VBScript runs only on the Microsoft Internet Explorer browser, and in Microsoft Office. It is possible to obtain plug-ins for Netscape so that it, too, can run VBScript. However, it is also possible to use VBScript on the server side; Microsoft's Active Server Pages use VBScript as its primary language.
Perl	Perl is also an interpreted language, but much more versatile and powerful than JavaScript or VBScript. You can learn more about Perl by visiting www.perl.com.

NOTE

Macro viruses are almost always written in Visual Basic. Other popular languages for creating viruses and attacks include C, C++, and various scripting languages including Perl, JavaScript, and VBScript. All of these languages provide many options to the creator of a malicious application.

Protecting Your E-mail

So far, this chapter has focused on defining terms, discussing how e-mail works, and how hackers have been able to attack e-mail clients and servers in the past. How exactly do you *protect* your e-mail? Next we describe the most popular choices, all of which will be expanded upon in future chapters.

Protecting E-mail Clients

You can protect e-mail clients by:

- Purchasing an anti-virus package
- Obtaining a personal firewall
- Encrypting your transmissions

Third-party Applications

Anti-virus applications such as Norton AntiVirus and McAfee VirusScan can scan your system for viruses. Almost any product that you buy offers the option of scanning e-mail message attachments before you open them. This service is quite valuable. However, this service can have two drawbacks:

1. Scanning attachments can take time and processor speed. As a result, you may find your computer's performance to be unacceptably slow.
2. If you do not update your anti-virus application regularly, the scan may not find a newer virus. It is easy to be drawn into a false sense of security when you assume your attachment scanning software is current.

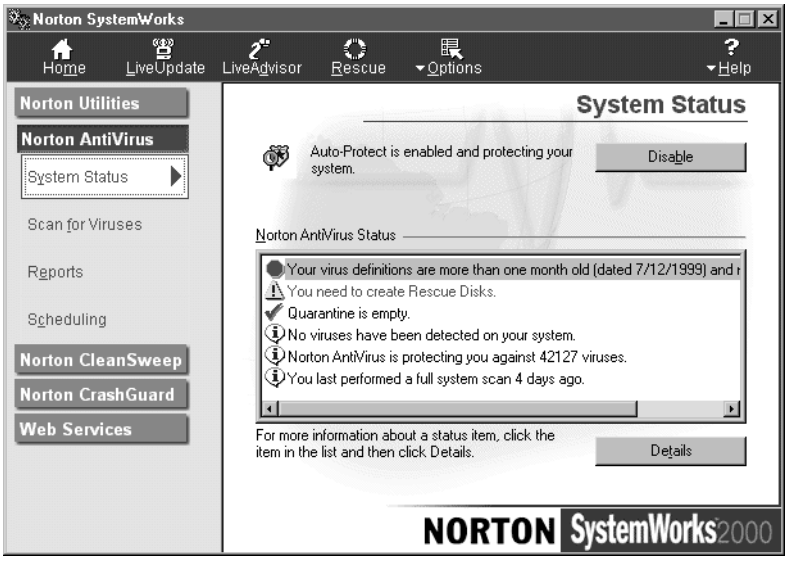
Figure 1.3 shows Norton SystemWorks, a typical application that contains an antivirus component.

Personal firewall software often includes an anti-virus scanner. However, a personal firewall takes the extra step of protecting your computer by closing down unnecessary ports. Personal firewall software can also:

- Tell you the IP address and/or resolved IP address of the hacker attacking your system.
- Filter out TCP/IP-related packets. For example, personal firewall software can block packets sent by the ping application.
- Disable a system from sending and/or receiving e-mail.

A personal firewall can provide additional services, depending upon the personal firewall vendor you select.

Figure 1.3 Norton SystemWorks.



Encryption

The chief way to protect an e-mail message on the client side is to use *encryption*. Using encryption makes it difficult for unauthorized users to read or tamper with your e-mail. There are three types of encryption used to secure information on the Internet:

1. **Private key encryption** The use of one password to encrypt and decrypt information.

