SYNGRESS®

4 FREE BOOKLETS
YOUR SOLUTIONS MEMBERSHIP

SYNGRESS 4 FREE E-BOOKLETS PUBLISHING

AVIEN Malware Defense Guide for the Enterprise

Stop the Stalkers on Your Desktop

- Complete Coverage of the Relationship Between Enterprise Security Professionals, Customers, Vendors, and Researchers
- In-depth Consideration of Key Areas of the 21st Century Threat Landscape
- Systems Security and DIY Defense Using a Range of Specialist Detection and Forensic Techniques and Tools

David Harley CISSP, Antivirus Researcher, former manager of the Threat Assessment Centre for the U.K.'s National Health Service

Ken Bechtel
Michael Blanchard
Henk K. Diemer
Andrew Lee
Igor Muttik
Bojan Zdrnja

FOREWORD BY ROBERT S. VIBERT
AVIEN ADMINISTRATOR

AVIEN Malware Defense Guide for the Enterprise

David Harley, CISSP,
Antivirus Researcher, former manager of the Threat Assessment Centre for the U.K.'s National Health Service

Foreword by **Robert S. Vibert,** AVIEN Administrator

Ken Bechtel
Michael Blanchard
Henk Diemer
Andrew Lee
Igor Muttik
Bojan Zdrnja

KEY	**SERIAL NUMBER**
001	HJIRTCV764
002	PO9873D5FG
003	829KM8NJH2
004	BAL923457U
005	CVPLQ6WQ23
006	VBP965T5T5
007	HJJJ863WD3E
008	2987GVTWMK
009	629MP5SDJT
010	IMWQ295T6T

PUBLISHED BY
Syngress Publishing, Inc.
Elsevier, Inc.
30 Corporate Drive
Burlington, MA 01803

AVIEN Malware Defense Guide for the Enterprise

Printed in the United States of America
1 2 3 4 5 6 7 8 9 0

ISBN 13: 978-1-59749-164-8

Publisher: Amorette Pedersen
Technical Editor: David Harley
Cover Designer: Michael Kavish
Copy Editor: Judith Eby
Indexer: Rich Carlson

For information on rights, translations, and bulk sales, contact Matt Pedersen, Commercial Sales Director and Rights, at Syngress Publishing; email m.pedersen@elsevier.com.

Lead Author and Technical Editor

David Harley CISSP (Lead Author, Technical Editor) has written or contributed to over a dozen books in the security and education fields, including "Viruses Revealed" (Osborne). He is a frequent presenter at security conferences and has many research papers to his credit, as well as consumer-level articles in many areas of computing. He runs the Small Blue-Green World security and publishing consultancy, and his roles there include authoring, reviewing and editing, antimalware and security research, and providing consultancy to the antivirus industry. He is also qualified in security audit (BS7799 Lead Auditor) and ITIL Service Management. For five years he ran the Threat Assessment Centre for the UK's National Health Service, specializing in malware and email abuse management consultancy. He previously worked in systems, application and network support for a major cancer research charity.

David's academic roots are in Computer Science, Social Sciences and Medical Informatics. His further qualifications include BS7799 Lead Auditor, ITIL Service Management, and Medical Informatics. His affiliations include the Red Team at QuantumLabs, a system testing and validation service, Team Anti-Virus, and the WildList Organization. He is a charter member of AVIEN and AVIEWS, serving as Disciplinary Committee Chairman, Adjunct Administrator of AVIEN, and from mid-2007 will serve as Transitional Administrator and CDO during the restructuring of AVIEN.

David would like to thank all his co-authors, not only for the excellent content they contributed but for their support, suggestions and encouragement. Many other members of AVIEN and AVIEWS also contributed input in the early stages of the book planning (about forty people were subscribed to the book's dedicated mailing list, over time), and they also deserve thanks. In particular:

- His wife Jude, who not only contributed content and late-night discussion, but put up with the ongoing hormonal changes and mood swings of an expectant author with patience and good humor.
- Andrew Lee and Robert Vibert for their unfailing support during some very rocky moments. Extra brownie points go to Andrew for his timely assistance in proofreading.
- The AVIEN Advisory Board and Disciplinary Committee and their individual members for their support and advice at times of extreme stress.

- Paul Dickens, whose cartoons grace the book's web site at www.smallblue-greenworld.co.uk/pages/avienguide.html.
- Mary Landesman for discussion on chapter planning.
- Jeannette Jarvis, who first suggested the idea of an AVIEN book to him.

He also owes special thanks to Amorette Pedersen and Andrew Williams of Syngress/Elsevier for their unfailing patience and support, even during the occasional prima donna outburst from the technical editor. ☺

There is forensic evidence of David's sticky fingers all over this book, but particularly Chapters 1, 2, 4, 6, 8, 10 and 11.

Foreword Author

Robert S. Vibert is the administrator and CDO of the Anti-Virus Information Exchange Network (AVIEN), the growing network of Security Professionals working in organizations with 1500 or more PCs who discuss Anti-Virus topics and keep each other informed about upcoming malware threats. He also acts as senior advisor to the administrator of AVIEWS (Anti-Virus Information & Early Warning System), AVIEN's sister organization, which brings together security specialists and researchers at both vendor and customer organizations. Robert has worked for more than 25 years as a consultant, mentoring and helping companies and individuals get the most out of their resources.

Author of five books and more than 200 articles on management, computer security and operations, Robert has also worked as a senior consultant for a major international consulting firm, is regularly interviewed by the media for his expert insights on computer security, and serves as an adviser to Canadian government departments. Currently, he acts as a mentor to several entrepreneurs and is developing the *Missing Lin*k series of books, workbooks, CDs and DVDs to provide practical information and processes to get the success you want in life in the areas of finance, relationships, emotional health, career and personal development.

As well as contributing the foreword on behalf of AVIEN, Robert also co-wrote Chapter 1.

Contributors

Paul Baccas is a researcher at Sophos plc, the UK security company. After reading Engineering Science at Exeter College, Oxford, he worked in various technical roles at Sophos, and is now mainly engaged in spam research. He is a frequent contributor to Virus Bulletin.

Paul assisted with technical editing on a number of chapters.

Ken Bechtel has been involved in corporate malware defense since 1988. His work history includes working in the Virus Lab at NCSA (later ICSA), performing virus analysis and Antivirus Product Certifications, as well as user education. He has worked and consulted for all levels of business, from small businesses to Fortune 500 companies. He is the author of several papers published by Security Focus, Virus Bulletin, and several other trade magazines. He has appeared 26 times on local and national news for interviews concerning various malicious code threats. Ken is a Founding Member and Adjunct Administrator of the Anti-Virus Information Exchange Network (AVIEN), member of Association Anti-Virus Asian Researchers (AAVAR), WildList Reporter since 1998, Founder of Team Anti-Virus, and member of several unofficial associations. Several of his papers and articles have been printed in Security Focus, Virus Bulletin, and several other trade magazines. His biggest literary contribution so far has been the "Handbook of Corporate Malware Protection."

Ken is devoted to his family, and enjoys all manner of outdoor sports, from fishing and camping to several shooting sports.

Ken co-wrote Chapters 1, 2 and 6.

Michael P. Blanchard, CISSP, GCIH (gold), CCSA-NGX and MCSE, has been an IT professional for over 16 years, and is currently a member of AVIEN. His current major duties include Malware analysis/ protection and assessment, vulnerability analysis and assessment, and other daily activities. Apart from some in-house training documents, Mike is also the author of the definitive whitepaper on the FunLove virus that he wrote to achieve his SANS GCIH gold certification (#350) in 2002, at www.giac.org/certified_professionals/practicals/GCIH/0350.php. Mike takes pride in his current professional role serving in the CIO's Office of Information Security and Risk Management as the Senior Antivirus Security Engineer overseeing the malware protection on a global scale at EMC^2 Corporation in Westborough, Mass, a role that he's had since 1999.

Before that, it was Mike's father who introduced him to the wonders of computers and building electronic devices back in the mid to late 1970's and up to programming in Fortran and Pascal in the mid 1980's on his father's Atari 800 and his High School's PDP-11. To this day, Mike says that he learned everything he knows from his Dad, and is happy to still be learning from him now that Mike is a Dad with his own two children.

In his spare time, Mike can be seen wandering around Renaissance faires, making Chainmaille armor and jewelry, spending time with his family, performing CubMaster duties for his local CubScout pack, or leveling up with friends in the computer MMORPG Everquest 2. Mike would like to thank his parents and his wife and two children for bearing with him and being very supportive while he locked himself in his computer room with his headphones on for months to complete his contribution to this project. Mike wishes to dedicate his contribution to his loving wife and children, and his late best friend Jim: he would have been proud.

Mike co-wrote Chapter 9.

Tony Bradley (CISSP-ISSAP) is the author of Essential Computer Security, co-author of *Hacker's Challenge 3*, and has contributed chapters to many other books. Tony is the Guide for the Internet/Network Security site on About.com, a part of the New York Times Company, where he has more than 30,000 subscribers to his weekly newsletter. He has written for a variety of other Web sites and publications, including PC World, SearchSecurity.com, WindowsNetworking.com, Smart Computing Magazine and Information Security Magazine. Currently a Security Consultant with BT INS, Tony has driven security policies and technologies for endpoint security and incident response for Fortune 500 companies for over 6 years. Tony is a CISSP (Certified Information Systems Security Professional) and ISSAP (Information Systems Security Architecture Professional). He is Microsoft Certified as an MCSE (Microsoft Certified

Systems Engineer) and MCSA (Microsoft Certified Systems Administrator) in Windows 2000, and he is recognized by Microsoft as an MVP (Most Valuable Professional) in Windows security.