2. **Public key encryption** The use of a key pair to encrypt and decrypt information.
3. **Hash encryption** A process that creates a numerically related hash of the information. This code is theoretically irreversible, and is used to help ensure a document has not been tampered with.

One of the most common ways to encrypt a document is to use a single string of text to encrypt it. If you have ever used Microsoft Word, for example, to encrypt a document, you have used private key encryption. This form of encryption is called *private key* because you must take measures to ensure that your password remains secret. If an unauthorized user were to learn the password to this document, then he or she would be able to open it.

Let's say that you have encrypted a Microsoft Word document that you wish to give to a friend. Suppose that for some reason you cannot simply call your friend and share the password. You could send an e-mail with the password, but doing this carries the risk that someone might sniff your e-mail message and get the password. So, how do you transmit this document and password to your friend? You could place the password in another document and encrypt this document, but now how do you transport this new password? It seems that this process has a logical flaw. In order to transmit the document securely, you must first transmit the password in an insecure manner.

The answer, at least as far as e-mail is concerned, is to use *public key* encryption. Applications such as Microsoft Outlook, Netscape Messenger, and Eudora Pro support public key encryption. Public key encryption involves the creation of a *key pair*. This pair is mathematically related. The first key, called a *private key*, must remain private at all costs. It will be placed in a hidden location on your hard drive. It is useful to think of a key pair as a whole that you then divide into halves. The pair always works together, even though the public key can be distributed freely.

You can safely give the *public key* to the most experienced hacker in the world. This is because even though these keys are related, it is very difficult (if not impossible) to use one key to defeat the other. However, a fundamental principle makes it possible for you to send a message to your friend. A user's private key can decrypt information encrypted to the user's public key. In other words, if Sandi were to encrypt a message to James' public key, then only James' private key can decrypt that message.

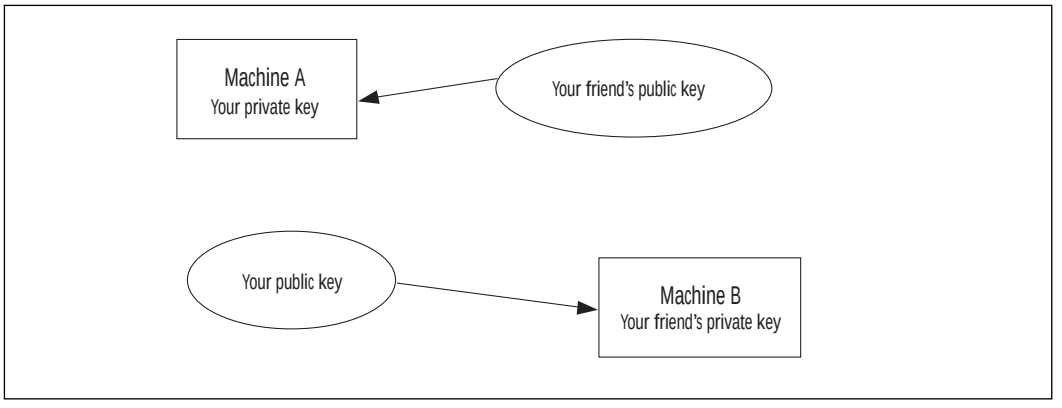
Let's spend some time on this concept. When you wish to send your friend an e-mail message, you each must create a key pair. You will keep your private key in a hidden place, and will never reveal it, or the password used to access it, to anyone. You never need to. The same principle applies to your friend. He or she will never reveal their secret key, or the password

that allows them to access their private key. However, both of you must give your public keys to each other. You have theirs, and they have yours.

Then, all you have to do is encrypt your e-mail message to your friend's public key. Now, not even you can read this message. Why? Because the only key in the world that can decipher this message is your friend's private key. Similarly, when they want to send you an encrypted e-mail message, they must encrypt that e-mail message with your public key. Then, when you receive the message, you can decrypt it with your private key. Figure 1.4 is meant to explain how you must first exchange public keys with a recipient before the messages are encrypted.

Whenever you exchange public keys, you are said to be establishing a *trust relationship* between you and your friend.

Figure 1.4 An established trust relationship between machine A and machine B.



NOTE

Dedicated servers exist that contain the public keys of many individuals. You can place your public key on these servers for others to download, or you can e-mail the keys to the person with whom you wish to establish a relationship. A quick solution might be to create space on an FTP or Web server that contains the public keys of those who wish to communicate securely.

Applications such as Pretty Good Privacy (PGP) use this technique. Commercial servers, such as Microsoft Exchange, also provide the ability to encrypt transmissions on the server side. You will learn more about implementing public key encryption in Chapters 2 and 3.

NOTE

Public key encryption has one drawback: It is extremely slow. As a result, most commercial applications use private key encryption to encrypt an e-mail message. They then use public key encryption to encrypt only the symmetric (private) password.

Hash Encryption and Document Signing

The third form of encryption in use today is *hash encryption*. Another name for this type of encryption is *one way encryption*, because once information is encrypted through this process, it is irretrievable. This process is used because it can help determine if a message has been tampered with. Public and private key encryption provide only one service: data encryption. When you need to transmit information across the Internet, it would also be nice if you could ensure that this information was not tampered with during transit.

One way to do this is to *electronically sign* a message by creating a hash of the message. *Hash codes* are created through a process that closely reads the contents of a message. Contents include the size of the message, the characters within it, and how they are arranged. Any single change in the document results in a different hash value. Therefore, if you were to create a hash of your e-mail, and someone were to tamper with the message, you could tell, because the hash value will change when you verify it.

Applications such as PGP use one way encryption to first create a hash of the document. Whenever you use an MUA such as Netscape Messenger to sign a document, you are using creating a hash of your e-mail message. You will learn more about implementing these concepts in Chapter 3.

Protecting the Server

Now that you know how to protect information emanating from an MUA, it is important to learn some of the ways to protect the MTA and MDA. These methods include:

- **Hardening the e-mail server's operating system** Hardening the operating system involves locking down unnecessary ports; upgrading your system using the latest, stable service patches and bug fixes; and changing default settings.

- **Placing your system behind a firewall** When implementing an e-mail server, you should place it behind a firewall. A firewall is a more powerful, robust version of a personal firewall. It resides on a separate system, then scans and filters out packets. By placing your Web server behind a firewall, you are essentially protecting all aspects of your system except those ports that are exposed to the Internet. For example, if you are using ports 25 and 110, then users will be able to connect to only these ports. A firewall, therefore, reduces the number of attacks that can be waged against your system.
- **Configuring the server to allow connections from certain hosts only** Most e-mail servers (or their underlying servers) allow you to control which systems can connect. Taking time to lock down your server can greatly increase security.
- **E-mail scanning** Scanning the body of an e-mail message protects e-mail users, as well as the MTA and the MDA. Once you have placed your e-mail server behind a firewall, you should then take steps to filter traffic that is passing through your e-mail ports.
- **Attachment scanning** Scanning attachments on the server side can consume an enormous amount of system resources, but it is often helpful. For example, once you learn about a particular virus attachment, you can program your attachment scanning software to block out only this attachment. Of course, for those administrators who are truly security conscious, the option to disallow all e-mail attachments is always available.

Summary

This chapter is an overview of the concepts that will be discussed throughout the book. You should now have an understanding of authentication, access control, and how e-mail servers and clients work together to send a message. From studying some of the past attacks, we can predict some of the common patterns attackers follow. We know, for instance, about some of the common attacks waged against MUAs, MTAs, and MDAs. From the Robert Morris worm to Melissa and Life Stages, we are now aware of the threats and issues that confront systems administrators.