Other books to which Tony has contributed include *Winternals: Defragmentation, Recovery, and Administration Field Guide, Combating Spyware in the Enterprise, Emerging Threat Analysis,* and *Botnets: The Killer Web App*. He is the lead technical editor and contributing author to the upcoming *PCI Compliance: Understand and Implement Effective PCI Data Security Standard Compliance*.

Tony co-wrote Chapter 4.

Henk K. Diemer (CISSP, MSC in Bio Physics) lives in Utrecht, in the Netherlands, with his wife Ieneke and three school age children. He brought to this book his experience as an independent AV management specialist with over 28 years – mostly – international ICT management experience in both the private and public sectors. Using computers and programming for his research since 1972, he has dedicated himself since 1996 to limiting the losses related to malicious code. Henk currently works for a large global Fortune 500 IT services company, as a senior IT security advisory specialist. Before that, he worked for a large Dutch multinational bank for 20 years, until IT there was largely outsourced in 2005.

Henk initiated, among other things, a workgroup for Dutch AV experts under the authority of the FI –ISAC NL and Dutch Banker Association, for sharing lessons learned and to help manage high profile malware incidents in banking. Today, his focus is primarily on improving local, regional and global services in the context of outsourced IT AV services, and to assist security management functions in creating and maintaining optimal conditions for success in outsourcing AV services.

Henk has had the pleasure of working with many other independent and dedicated AV specialists in AVIEN, Virus Bulletin and the Anti-Phishing Working Group, and many others committed to the sharing of best practices or lessons learned. He wishes to express his warm gratitude to all who made his contribution to this book possible.

Henk wrote most of Chapter 7.

Ken Dunham is Director of the Rapid Response Team at iDefense, a VeriSign company, overseeing all Rapid Response and global cyber-threat operations. He frequently briefs upper levels of federal cyber security authorities on emerging threats, and regularly interfaces with vulnerability and geopolitical experts to assemble comprehensive malicious code intelligence and to inform the media of significant cyber-threats. Ken is regularly rated as a top speaker at events including the Forrester Security Summit, GFIRST, ISSA, Pentagon and others. He regularly discovers new malicious code, has written anti-virus software for Macintosh, and has written about malicious code for About.com, SecurityPortal, AtomicTangerine and Ubizen. He is a member of AVIEWS, InfraGard, an RCG Information Security Think Tank, CME, International High Tech Crime Investigation Association, the WildList Organization and others. He is also a certified reverse engineer and regularly analyzes top threats of concern for top tier clients.

Ken authored Bigelow's Virus Troubleshooting Pocket Reference, "The HyperCard Roundup" (on HyperText programming), and is a regular columnist for two information security magazines. He is also the founder and President of the Boise, Idaho, Information Systems Security Association chapter. He is also the founder and President of the Idaho InfraGard chapter, in conjunction with the FBI. He holds several security certifications, serves as the VeriSign Forum for Incident Response and Security Teams (FIRST) lead representative, and is a member of the North American Incident Response Team (NAIRT).

Ken co-wrote Chapter 5.

Enrique González is a Senior Virus Researcher at Microsoft Corporation. Before joining Microsoft, Enrique was a Senior Security Researcher with Websense where he lead Websense Security Labs' EMEA team, being also spokesperson for the Lab in the EMEA region. Enrique's background includes positions at Panda Software where he analyzed and researched malware from old DOS viruses to the latest threats. He is a frequent presenter at conferences and events such as APWG, AVAR, CISCI, and so on. His presenting work includes malware cases and technologies, research on future attack vectors such as VoIP, as well as current and upcoming threats. Enrique also co-founded a security systems company in Spain. Enrique's contribution to the book would have not been possible without his parents' hard work and support of his education. His wife and his children have also played a key role, supporting him and bringing him the joy he needs to keep working hard for them.

Enrique co-wrote Chapter 5.

Judith Harley teaches ICT and business communications at a secondary school in the UK. Even before qualifying as a teacher, she was a qualified adult training instructor and assessor, and also worked in user support and systems and security administration in the public sector. She has many years of experience in writing training manuals, policies, FAQs and other documentation, and has published articles in educational periodicals. She was co-author, with David Harley and Eddy Willems, of "Teach your children well" for the 2005 Virus Bulletin International Conference, and also co-wrote two chapters for "Coming of Age – an introduction to the new world wide web", 2nd Edition (Freedman).

Judith co-wrote Chapter 8.

Andrew Lee (CISSP) is Chief Research Officer of ESET LLC. He was a founding member of the Anti-Virus Information Exchange Network (AVIEN) and its sister group AVIEWS (AVIEN Information & Early Warning System), is a member of AVAR and a reporter for the WildList organisation. He was previously at the sharp end of malware defense as a systems administrator in a large government organisation.

Andrew is author of numerous articles on malware issues, and is a frequent speaker at conferences and events including ISC2 Seminars, AVAR, Virus Bulletin and EICAR. When he is not sitting at the computer or in an airport somewhere, he enjoys reading, photography, playing guitar, and the martial art of Ki-Aikido.

Andrew co-wrote Chapters 10 and 11.

Jim Melnick is Director of Threat Intelligence at iDefense, leading the global threat intelligence group that focuses on cyber threats around the world, from nation states and hacker groups to new technologies. His "Weekly Threat Report" on cyber threats, which he founded and edits for iDefense/VeriSign, was dubbed by Business Week in 2005 as including "some of the most incisive analysis in the business." Prior to joining iDefense, Jim served with distinction as a civilian analyst for more than 16 years in the U.S. Army and the Defense Intelligence Agency in a variety of roles, including intelligence, psychological operations, international warning issues, information operations and Russian affairs.

Jim has been published in numerous military and foreign affairs journals, and has received numerous military and related awards, including a Presidential Commission medal for his work on the Y2K problem in support of the National Intelligence Council. He also recently retired from the U.S. Army Reserves as a Colonel in Military Intelligence. His last military assignment was with the Office of the Assistant Secretary of Defense for Networks and Information Integration. Jim has a Master of Arts in National Security and Strategic Studies from the U.S. Naval War College, a Master of Arts in Russian studies from Harvard University, and a Bachelor of Arts with Honors in Political Science from Westminster College.

Jim co-wrote Chapter 5.

Igor Muttik, PhD is a senior architect with McAfee Avert™. He started researching computer malware in 1980s when anti-virus industry was in its infancy. He is based in the UK and worked as a virus researcher for Dr. Solomon's Software where he later headed the anti-virus research team. Since 1998 he has run Avert Research in EMEA and switched to his architectural role in 2002. Igor is a key contributor to the core security technology at McAfee. He takes particular interest in new emerging malware techniques, and in the design of security software and hardware appliances. Igor holds a PhD degree in physics and mathematics from Moscow University. He is a regular speaker at major international security conferences and a member of the Computer Antivirus Research Organization.

Igor wrote Chapter 3.

David Phillips has been working at The Open University (OU) since 1986, transferring into computer support full time in mid-1996. He has spent over 14 years in the antivirus field, involved in the implementation and support of staff and students at the OU. A speaker at the 1998, 1999, 2001 and 2003 Virus Bulletin conferences, he has also presented for SecureIT Europe and others including workshops at NetFocus2006. In 2003 he created a short course at the OU, T187 Vandalism in Cyberspace aimed at educating the home users in malware and malware protection issues which is currently being presented two times a year, until 2009.

David co-wrote Chapter 8.

Paul Schmehl is Senior Information Security Analyst at the University of Texas at Dallas, and has many years of experience in antimalware administration. A number of his articles have been published by SecurityFocus and Claymania, on such topics as AV software evaluation, firewall and AV product reviews, and protection for the enterprise and for small businesses. He is a frequent contributor to security lists, and a founder member of AVIEN. His presentation on "Barbarians at the Gateways: Defeating Viruses in EDU" has been featured at SIGUCCS and EDUTEX.

Paul co-wrote Chapter 6.

James M. Wolfe, CHS-V is the Technical Director of the European Institute for Computer Anti-Virus Research (EICAR). His other memberships include AVIEN, Team Anti-Virus, the US-CERT CME project, and he is a reporter for the WildList Organization. He is an Associate Member of the prestigious Computer Anti-Virus Research Organization (CARO). He is also an Adjunct Professor at the University of Central Florida and Webster University, teaching Information Security, Ethics, Counter-Terrorism and Homeland Security. He has a Bachelor of Science degree in Management Information Systems and a Master of Science degree in Change Management from the University of Florida. He holds a Level 5 Certification in Homeland Security from the American College of Forensic Examiners Institute. Currently, he is working on a Bachelor's degree in Anthropology. He plans to begin his Doctorate soon.

He has published articles in the Virus Bulletin and EICAR magazines. He co-authored a chapter in the 2003-2005 editions of the Handbook of Information Security Management by Micki Krause and Hal Tipton. He is a five-time honoree in "Who's Who in America." He routinely presents at conferences all over the world, usually in the Anti-Virus, Terrorism, and Security arena.

James would like to dedicate his contribution to Krista and Cymoril, who never waver in their support even when the trolls are attacking at 3am, and to Mom for giving her wisdom and strength.

James co-wrote chapter 1.

Bojan Zdrnja (GCIA, CISSP, RHCE) is Security Implementation Specialist at the University of Auckland, New Zealand. He previously worked as a security consultant and security team leader at the Faculty of Electrical Engineering and Computing, University of Zagreb, as part of a commercial team working on external projects. He was also a member of several Incident Response Teams for the Croatian CERT. He is a handler for the Internet Storm Center (ISC) and is also on the SANS Advisory Board and one of the GIAC Gold Advisors. Specialized areas of interest include analyzing malware, forensic analysis, incident handling. His publications include a security column for a Croatian computer magazine, the book *What Are Computer Viruses?* (Syspring), and diaries for the Internet Storm Center.

Bojan co-wrote Chapter 9.

Contents

Foreword

This book recognizes that the combined membership of AVIEN and AVIEWS are uniquely qualified to pass on their combined knowledge and the benefits of their experience at the leading edge of anti-malware defense to others facing the challenges of new generations of malware.

The collective membership of the two organizations comprises many of the brightest minds working on malware-related issues.

This book also demonstrates the value of combining the practical research skills of some members with the writing experience of others. The end result is a wonderful blend of deeply researched and yet easily accessible information.

David Harley was the logical choice for heading up this project, not only because he has been involved with AVIEN since its earliest days, but also due to his extensive experience in managing very large installations of anti-virus defenses and his impeccable credentials in writing and editing in the security arena, especially in antivirus.

David has also extensive research experience, independence from commercial influence and the respect of his peers in the anti-malware field, a field that has seen his contributions for many years.

—Robert S. Vibert
Administrator, Anti-Virus Information Exchange Network

Preface

This guide begins with a brief discussion of the Anti-Virus Information Exchange Network (AVIEN) and its sibling the Anti-Virus Information and Early Warning System (AVIEWS). AVIEN members include some of the most knowledgeable systems administrators, security managers, and independent researchers around, representing the best-protected large organizations in the world, and employing millions of users.

All AVIEN members are also members of AVIEWS, which also includes representatives of most of the major security vendors and specialist researchers in the antimalware arena. This guide is thus a unique collaboration between the security vendors and researchers who know the most about malicious code and the technology for dealing with it, and the most knowledgeable security administrators using those technologies in real-life customer situations. It offers a unique insight into the nuts and bolts of enterprise security management, combining technical depth and strategic breadth of vision in the difficult area of malicious code management. No one in the security management business can afford to ignore it.

You'll find out a lot more about AVIEN and AVIEWS in the first chapter. (And if you're doing the same sort of job that our members are, you might even want to consider joining us.) We anticipate that the groups who'll particularly benefit from this book (apart from information security people in general) include generalist Information Technology (IT) managers, support staff, Human Resource (HR) professionals, and senior management both in the public and private sectors. Teachers and other educationalists should certainly find it of practical use, but it should have applications in some more academic courses, too. We'd be surprised if many law enforcement professionals with a technological remit didn't find it useful, though our take on forensics is more Do it Yourself (DIY) than CSI.

From the early days of malicious software, the administrators responsible for protecting a corporate site have had to pick a path along the Misinformation Superhighway, between muddy puddles of misinformation:

- Marketing hype
- Newspaper and magazine articles based on recycled press releases and sheer guesswork
- Laws and policies based on window dressing and political expediency
- Fear, uncertainty and doubt

How is the hard-pressed system administrator or security manager to make sense of it all?

Books about malicious software tend to come from a restricted range of experience. Some of the best have been written by people representing the research community. These are often of most use and interest to other researchers, though the most "interesting" information tends to be exchanged over less public channels. A few have been written by virus writers (or hacker wannabes), where the content is often wildly fanciful and rarely of real use in terms of defending an enterprise. Most are written by people on the fringes of anti-virus: academics, specialist journalists, security generalists, and even lawyers. Uniquely, this book combines the expertise of truly knowledgeable workers at the coalface with that of experienced researchers with unmatched experience in the analysis of malicious code, and the research and development of defensive programs.

This book owes some of its genesis to a phone call I received from Jeannette Jarvis a couple of years ago, saying, in effect, "Wouldn't it be great to write a security book making use of the expertise in AVIEN, written by people who learned what the issues were and how to deal with them by actually doing the job, instead of telling the people with real experience what they should be doing?" Over the next year or so, I heard variations on the same thought from several people, and had to agree. So, when I finally managed to escape from the bureaucracy that was paying my salary and sucking my blood at the time, and move into full-time authoring and editing, I jumped at the chance to put that idea into practice.

While some of the authors here are new to book authoring, nearly all are experienced writers and presenters of security-related conference papers, articles, white papers, technical documentation, and manuals, and the team includes several individuals who are well known as authors of security books in their own right, including David Harley, Robert Vibert, Tony Bradley, Ken Bechtel, Bojan Zdrnja, and Ken Dunham, all of whom have particular experience and expertise in the management of malicious code. And, I'm pleased to say, one or two of the less experienced contributors have been pleased enough with the result to want to do more. I hope our readers will be as pleased with the result as we are.

—David Harley
Lead Author and Technical Editor

Introduction

The *AVIEN Guide to Managing Malware in the Enterprise* covers the following main areas:

- The relationship between enterprise security professionals, customers, vendors and researchers, stripping away the myths
- In-depth consideration of key areas of the 21st century threat landscape, especially the bits populated by malicious software
- Defense in depth as the cornerstone of enterprise security
- Systems security and DIY defense using a range of specialist detection and forensic techniques and tools
- Education and communications
- Governance, especially in relation to outsourcing.

We focus particularly on malware and anti-malware technologies (anti-virus, anti-spam, anti-Trojan, anti-adware, anti-spyware, corporate and desktop firewalls, gateway filtering and so on), but it isn't practical to isolate these from other security technologies, so some consideration of associated product and service types is inevitable. Despite the input of a number of contributors from the security industry, the book explores and clarifies core concepts rather than particular brands, though it does look in some detail at specific tools for network defense and malcode analysis.

Chapter 1 ("Customer Power") wasn't particularly intended to be dominated by Team Anti-Virus, a group of independent antivirus researchers, but it seems to have worked out that way. In the first section, Robert Vibert, founder of AVIEN and

AVIEWS, recounts the history of these two organizations, a story of more than historical interest. In the second section, David Harley takes up the theme of the sometimes stormy relationship between the antivirus industry and its customers, and tries to dispel some common myths. James Wolfe then takes up the baton to consider the roles of the independent researcher, the vendor-employed specialist, and the corporate security specialist. Finally, David Harley looks at security certification in the context of malware research, and he and Ken Bechtel consider whether there is a need for a specialist certification for antimalware administrators.

Chapter 2 kicks off with a consideration of the thorny issue of malware nomenclature by Ken Bechtel, and then David Harley takes a brief historical look at how we got here, before expanding on some of the (mostly) malware-related problems we face today (rootkits, spam, phishing, muledriving, hoaxes).

In Chapter 3, Igor Muttik brings his considerable experience in malware research to bear onto threats and countermeasures in the context of the World Wide Web, while Chapter 4, by Tony Bradley and David Harley, tackles bots and botnets, arguably Public Cyber-Enemy Number One.

Chapter 5 takes us into the underworld: David Harley reviews the history of old-school virus writing, while Enrique Gonzales considers some criminal business models. Ken Dunham and Jim Melnick offer a fascinating case study, concerning a Chinese hacking group. Finally, Enrique looks into his crystal ball in the hope of predicting some future malware hotspots.

Chapter 6 covers Defense in Depth: Paul Schmehl takes a broad look at DiD in the enterprise, and Ken Bechtel covers many of the implementation angles, while David Harley looks at some specific tools and technologies. Henk Diemer takes another view in Chapter 7, where he offers some sound advice on how to avoid the perils and pitfalls of outsourcing, incorporating a few horrible examples of how not to do it from David Harley's casebook. In Chapter 8, David Phillips offers some insights into user education from an educationalist's perspective, while David and Judith Harley look at various aspects of security in schools and other educational establishments.

Michael Blanchard and Bojan Zdrnja take us back to the hands-on, hands-dirty approach to security management in chapter 9, considering malware analysis and forensics techniques and tools, starting from basics and progressing to advanced forensics.

In Chapter 10, David Harley and Andrew Lee continue the D-I-Y theme, discussing at length some of the thorny issues around the evaluation and testing of antimalware software. Finally, Robert Vibert, Andrew Lee and David Harley borrow Enrique's crystal ball to look at future developments in AVIEN and AVIEWS, incorporating an unashamed attempt to entice you into joining us.

OK, not quite finally. Since even a book of this length can't tell you everything about enterprise security, we include some further printed and online resources you may find useful, and since inconsistent terminology appertaining to malicious software sometimes baffles even the Great and the Good in other areas of security, we include a fairly brief malware-specific glossary.

—David Harley
Lead Author and Technical Editor

Chapter 1

Customer Power and AV Wannabes

Solutions in this chapter:

- History of AVIEN And AVIEWS
- Antivirus Vendor Image
- So You Want to Be a Bona Fide Computer Anti-malware Researcher?
- You Should be Certified
- Should There Be a Vendor-independent Malware Specialist Certification?

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

In the first section of this chapter, Robert Vibert, founder of the Anti-Virus Information Exchange Network (AVIEN) and the Anti-Virus Information and Early Warning System (AVIEWS), relates the historical origins and development of these two closely linked organizations. His story is important. While these are significant and interesting organizations in their own right, their story also reflects an important phase in the history of viruses and virus management. In the few years since AVIEN was founded, we've seen the focus shift across the board from virus management to malicious software (malware) management. Furthermore, where a Gods and Ants view once predominated in the antivirus industry, there is a more harmonious relationship between the antivirus industry and other security professionals.

In fact, it sometimes seems that everyone outside the antivirus industry is a virus/antivirus expert, in his or her own estimation (False Authority Syndrome). On the other hand, it also seems that people within the security industry think they have sole custody of all security knowledge, and that the rest of the world knows just enough to put their hands in their corporate pockets and pay for the solutions that are offered them. The truth is out there somewhere between "AV knows nothing" and "AV knows everything." In the second section, David Harley looks at the uneasy relationship between the anti-malware industry and its customers, in the hope of finding it.