We have introduced the most popular methods for securing e-mail servers. From encrypting transmissions to installing third-party scanning software, many options are available to you. The following chapters are designed to provide you with real-world solutions.

FAQs

Q: Why would a hacker want to conduct a denial of service attack?

A: The first reason is that it is easier to conduct a denial of service attack than it is to formulate an attack that allows a user to authenticate. Therefore, you tend to see a lot of script kiddies who gain a quick, cheap sense of satisfaction watching an e-mail server crash. However, more sophisticated reasons exist to conduct a denial of service attack. Should a malicious user want to hijack a connection between your e-mail server and a client logging in, they would want to conduct a denial of service attack against the client in order to take over the connection and log in. So, although many denial of service attacks are conducted just to watch the server die, there are times when a DoS attack is a step in a more sophisticated process.

Q: What attacks are e-mail servers most prone to?

A: The answer has to do more with how well you have protected the e-mail server. Recently, worm-based attacks, such as Melissa, have been the most devastating. However, e-mail servers that scan e-mail bodies and e-mail attachments can greatly reduce attacks. Furthermore, if the server is placed behind a firewall, it will be much safer.

Q: If worms attack the e-mail client, then why do the e-mail servers (the MTA and the MDA) get overwhelmed as well?

A: Because the MTA must process hundreds of thousands of requests in a very short period of time. Also, the MDA can become bogged down because it has to deliver all of these messages to users. This is especially true if the MDA is housed on the same server.

Q: Is it possible for an MTA to encrypt messages?

A: Yes. One of the drawbacks of encryption on the part of the MTA is that encryption can slow down the delivery process. Also, MTA-based encryption is usually proprietary; only those systems within a company organization can encrypt their e-mail messages; if they have to send messages outside the company, or to other MTAs, the message will no longer be encrypted.

Q: Where can I learn more about viruses, worms, Trojans, and illicit servers?

A: One of the many sites that explains cryptography is the United States National Institute of Technology (NIST), at http://csrc.nist.gov/nistpubs/800-7/node207.html. You can also search the www.cryptography.com site. As of this writing, the following link contains a valuable list of resources: www.cryptography.com/resources/index.html.

Q: This chapter has discussed the possibility of encrypting e-mail messages. Is it possible for someone to find an application that can decrypt messages without your authorization?

A: Yes. There really is no such thing as an infallible encryption process. If a government or large corporation wished to devote enough resources, such as multi-million dollar supercomputers, it is possible that they could decrypt your e-mail message. Readily available products can still encrypt transmissions so that even the most sophisticated computers would take days, if not weeks or months, to decrypt messages.

Q: In public key encryption, what happens if someone obtains my private key?

A: You will have to generate a new key pair. If your private key gets published, then anyone can plug this private key in to the appropriate application, such as PGP, and read your messages.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Y

Z

The Global Knowledge Advantage

Global Knowledge has a global delivery system for its products and services. The company has 28 subsidiaries, and offers its programs through a total of 60+ locations. No other vendor can provide consistent services across a geographic area this large. Global Knowledge is the largest independent information technology education provider, offering programs on a variety of platforms. This enables our multi-platform and multi-national customers to obtain all of their programs from a single vendor. The company has developed the unique CompetusTM Framework software tool and methodology which can quickly reconfigure courseware to the proficiency level of a student on an interactive basis. Combined with self-paced and on-line programs, this technology can reduce the time required for training by prescribing content in only the deficient skills areas. The company has fully automated every aspect of the education process, from registration and follow-up, to "just-in-time" production of courseware. Global Knowledge through its Enterprise Services Consultancy, can customize programs and products to suit the needs of an individual customer.

Global Knowledge Classroom Education Programs

The backbone of our delivery options is classroom-based education. Our modern, well-equipped facilities staffed with the finest instructors offer programs in a wide variety of information technology topics, many of which lead to professional certifications.