Various members of Team Anti-Virus, a loose grouping of independent antivirus researchers, have been considering the issues around professional expertise and qualifications inside and outside the security industry for some years. In the last section, James Wolfe compares the roles of the independent researcher, the vendor-employed specialist, and the corporate security specialist, and David Harley and Ken Bechtel look in more detail at certification issues.

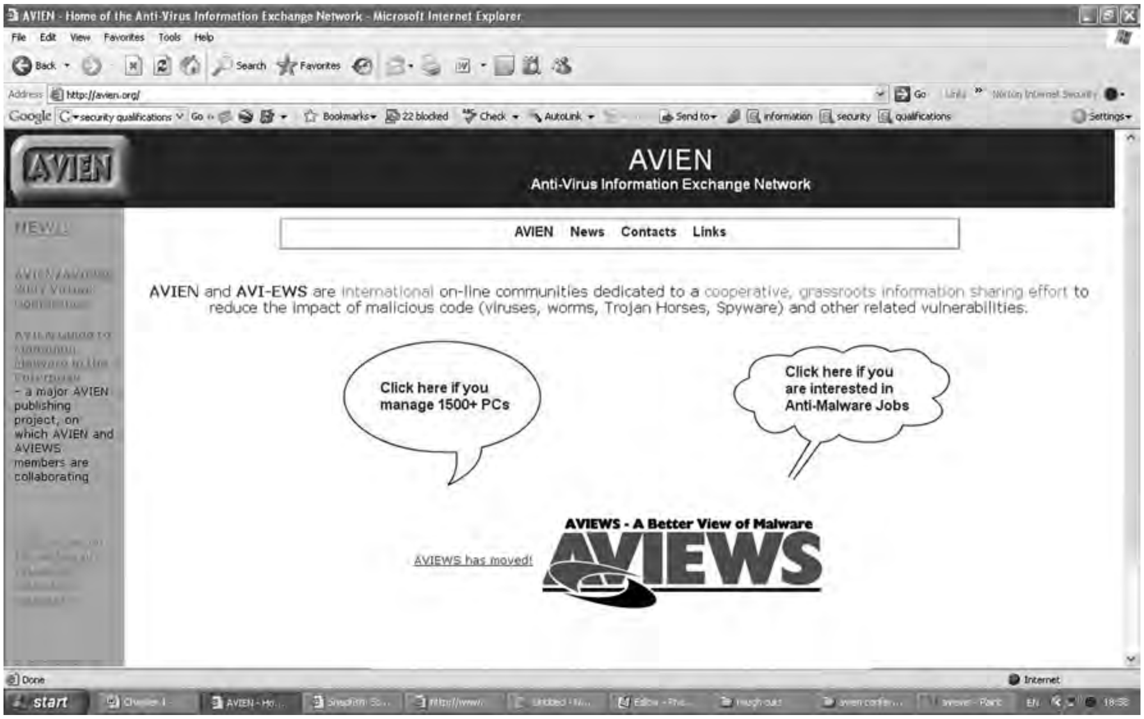
History of AVIEN and AVIEWS

This isn't a book about AVIEN (see Figure 1.1) and AVIEWS, though it starts and finishes with them. If there is something really important about these groups, though, it's their membership, combining the talents of a high percentage of the most able administrators, researchers, support professionals, and security experts in the world. And telling you something about them will tell you something about the world we all live in.

Background: So Who Is Robert Vibert?

For the years 1993 to 1999, I was heavily involved in the antivirus world. My companies in Portugal and Canada sold millions of dollars worth of antivirus software to large corporations, government agencies, departments, and financial institutions. During those years, it was said that I walked, talked, lived, breathed, slept, and dreamt about antivirus software and solving the malware problems faced by my customers.

Figure 1.1 The AVIEN Welcome Screen (But Members Get To See A Lot More!)


AVIEN - Home of the Anti-Virus Information Exchange Network - Microsoft Internet Explorer
File Edit View Favorites Tools Help
http://avien.org/
AVIEN
AVIEN
Anti-Virus Information Exchange Network
NEW!
AVIEN News Contacts Links
Click here if you manage 1500+ PCs
Click here if you are interested in Anti-Malware Jobs
AVIEWS has moved!
AVIEWS - A Better View of Malware
AVIEWS
Done
Internet
start

– a major AVIEN publishing project, on which AVIEN and AVIEWS members are collaborating

AVIEN and AVI-EWS are international on-line communities dedicated to a cooperative, grassroots information sharing effort to reduce the impact of malicious code (viruses, worms, Trojan Horses, Spyware) and other related vulnerabilities.

In the middle of 1999, I jumped ship from the sales world, due in part to McAfee's taking over Dr. Solomon's, the company that made the antivirus company that my companies had been selling as our flagship product. The other part of this decision was the fact that prior to selling software, I had worked for a number of years as an independent consultant and I longed to return to that role.

From 1999 to 2001, I once again acted as an independent security consultant, advising the Canadian government on antimalware defenses and providing security audit services. It was also during this time that AVIEN took shape. I'll tell that story in a moment, but first, some context.

AV Vendor/Researcher Lists and Groups

For many years, security specialists around the world working to defend their organizations against attacks from viruses, worms, and other forms of malware, had essentially two choices if they wanted to learn more about this topic: work in relative isolation or be invited to join a vendor-oriented group.

The vendor-oriented groups (CARO, REVS, VForum, AVPD, and so forth) were designed from the beginning to respond to the need to share information on malware, but membership was usually restricted to those who worked for a software vendor or occasionally to corporate employees and university researchers who were invited to join to share their insights.

TIP

The antivirus industry is as generous with its acronyms as the rest of the computer industry.

CARO is the Computer Antivirus Research Organization, a shadowy group of (mostly Old Guard) anti-virus (AV) researchers.

REVS was a (now defunct) attempt to streamline sample sharing between AV companies.

VForum is a virus/malware researcher mailing list.

AVPD is the Antivirus Product Developer Consortium.

For the vast majority of security specialists working in large organizations on malware defense, there was little hope of entering that circle. There were some in the antivirus industry who defended the exclusivity of the groups as necessary due to trust issues. In the early days of fighting viruses, the level of caution around storage and distribution of viruses was quite high. I still remember all the precautions we took to make sure that viruses did not fall into the wrong hands, and how annoyed many were that people were actually selling viruses on CDs.

To become a member of these vendor-centric groups was quite an achievement, and the growth of their membership numbers was slow.

Meanwhile, a growing number of security specialists in larger organizations had a need to understand the threat their organizations faced. They could read the few books on the subject that were published, try to sift some information out of the noise on Usenet groups like www.alt.comp.virus, take training from vendors, and do their own research. Meanwhile, the complexity of mounting defenses grew constantly, as more and more operating systems and networks were subject to malware attacks.

While I worked selling antivirus software, I would have access to the researchers who dealt directly with the viruses and who discussed in hushed tones the gaping security holes in operating systems, hoping that the authors of malware would not stumble across them. While it certainly felt good to be on the edge of the inner circle and to rub shoulders with the most talented antimalware researchers, there was no shortage of people wanting better access to the information that would make their jobs easier.

VB 2000: A Star is Born

Every year, Virus Bulletin, a UK publication focused on malware and spam issues, organizes a conference in the fall. In 2000, the venue was Orlando, Florida, and I was scheduled to speak about "Anti-Virus Deployment - Doing it Right."

I decided to take my family to Florida, as the VB conference was being held near the amusement parks, and we piled into the car and drove from our home near Ottawa, in Ontario, to Orlando. On the way, I stopped in to see Ken Bechtel, a fellow traveler on the antivirus road. I mentioned the conversations I had been having with some of the security folks at Nortel Networks, mainly John Morris and Peter Sherwood, and with some of my fellow ISSA members. These conversations centered on the need to exchange information on the virus threat and how to best leverage investments in anti-virus defenses.

Ken had also had some ideas along the same lines and we agreed to float the topic at the VB conference.

Cocktails For Two — and More

During the opening cocktail hour, Ken and I started discussing the need for better sources of up-to-date information and resources for dealing with malware threats, as well as the need to stop re-inventing the wheel and to learn from the efforts of other anti-virus specialists.

Soon, a small group of anti-virus specialists from companies such as Nortel, Boeing, and Prudential were gathered and plotting the start of a forum where they could talk openly about their issues concerning AV companies and products and share ideas.

During my presentation on anti-virus solution deployment in enterprise environments, I offered to coordinate the formation of a group of like-minded people to discuss these topics. Inside AVIEN, we fondly refer to this as our "conception." During the remainder of the conference, people pressed their business cards into my hands and I diligently filed them in my pockets.

After the Hangover

A few weeks later, after arriving back home, I contacted these people to confirm that they were really interested in collaborating like this. The response was overwhelmingly in favor of going forward with the plan, and as a result, the world witnessed the formation of a closed, private network. Early on, members agreed that there would be some restrictions on who could belong to AVIEN, so as to ensure that topics discussed were those most important to the majority of members.

One Day at a Time

Since the beginning, members of AVIEN have always been people who look after medium- to large-sized organizations, with at least 1,500 PCs under their care. They have always been employed only by organizations that do not sell anti-virus software, and they have always agreed to abide by a strict code of conduct with regard to confidentiality and mutual respect.

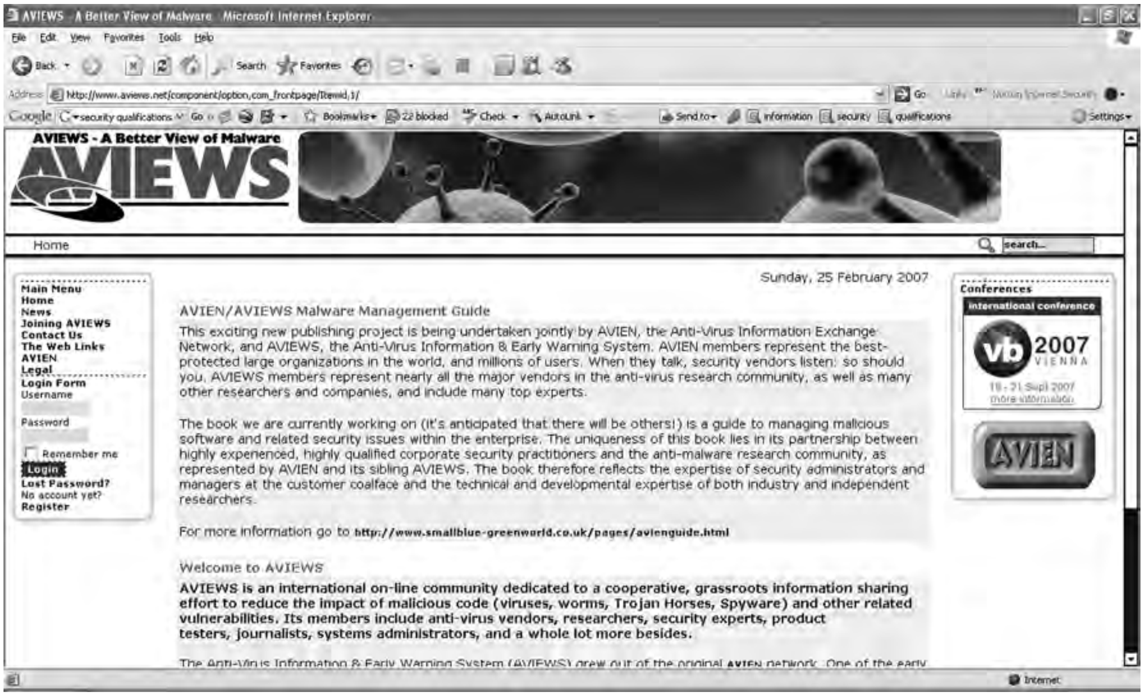
The main activities of AVIEN occurred on several e-mailing lists where discussions focused on deployment of AV software, new viruses that were spotted, and lots of "get-to-know-each-other" conversations. Initially hosted on my e-mail server, it soon became necessary to move the lists to more robust and reliable e-mail systems, especially as warnings about new malware attacks become more common.

The Early Warning System (EWS), created by AVIEN so as to share information between members about new attacks, proved to work very effectively. Not only did it help a number of people save their organizations from major malware attacks, but it was also monitored closely by certain security organizations, which promptly recycled the information in various forms.

Oh No, The Users Are Ganging Up On Us!!!

Almost from the start, the anti-virus vendor community was suspicious of the aims and intent of AVIEN. Many thought that users were looking for any excuse to bad-mouth the vendors and that AVIEN members spent their time talking about them. Even when this was firmly denied, some AV gurus found it hard to accept, or perhaps it was their egos that found it hard to conceive an information exchange network that was doing very nicely without them, thank you.

Figure 1.2 The AVIEWS Web Site



Sunday, 25 February 2007

AVIEN/AVIEWS Malware Management Guide

This exciting new publishing project is being undertaken jointly by AVIEN, the Anti-Virus Information Exchange Network, and AVIEWS, the Anti-Virus Information & Early Warning System. AVIEN members represent the best-protected large organizations in the world, and millions of users. When they talk, security vendors listen: so should you. AVIEWS members represent nearly all the major vendors in the anti-virus research community, as well as many other researchers and companies, and include many top experts.

The book we are currently working on (it's anticipated that there will be others!) is a guide to managing malicious software and related security issues within the enterprise. The uniqueness of this book lies in its partnership between highly experienced, highly qualified corporate security practitioners and the anti-malware research community, as represented by AVIEN and its sibling AVIEWS. The book therefore reflects the expertise of security administrators and managers at the customer coalface and the technical and developmental expertise of both industry and independent researchers.

For more information go to **http://www.smallblue-greenworld.co.uk/pages/avienguide.html**

Welcome to AVIEWS

AVIEWS is an international on-line community dedicated to a cooperative, grassroots information sharing effort to reduce the impact of malicious code (viruses, worms, Trojan Horses, Spyware) and other related vulnerabilities. Its members include anti-virus vendors, researchers, security experts, product testers, journalists, systems administrators, and a whole lot more besides.

The Anti-Virus Information & Early Warning System (AVIEWS) grew out of the original **AVIEN** network. One of the early

In any case, AVIEN initially received a somewhat grudging welcome from the vendor community, and some of them even tried to infiltrate AVIEN to find out what was going on. There was a constant stream of requests for special access to AVIEN from vendors and from those who did not meet the 1,500 PC requirement. In the end, a solution was found by opening the door to a major subset of the mailing lists to anyone willing to pay the annual support fee, which covers the administrative costs of the lists.

AVIEN was thus the catalyst for the formation of AVIEWS, which encompasses not only people in large organizations, but vendors and smaller organizations as well.

Today, the two groups exist in harmony, and all AVIEN members are automatically members of AVIEWS, though not vice versa, and there are rules against misusing information received through the lists for direct marketing purposes.

The Objectives of AVIEN and AVIEWS

Members of AVIEN and AVIEWS (as was shown in Figure 1.2) share common goals, which are to:

- Share information about the anti-virus and malware reality in each organization
- Share information about the techniques used to combat viruses and other malware
- Share information about anti-virus products
- Share information about viruses causing problems
- Participate in an Earl Warning System (EWS)

AVIEN Membership Benefits

AVIEN members receive a number of benefits:

- They discuss with their peers the anti-virus software/hardware issues that concern them
- They receive support in their efforts to implement changes in how defenses are organized
- They receive a subscription to all the AVIEWS services, in addition to AVIEN-specific mailing lists
- They enjoy the warmth of a community of practice, which has developed

Alerts and Advisories

Not only do the organizations send out alerts, but members also inform each other about suspicious incidents before they explode into real alert-type situations. For example, the VBS/Homepage malware was being discussed by AVIEN and AVIEWS members the day before it first made its mark on the world. Nowadays, topics for discussion range far wider, just as the range of problems faced by administrators has widened.

Peer Discussions

Those joining AVIEN and AVIEWS get access to a number of discussion mailing lists, where they can discuss viruses and what's going on in the AV world, including what products are catching or missing which pieces of malware.

Some of the topics members have discussed:

- Characteristics of malicious code as it is discovered, particularly those programs that propagate quickly by exploiting common vulnerabilities.
- Problems/insights on enterprise deployment of the different AV packages with emphasis on pitfalls and timesaving techniques.
- Tweaking the different AV heuristic detection engines to reduce false-positives without impairing/decreasing AV capabilities.
- Lessons learned - Are you seeing a problem with vendor X and what did you do about it?
- Virus countermeasures other than AV scanning software. AVIEN members may not have invented generic filtering, but certainly made a major contribution to refining it.
- Monitoring virus activity within a corporation.
- Techniques for fighting major virus outbreaks.
- Common problems such as the lack of an effective virus naming convention.
- Software distribution methods and issues.
- Verification methods - How do you check to make sure your user base is up to date?
- MS Exchange - What works best on a clustered environment?
- Opinions on the trend towards AV companies providing on-site services.

AVIEN Projects

AVIEN and AVIEWS members have participated in a number of projects apart from this book project (others are planned). The certification project for anti-virus professionals currently resting with Team Anti-Virus is described at the end of this chapter.

In 2006, the first AVIEN Virtual Conference, "Battling Malware: A View from the Trenches," was attended by 156 people across 14 countries and attracted many positive comments. Speakers and presentations included:

- The Fog of War: Informational Challenges to Malware Defense and Incident Response (Gaby Dowling).
- Spy-Where? (Mary Landesman)
- Diagnostic Tools - The Next Stage (John Alexander)

- Criminalization of Code (Ken Dunham)
- Mobile Threats (Mikko Hypponen)
- Weapons of Bot Destruction: Conventional and Non-conventional Tactics to Defend a Network Against an Evolving Threat (John Morris and Eric Kedrosky)

The 2007 conference on "The New Face of Malware – Stories from the Battlefield" was also very successful, and included the following speakers and presentations:

- Rootkits: No Longer Just a *nix Problem (Martin Overton)
- The Common Malware Enumeration (CME) Initiative (Desiree Beck)
- Collaborative Response to Targeted Attacks (Matt Ziemnaik)
- Hackers' Favorite Hiding Places - Initial Places to Look for rootkits, Trojans and Other Malware (Paul Schmehl)

Other public initiatives have included a public "call to action" on the spyware threat and a petition about the dangers of teaching the writing of viruses as a tool for learning about anti-virus defenses.

Less public projects have included a virus encyclopedia project, surveys, a repository of useful tools, informational resources, and so on.

Anti-virus Vendor Image

"Nobody likes me, everybody hates me, going out into the garden to eat worms." Not to mention viruses, and Trojans, and quite a few other types of malware. And, in the Blue Corner (as they used to say in the wrestling broadcasts), there's anti-virus software. Nearly everyone uses it, nearly everyone resents having to use it, and few people understand it as well as they want you to think they do.

This section considers the tensions between vendors and customers, AV vendors and other security vendors and agencies, the anti-virus community in its broadest sense and the rest of the world. We then try to sort out the truth behind some of the myths and half-truths, and look at the real competencies of a little-understood and little-appreciated community venture.

AVIEN & AVIEWS: Independents and Vendors in Anti-Malware Research

I've no wish to overstress points already made by Robert Vibert, or made in the next section by James Wolfe, but the closely related organizations, AVIEN and AVIEWS, represent an unusual type of partnership between two not entirely disparate groups:

- Enterprises that use security software (especially for the management of malicious code)
- Vendors who supply those products and services.