Custom Learning Solutions

This delivery option has been created for companies and governments that value customized learning solutions. For them, our consultancy-based approach of developing targeted education solutions is most effective at helping them meet specific objectives.

Self-Paced and Multimedia Products

This delivery option offers self-paced program titles in interactive CD-ROM, videotape and audio tape programs. In addition, we offer custom development of interactive multimedia courseware to customers and partners. Call us at 1-888-427-4228.

Electronic Delivery of Training

Our network-based training service delivers efficient competency-based, interactive training via the World Wide Web and organizational intranets. This leading-edge delivery option provides a custom learning path and "just-in-time" training for maximum convenience to students.

Global Knowledge Courses Available

Microsoft

- Windows 2000 Deployment Strategies
- Introduction to Directory Services
- Windows 2000 Client Administration
- Windows 2000 Server
- Windows 2000 Update
- MCSE Bootcamp
- Microsoft Networking Essentials
- Windows NT 4.0 Workstation
- Windows NT 4.0 Server
- Windows NT Troubleshooting
- Windows NT 4.0 Security
- Windows 2000 Security
- Introduction to Microsoft Web Tools

Management Skills

- Project Management for IT Professionals
- Microsoft Project Workshop
- Management Skills for IT Professionals

Network Fundamentals

- Understanding Computer Networks
- Telecommunications Fundamentals I
- Telecommunications Fundamentals II
- Understanding Networking Fundamentals
- Upgrading and Repairing PCs
- DOS/Windows A+ Preparation
- Network Cabling Systems

WAN Networking and Telephony

- Building Broadband Networks
- Frame Relay Internetworking
- Converging Voice and Data Networks
- Introduction to Voice Over IP
- Understanding Digital Subscriber Line (xDSL)

Internetworking

- ATM Essentials
- ATM Internetworking
- ATM Troubleshooting
- Understanding Networking Protocols
- Internetworking Routers and Switches
- Network Troubleshooting
- Internetworking with TCP/IP
- Troubleshooting TCP/IP Networks
- Network Management
- Network Security Administration
- Virtual Private Networks
- Storage Area Networks
- Cisco OSPF Design and Configuration
- Cisco Border Gateway Protocol (BGP) Configuration

Web Site Management and Development

- Advanced Web Site Design
- Introduction to XML
- Building a Web Site
- Introduction to JavaScript
- Web Development Fundamentals
- Introduction to Web Databases

PERL, UNIX, and Linux

- PERL Scripting
- PERL with CGI for the Web
- UNIX Level I
- UNIX Level II
- Introduction to Linux for New Users
- Linux Installation, Configuration, and Maintenance

Authorized Vendor Training

Red Hat

- Introduction to Red Hat Linux
- Red Hat Linux Systems Administration
- Red Hat Linux Network and Security Administration
- RHCE Rapid Track Certification

Cisco Systems

- Interconnecting Cisco Network Devices
- Advanced Cisco Router Configuration
- Installation and Maintenance of Cisco Routers
- Cisco Internetwork Troubleshooting
- Designing Cisco Networks
- Cisco Internetwork Design
- Configuring Cisco Catalyst Switches
- Cisco Campus ATM Solutions
- Cisco Voice Over Frame Relay, ATM, and IP
- Configuring for Selsius IP Phones
- Building Cisco Remote Access Networks
- Managing Cisco Network Security
- Cisco Enterprise Management Solutions

Nortel Networks

- Nortel Networks Accelerated Router Configuration
- Nortel Networks Advanced IP Routing
- Nortel Networks WAN Protocols
- Nortel Networks Frame Switching
- Nortel Networks Accelar 1000
- Comprehensive Configuration
- Nortel Networks Centillion Switching
- Network Management with Optivity for Windows

Oracle Training

- Introduction to Oracle8 and PL/SQL
- Oracle8 Database Administration

Custom Corporate Network Training

Train on Cutting Edge Technology

We can bring the best in skill-based training to your facility to create a real-world hands-on training experience. Global Knowledge has invested millions of dollars in network hardware and software to train our students on the same equipment they will work with on the job. Our relationships with vendors allow us to incorporate the latest equipment and platforms into your on-site labs.