NOTE

Why are they not entirely disparate? For one thing, because there are a number of groups and individuals that don't exactly fit into either of these main groups, as James Wolfe tells us later in this chapter. Not all reputable anti-virus researchers are aligned to security vendors (or to major enterprises). In fact, it's increasingly common for researchers outside the anti-virus industry to "cross the floor" to the opposite side of the House of AV/Customer Representatives, and sometimes back again. Sometimes, whole organizations may stop being just customers and become security vendors in their own right. Microsoft is perhaps the most obvious example: while out-and-out security products are a small part of its product range, the company is, nevertheless, a major player in the security industry. For this and other reasons, it gets harder to separate the anti-virus industry from the rest of us.

There is actually a potential tension between *all* consumers and suppliers. The consumer fears that the supplier may charge or overcharge for a product that may be unnecessary, inadequate, or both. (There is an exception of sorts: luxury items for which the main selling point is that they are "reassuringly expensive." Even here, though, there is the potential fear that the item may be less exclusive than the buyer expects.) The vendor, meanwhile, will have a number of fears relating to loss of revenue: fraudulent purchases, piracy copyright/intellectual property rights (IPR) issues, post-sales support issues, and so on.

It would be naive to suggest here that these issues don't ever affect the relationships between the organizations represented in AVIEN who buy security software, and the security vendors who are represented (among other groups) in AVIEWS. Indeed, the fact that security vendors are not eligible for representation in AVIEN gives the lie to such a suggestion. Sometimes, people want to talk about AV issues without worrying about offending or being overheard by vendors. To some members, though, one of the most valuable benefits of AVIEWS membership is the opportunity for end-users who are adept at using the software in real-world environments, to exchange views and information with those who are adept at creating and maintaining the software.

The main practical distinction between the two (apart from the more stringent membership criteria for AVIEN members) is that AVIEN members have access to a few resources that are not available to non-members as well as to all AVIEWS resources. These distinctions are mostly made to protect the membership from inappropriate and/or inadvertent intrusion from commercial bodies, and reflect the parent organization's origins as a special interest group and information exchange network for customers rather than for vendors.

Furthermore, these concerns were reflected by initial suspicions within the AV vendor community that AVIEN was not only anti-vendor but also engaged in the free exchange of malware samples. (See my article "Setting the Record Straight," published in Virus Bulletin in November 2001, for some thoughts on these earlier misconceptions.) In fact, several mechanisms exist for AVIEN and AVIEWS members to submit samples of malicious software to vendors:

- Through AVIEN members who have set up mailing mechanisms accordingly that members can use to send samples to several vendors at once
- Through sites like VirusTotal that test samples against multiple scanners and may submit them to vendors
- Via the addresses supplied by individual vendors for that purpose (see the chapter on DIY malware analysis for some of these).
- Via off-list communications.

However, the rules under which both organizations operate forbid the direct solicitation or exchange of malware samples on-list. This is as much in order to avoid administrative and security risks associated with the exchange of improperly secured samples, as it is in order to avoid criticism by non-participant vendors.

However, the organization has changed, and perceptions have changed even more. The development of AVIEN's sister organization as a common meeting ground for vendors and customers allayed many mutual suspicions. In fact, some vendors have been initially dismayed on joining AVIEWS that there is no direct sample exchange between members and vendors, and no sample repository, as maintained by some researcher lists. However, they seem to have been happy enough to stay and talk to some of their most knowledgeable customers. Furthermore, the relationship between the two groups was not as clear-cut and demarcated as I've represented it here. These vendor representatives were not generally marketroids, but well-respected, ethically scrupulous hands-on researchers, more than happy to cooperate with customers and representatives of other companies with the common aim to reduce the impact of malicious code in all sectors. The customer representatives were not just hands-off Information Technology (IT) managers, but seasoned security administrators, many of whom were highly respected in their own right as researchers into and writers about security and anti-virus issues.

NOTE

In fact, it would be wrong to give the impression that these groups had never met on equal terms before. Many of the active contributors to AVIEN and AVIEWS mailing lists were also trading opinions and information on lists associated with the WildList Organization, the European Institute for Computer Anti-virus Research (EICAR), university-hosted anti-virus research groups, and so on.

The innovation in this case was the fact that the lists were dominated and controlled by administrators at the coalface, rather than vendor-employed researchers or academics, resulting in a degree of informed discussion and mutual respect rarely found in organizations that interface between customer and supplier.

James Wolfe discusses independent, enterprise and vendor researcher roles and relationships in the anti-malware community in the "AV Wannabe" topic section later in this chapter.

Favorite Myths

Perhaps the most prevalent speech defect in the world is an inability to say "I don't know..." when asked a question to which the answer is outside your competence and knowledge. Thus pop singers and soccer players are able to masquerade as experts on politics, media stars whose main talent is for reading an autocue or for the recycling of press releases become "instant experts" on computer security, and real security experts whose expertise is in other areas spread misconceptions about malicious software. This readiness to open mouth without engaging brain is sometimes referred to as "ultracrepidarianism."

NOTE

The term "ultracrepidarian" comes from the Latin *ultra crepidam* ("beyond the sole of the shoe"), and is drawn from a story relating to the ancient Greek painter Apelles. It's said that a cobbler criticized his representation of a human figure in a painting. He accepted criticism from a cobbler about the way he had portrayed the figure's foot and slipper. However, he dismissed the criticism of the rest of the figure's leg as beyond the cobbler's specialist expertise.

Rob Rosenberger, who first drew my attention to the term (if not the psychological phenomenon) suggests that 'Most people who claim to speak with authority about computer viruses have little or no genuine expertise. Some virus experts describe it as "False Authority Syndrome"– the person feels competent to discuss viruses because of his or her job title, or because of his expertise in another computer field, or simply because he knows how to use a computer.' (www.vmyths.com/fas/fas1.cfm.htm)

Tools & Traps

The Dunning-Kruger Effect

In the age of the blogger and the Web-syndicated article where readers are encouraged to add comments on the original article to the same Web page, I note that people whose only claim to expertise in security is that they use, or decline to use, anti-virus software, are nonetheless eager to pronounce on the competence or otherwise of the industry.

This calls to mind a phenomenon called the Dunning-Kruger effect, which refers to a study in which they tested a number of hypotheses, including 1) that incompetent people "compared with their more competent peers, will dramatically overestimate their ability and performance relative to objective criteria," and 2) that they also fail to recognize genuine skill in others – "Incompetent individuals will suffer from metacognitive skills, in that they will be less able than their more competent peers to recognize competence when they see it – be it their own or anyone else's."

The study concludes that "those with limited knowledge in a domain suffer a dual burden: Not only do they reach mistaken conclusions and make regrettable errors, but their incompetence robs them of the ability to realize it."

The study is documented in "Unskilled and Unaware of It: How Difficulties in Recognizing One's Own Incompetence Lead to Inflated Self-Assessments," by Justin Kruger and David Dunning, Cornell University: it can be found in the Journal of Personality and Social Psychology, vol. 77, no 6, p 1121-1134 (1999), or at http://72.14.209.104/search?q=cache:vlX_V9SqScUJ:www.apa.org/journals/features/psp7761121.pdf

I don't, of course, claim that it's impossible for anyone but the anti-virus industry to rate the competence of people and products within that industry, or to make lucid and valid comments on issues related to malicious software. However, it's clear that readiness to comment publicly is not, in itself, proof of knowledge. The Internet's status as the world's most fertile source of misinformation and urban legend could constitute a major book in its own right. For the moment, though, I'll restrict myself to a few of the most relevant myths about the anti-virus industry.

"Anti-virus Only Catches Known Viruses"

It's not only its customers who have a "down" on the anti-virus industry. Colleagues in the security industry have frequently made observations that, while taken to be authoritative on account of the source, are surprisingly poorly founded.

In August 2006, Alan Paller of SANS commented in a SANS newsletter, with reference to a poorly implemented comparative anti-virus product test by Consumer Reports for which "new viruses" were created, with the intention of testing heuristic capabilities, that "This controversy is especially problematic for the leading AV companies because they have traditionally not done well in finding and blocking new viruses quickly. But for goodness sakes, if they don't do well at finding and blocking new viruses, why are we buying them?" (www.sans.org/newsletters/newsbites/newsbites.php?vol=8&issue=65&rss=Y - sID320)

The flaws in the methodology used in this test have been explored elsewhere (See "AV Testing SANS Virus Creation" in Virus Bulletin, October 2006), Some of the reasons why it isn't necessary to create viruses in order to test heuristic detection are discussed further in the chapter on anti-malware testing and evaluation. However, this particular quotation includes a couple of widespread fallacies.

"Traditionally," finding and blocking new viruses is something that anti-virus companies used to be quite good at, even (or especially) before the widespread use of heuristic scanning. New viruses and variants appeared comparatively rarely and spread comparatively slowly. It was not unusual for a virus writer to supply anti-virus researchers with a copy of their latest creations even before one of their customers caught and sent one in. As long as you weren't one of the first victims, you stood a good chance of not sustaining any damage when a virus got to you, even though it might be weeks or even months before the next batch of updates (signatures, if you must) got to you. As new virus types appeared that propagated more quickly using the Internet, so did faster ways of disseminating updates. As the speed at which viruses and variants appeared accelerated, so did the rate at which they could be analyzed.

What Paller may have meant was that anti-virus isn't good at detecting viruses that are so new that they haven't yet been analyzed, so they haven't been added to definitions databases. It's true that the "traditional" model of "find a virus, analyze it, write a definition to detect it, make the definition available to customers," is by definition unable to provide adequate anti-virus defense in a modern high-risk environment. However, this isn't what modern antimalware technology does: in fact, it hasn't been for many years. Almost from the beginning, simple "signature" scanning has been bolstered with less specific technologies such as behavior monitoring, integrity checking, and generic filtering, and more recently by rule-based heuristic analysis and scoring, as well as generic signatures (all of these technologies are discussed at greater length in the chapter on "Defense in Depth," in the section on generic defenses).

Heuristic and other generic technologies don't, it's true, offer anything like 100 percent detection of all viruses (let alone all other malware), but there's nothing traditional about that, either.

Are You Owned?

How Good is Anti-virus at Detecting Unknown Malware

One of the conclusions of the AusCERT 2006 Computer Crime and Security Survey (http://www.auscert.org.au/images/ACCSS2006.pdf) is that 60% of "malware developed for the purposes of stealing personal information and account credentials" (i.e., banking Trojans, backdoors, and so forth used in phishing attacks) "are not detectable by anti-virus software at the time they are discovered in the wild."

The survey therefore concludes that computers kept updated with the very latest "anti-virus software signatures" are "likely to be vulnerable to such attacks about 60 percent of the time." This is actually slightly misleading. It doesn't take into account the impact of other elements of a multilayered defensive infrastructure (such as generic filtering of executable downloads and specific filtering of malicious URLs) on the potential problem, or even consider the effects of good educational practice. Still, the essential point is the difficulty of getting anywhere near 100 percent detection of malware.

The survey goes on to state, more-or-less correctly, that attackers "work to increase the effectiveness of their attacks by modifying Trojan malware to create new variants that are unlikely to be detected by most up to date anti-virus software upon release." I'd comment only that Trojans and other malware are not necessarily considered to be "variants" if the code remains the same but is obfuscated by using different runtime packers (see Chapter 9, "DIY Malware Analysis" for more discussion of the use of packers for obfuscation).

However, it's also worth noting that while 60 percent detection might be a reasonable estimate for phishing Trojans and similar, it isn't necessarily accurate for other kinds of malware. As Andrew Lee and myself have pointed out in a paper on "Heuristic Analysis – Detecting Unknown Viruses" that should, by the time you read this, be available from www.eset.com/download/whitepapers.php, replicative malware is conceptually easier to detect than non-replicative malware. This is because some types of code strongly represent a self-replicative functionality, whereas other types of "malicious" behavior can be much more difficult to infer from automated analysis.

The sad fact is heuristic detection was arguably more effective in controlling malware in the mid-90s than it is now, even though the detection technology is much more advanced than it was then. The reasons for this are far more complex than simple incompetence (the anti-virus community includes some of the most able researchers and programmers in *any* field), and are considered elsewhere in this book. They do, however, reflect the increased complexity of the 21st Century threatscape (See Chapter 2, "Stalkers on your Desktop") and the range of threat types that now has to be addressed by security vendors in this space.

For example, the decline in the proportion of viruses to other forms of malware, as malware authors found that the ego-gratification of writing a virus that spreads fast and wide doesn't fit well into a "malware-for-profit" business model.

Paller also suggested that "They should stop complaining and instead thank Jeff Fox and the editors at Consumer Reports for helping to do important product improvement research for them." I won't labor the deficiencies of the testing methodology used in that particular test here (though we have *much* more to say about testing in due course!), but it is naive to assume that the anti-virus companies are not aware of the issues of partial protection described above. Unfortunately, neither the AV industry nor its many critics have managed to come up with an alternative that is both viable and widely accepted and that comes near to detecting 100 percent of all malicious code with no risk of false positives.

"Vendors Protect Their Own Revenue Stream, Not Their Customers"

Welcome to the world of commercial reality. There are some very nice people working for security vendors, but most of them have to make a living, and the subscription model favored by most anti-virus vendors does have an obvious commercial advantage over a once-and-for-all fix. But is there such a thing as a one-time fix that fully replaces the functionality of a continuously updated AV scanner? Well, there are solutions that will prevent (nearly) all occurrences of given classes of malware.

These are mostly generic; that is, they work by blocking all *potentially* dangerous objects rather than by discriminating between specific objects on the grounds that they are known to be dangerous or harmless. The obvious disadvantage to this approach is that harmless objects may be valuable or necessary to the enterprise, and must be either sacrificed or validated in some way as an exception.

However, the need to allow exceptions may be in itself a potential vulnerability. Exact identification of a specific threat, however, doesn't have these issues. A known virus (virus-specific) scanner might incorrectly identify an object as infective or malicious (a false positive), but this actually happens quite rarely. There are other approaches based less on technical solutions (generic or specific) than on policy enforcement. Nick FitzGerald, former editor of Virus Bulletin and a security expert and anti-virus researcher of formidable reputation in his own right, has referred to a concept he calls integrity management (*http://archives.neohapsis.com/archives/fulldisclosure/2005-02/0033.html*).

This is essentially the application of a general principle of security (sometimes called "deny by default") to virus management. The classic virus scanner allows all code to run unless it's known to be malicious (essentially a form of blacklisting), but integrity-management-oriented software allows only whitelisted applications to run. The integrity management equivalent to the definitions database in a conventional anti-virus application doesn't contain definitions (or signatures) for malicious applications. Instead, it defines exceptions to the "deny all" rule.

Applying this principle to preventing the execution of malicious code is not new. The classic "integrity checker" or "checksummer" is a variation on the "Integrity Shell" component of Fred Cohen's Integrity Toolkit (*http://all.net/books/integ/japan.html*), whereby an application is "vetted" before it's executed. Some "real time" scanners have combined a variety of generic approaches (heuristic analysis, behavior analysis, even integrity checking), though usually to complement virus-specific detection rather than replace it.

Tools & Traps

Updates and Patches

In practice, there are probably no commercial one-time fixes that don't require ongoing maintenance and development to take into account changes in the operating environment such as system and application patches, new vulnerabilities and exploits, and so forth: thus, a comparatively generic application such as a personal firewall may be updated on a monthly or more frequent basis. Indeed, the personal firewall on the laptop I'm using to type this paragraph is updated by the vendor more often than the first commercial anti-virus package I used, which was updated every three months (those were the days!).

We can draw a distinction between definitions updates (or signatures) that detect specific threats, and patches that update and/or extend the functionality of a basic package. However, the distinction is less crucial when it applies to a package whose primary function is security.

More to the point, the revenue streams generated by the subscription model are clearly not dependent on the definitions or signatures database model.

The problem, then, is not necessarily that there are no alternatives, or even that the alternatives have not been available within anti-virus product ranges. Rather, it's the fact that the alternatives have not received the customer acceptance that would persuade the vendors to focus on them. There seems to be an idea that there is a 100 percent effective solution out there somewhere, which would eliminate the virus problem at a stroke, but that the anti-virus industry is concealing it in order to protect their own revenue stream. It makes for an engaging conspiracy theory, but if such a solution did exist, it would probably be highly generic, and most customers would hate it because of the negative impact it would have on business processes.

"Vendors Only Know About and Detect Viruses"

This isn't exactly so. Anti-virus vendors detect a great many other things, including all sorts of worms, bots, Trojans, germs, droppers, bacteria, garbage files, and test files. In fact, AV products probably detect a greater overall range of threats than any other specialized security service than I can think of, but the sad fact is this is really about the understandable but unrealistic wish to have one application fix every conceivable security problem.

"They Write All the Viruses"

No one believes that doctors create or spread diseases, or that crime is an invention of law enforcement agencies, but the world believes that AV companies are a force for evil in the same way that Hollywood believes the NSA or CIA to be. Since this curious notion keeps coming up, there are a few points that need to be raised

- AV companies don't normally employ virus writers, if only to avoid encouraging other virus writers:
  1. Being able to write a virus doesn't prove anything about your being capable of developing or maintaining an anti-virus or other security product.
  2. Anti-virus developers are expected to be trustworthy. Willingness to write viral code is not trusted.
  3. No developer with two neurons to rub together will run the risk of allowing another vendor's marketing department to garner competitive advantage.
- Many researchers don't believe it's necessary to write viral code, even for research purposes, and go to extreme lengths to avoid doing so.
- There is quite enough malware being produced already without adding to it gratuitously. What's more, there are some astonishingly talented coders working for AV companies. If they did write the malware, the general standard would probably be noticeably higher.

In an article for Virus Bulletin (November 2006) I did suggest, quite seriously, that some people would actually be reassured if they thought that AV companies did write viruses, on the grounds that people who write viruses must be well equipped to defend against them. However, this latter presumption doesn't hold much water. In general, the ex-hackers in the security industry—whom the AV sector tends to avoid like the plague—are those who managed to get caught, which doesn't necessarily inspire confidence. What's more, there's a great deal more to defending against malware than creating software.

"Anti-virus Should Be a Free Service: After All, There Are Free Services That Do a Better Job"

Commercial AV is sometimes seen as suspicious if not unethical, because people are paid to maintain it. At the same time, well-intended but not necessarily effective or well-supported freeware is seen as not only "nobler" but also somehow more efficient. It's true that there have been excellent non-commercial anti-virus programs that have filled a serious gap in the market for a while, but they've either been generic (e.g., Padgett Peterson's "Macrolist" and "Disksecure") or in a niche market. John Norstad's "Disinfectant" for the Mac was a good example of the latter. Even during its heyday, though, Norstad never claimed that it detected all varieties of Mac malware, and he was pretty much forced to retire the program when it became clear that people were assuming that it was a complete solution, even when the mid-1990s epidemic of macro viruses made that expectation unsustainable for a freeware product.

There may still be room for freeware, open source, and so on in many environments. However, there are governance, contractual, and support issues in enterprise malware defense that such products generally do little to address.