Maximize Your Training Budget

Global Knowledge provides experienced instructors, comprehensive course materials, and all the networking equipment needed to deliver high quality training. You provide the students; we provide the knowledge.

Avoid Travel Expenses

On-site courses allow you to schedule technical training at your convenience, saving time, expense, and the opportunity cost of travel away from the workplace.

Discuss Confidential Topics

Private on-site training permits the open discussion of sensitive issues such as security, access, and network design. We can work with your existing network's proprietary files while demonstrating the latest technologies.

Customize Course Content

Global Knowledge can tailor your courses to include the technologies and the topics which have the greatest impact on your business. We can complement your internal training efforts or provide a total solution to your training needs.

Corporate Pass

The Corporate Pass Discount Program rewards our best network training customers with preferred pricing on public courses, discounts on multimedia training packages, and an array of career planning services.

Global Knowledge Training Lifecycle

Supporting the Dynamic and Specialized Training Requirements of Information Technology Professionals

- Define Profile
- Assess Skills
- Design Training
- Deliver Training
- Test Knowledge
- Update Profile
- Use New Skills

Global Knowledge

Global Knowledge programs are developed and presented by industry professionals with "real-world" experience. Designed to help professionals meet today's interconnectivity and interoperability challenges, most of our programs feature hands-on labs that incorporate state-of-the-art communication components and equipment.

ON-SITE TEAM TRAINING

Bring Global Knowledge's powerful training programs to your company. At Global Knowledge, we will custom design courses to meet your specific network requirements. Call (919)-461-8686 for more information.

YOUR GUARANTEE

Global Knowledge believes its courses offer the best possible training in this field. If during the first day you are not satisfied and wish to withdraw from the course, simply notify the instructor, return all course materials and receive a 100% refund.

REGISTRATION INFORMATION

In the US:
call: (888) 762–4442
fax: (919) 469–7070
visit our website:
www.globalknowledge.com

SYNGRESS SOLUTIONS...

AVAILABLE NOW
ORDER at
www.syngress.com

HACK PROOFING YOUR NETWORK INTERNET TRADECRAFT

Systems and software packages are being connected to the Internet at an astounding rate. Many of these systems and packages were not designed with security in mind. IT professionals need to keep their systems secure: this book shows them how to make a meaningful security assessment of their own systems, by helping them to think like a hacker. Using forensics-based analysis, this book gives the reader crucial insights into security, classes of attack, diffing, decrypting, session hijacking, client and server holes, and choosing secure systems.

ISBN: 1-928994-15-6

Price: $49.95

AVAILABLE
DECEMBER 2000
www.syngress.com

MISSION CRITICAL! INTERNET SECURITY

This essential reference focuses on security at the protocol layer, specifically on Internet Protocol (IP), the protocol that is the standard for directing traffic on the Internet. Readers will find coverage of host security, WAN connectivity, Lucent and Cisco hardware, and firewall architecture.

ISBN: 1-928994-20-2

$59.95

AVAILABLE
JANUARY 2001
www.syngress.com

MISSION CRITICAL! WINDOWS 2000 E-MAIL CONFIGURATION

PC users, both at home and business, will find this book valuable for its coverage of all the popular e-mail clients, such as Outlook and Outlook Express. In addition, System and E-mail Administrators will find the coverage of large system E-mail providers such as Exchange, indispensable. The book discusses installation and management of all the major e-mail programs, as well as mobile e-mail issues, Web-based e-mail, e-mail security, and implementation of e-mail within multinational companies.

ISBN: 1-928994-25-3

Price: $49.95

solutions@syngress.com

SYNGRESS®