AV Wannabe

The term virus researcher has long been sullied by its association with hobbyist virus writers claiming to be engaged in research. Nevertheless, independent anti-virus researchers have played an important part in the fight against malicious software, and many individuals within AVIEN have contributed to that struggle while working outside the anti-virus industry. James Wolfe, an independent anti-virus researcher with many years experience in enterprise security, asks, "So you want to be a bona fide computer anti-virus researcher?," and considers the roles of and the relationship between the different types of researchers within and outside the anti-virus industry, including:

- Anti-virus Company Analysts
- Independent Researchers
- Technical and Psychological Analysts
- Corporate Anti-virus and Security Administrators and Analysts

So You Want to Be a *Bona Fide* Computer Anti-Malware Researcher?

For some time, people have asked what is meant by a *bona fide* computer anti-virus (or, more often nowadays, antimalware) researcher. As someone who has been a "hobbyist" and independent anti-virus researcher for more than 14 years, I have waited with much enthusiasm for someone to come up with an authoritative definition. Within the AV and antimalware communities, there has long been discussion about who can truly be considered an antimalware researcher. Is it the code monkey (a programmer, especially a low-level, hands-on coder) who spends countless hours ripping through pages of source code to find out what a virus is supposed to do? Is it the person in the AV company Research and Development (R&D) lab who develops the updates to security products to fight the latest malware? Or is it perhaps the person who looks at trends in malware authoring and profiles virus writers, in order to help predict what will happen next? How about the person sitting in a large corporation who perhaps does most or all of the above in their efforts to protect their company? (Nah, everybody knows that corporate types don't know what they want or need, right?) While the types and range of threats faced by the anti-malware researcher have changed and broadened immensely in recent years, there is no less need for specialist skills and very specific personal qualities.

In the Beginning...

In the beginning there were a few individuals who pretty much defined the field of anti-virus research. Visionaries like Frederick Cohen, Alan Solomon, Klaus Brunnstein, Fridrik Skulason, and Vesselin Bontchev were the cornerstone and, in a sense, the founding fathers of our industry. Research groups began to form. Chief among them was the Computer Anti-virus Research Organization (CARO) (www.caro.org). Companies were springing up that were developing products to fight the virus problem. These products were largely based on concepts derived from the work of these early players in the industry. To ensure that the products were doing the job that they were supposed to, Joe Wells' WildList was born. By establishing a baseline source of authenticated samples of viruses known to be "In the Wild," accurate tracking of malware epidemiology and trends became realistic options, as did valid performance testing of anti-virus software. (See Chapter 10 on "Testing and Evaluation," by Andrew Lee and David Harley). Now that we could analyze the impact of the threat and could fight it (at least at a basic reactive and technical level), we became concerned with who was developing these threats and why, so other individuals began looking into the identities and attributes of the people who wrote malicious code.

In recent times we've come into an age of blended threats. These threats target exploits within computer software to insert their viral and/or other malicious programs. As these threats become more common, we've seen a new type of individual come on to the AV scene, the corporate anti-virus specialist. Because of the potential for catastrophic data loss, corporations have become increasingly concerned with the threat of malicious code. Most feel that it is essential for a business to employ personnel who understand threat and counter-threat technologies and can liaise with the company's chosen vendor(s) to ensure that the impact of such an event is lessened.

The purpose of this section is to define these four different types of individuals—malware analysts, product developers, technical and psychological research specialists, and corporate malware management specialists—and then look at which characteristics are really important for a virus researcher to have. Note that since it is difficult to apply the "scientific method" to attributes that are largely subjective, this chapter is largely the opinion of this author, based on observations of those individuals whose current job function deals almost entirely with computer viruses.

Anti-virus Company Analysts

AV company analysts have, in a sense, one of the most important yet least recognized jobs in our industry. Very often, they sit in their employers' laboratories and reverse-engineer viruses day after day, with little recognition coming their way. Their work allows the product developers to add the needed definitions (search strings or, popularly but not altogether accurately, signatures) to keep their virus scanning products current. These folks seem to move around a lot, and if they are particularly sharp they don't spend too long analyzing code at this level, but usually end up in more exciting technical areas like R&D. Nonetheless, this form of analysis is very important, though not particularly glamorous, and not necessarily well paid. It allows the individual to get a good foundation in the inner workings of what makes the malicious code work. Many of the big names in our industry still keep their skills honed by reverse engineering and analyzing source code.

Independent Researchers

Independent researchers are not tied to any particular company, though they may work with or for security vendors on some sort of ad hoc basis. They often perform many of the same tasks as the individuals who work for the AV companies. The primary difference is that they don't do it because they have to, but because they have a passion for the security field and enjoy the work. Padgett Peterson is probably one of the most respected individuals to fall into this category. He has written many programs that deal with viruses, including a generic AV program and Macrolist, an excellent tool that addressed the problem of macro viruses when they first became a significant problem. Other notables in this category include Richard Ford, Eddy Willems, and Nick FitzGerald.

Technical and Psychological Analysts

As we began to understand the virus threat and develop solutions to counter it, we tried to figure out what type of individual writes viruses. Sarah Gordon became the most widely recognized expert in profiling virus writers as a result of her analyses of the technological and psychological factors that influence virus writing. For many years she has interviewed virus writers via e-mail and telephone. No one can deny that valuable insights and information that have been provided to us because of Sarah's efforts. As a direct result of her research and that of those who followed her, we learned a great deal about the types of person who wrote viruses, as well as the victims. These insights in to the Virus Exchange (vx) community have helped us to be a little more proactive in an otherwise reactive industry. However, as the balance has shifted away from the hobbyist virus writer to "for profit" malware authors (this shift is also reflected in the decline in virus numbers proportionally to other forms of malicious software), our perception of *who's who* and *who does what* in malware authoring and dissemination has also changed.

Corporate Anti-virus Specialist

One of the least recognized factors in the anti-virus research equation is that there are people in corporate enterprises and the public sector who fight viruses and other malware on a daily basis. Some of these people know the AV vendors products inside out, but they also understand how to reverse engineer viruses and analyze code. These individuals also must directly interact with end-users and management, if they are to be effective in their job. Not all of these individuals are true researchers, but many of them are as knowledgeable as specialists within AV companies and make just as important a contribution. While there are many individuals jointly responsible for a single product or one specialist area within a security company, corporate security specialists are characteristically responsible for many areas and report to all levels of management. A bad decision on their part could cost their employers millions of dollars in lost productivity, data loss, and lost income.

There are two major sources of frustration for these individuals. If they are doing their job successfully, the organization runs smoothly, at least in terms of freedom from the effects of malicious code or over-intrusive security software. Thus, the best result their management can hope for is to hear nothing but silence from these specialists, because there are no problems to report. Accordingly, they often don't get a great deal of recognition and respect, and the importance of their work can be overlooked entirely, sometimes to the point where the necessity of their role is called into question. Furthermore, other researchers and many of the AV companies often don't seem to think that someone who works for anyone other than an anti-virus company can know what they need or want, though it's not unknown for talented individuals like Andrew Lee to move from a customer organization into the anti-virus industry.

What is a Researcher?

The problem with formulating an exact definition of an anti-virus researcher is one of personal perception and subjectivity. Occasionally, a pseudo-scientific method is employed, but this isn't always appropriate for two reasons:

1. Assigning static, pseudo-objective attributes to a largely subjective set of characteristics does not qualify as a true scientific definition.
2. The individual is attempting to justify themselves as a researcher when they have no concept as to what the application of true scientific method even is. (See: Frank Wolfs, "Introduction to the Scientific Method," http://teacher.nsrl.rochester.edu/phy_labs/AppendixE/AppendixE.html)

The real measure of a virus researcher isn't where he or she works; it's a combination of what they know, what they do, and, at least as importantly, who they are.

Researcher Skill-Set

In this section I will discuss the basic knowledge skills that individuals who work in the industry have (on the basis of personal discussions and observation.)

- A good understanding of computers is necessary (see Sarah Gordon's article on "What is an anti-virus researcher?" – www.badguys.org/researchers.html). This applies not only to Operating Systems and Commercial Off-the-Shelf (COTS) packages, but also to the hardware involved.
- Knowledge of some programming languages is essential if an individual is required to examine source code. In such a case, knowledge of assembly language has, traditionally, been perceived as being of particular importance (SpaceIT.uk, Job Opportunities, www.spaceit.uk.com/jobrefs/security.html).
- Networking and related technologies are almost indispensable. While there maybe many experts within an organization that work in this area, it is still important to understand the implications if the malicious code being examined were to get into a network. A strong grasp of these technologies is also necessary if the researcher is planning on setting up a testing lab (Vivianne Fisher: "Looking inside an anti-virus lab." www.zdnet.com.au/newstech/security/story/0,2000024985,20267231,00.html).
- As in any area of scientific research, the anti-virus researcher needs to understand how to investigate, observe, and report.

As a virus researcher becomes more experienced, he or she will begin to understand what the malware authors were trying to accomplish with the program. Does this mean that they are different sides of the same coin? Not really, because although the knowledge is there, the ethical maturity of the virus researcher would prevent them from crossing the ethical line, and thus turning into a virus writer (See Sarah Gordon's paper for the Virus Bulletin 2000 conference on "Virus Writers: The End of Innocence" (www.research.ibm.com/antivirus/SciPapers/VB2000SG.html).

What Makes a Researcher?

In this section, I explore the characteristic traits of an anti-malware researcher. Some of these traits are more suited to the corporate specialist but are nonetheless commonly held to be valid.

- **Extreme Tolerance for Stress** Malware research is characterized by hours of mind numbing boredom followed by moments of sheer terror. This is particularly true for the corporate specialist, who may be expected to anticipate what protection is needed for a 100,000 or more computers. (AVIEN members have included individuals responsible for over three million computers!) They also must perform their jobs with a minimum of staff, internal support, and funding.

- **A Passion for Obscure Knowledge** Malware research, and especially virus research, by its very nature is a somewhat esoteric scientific field. Researchers have to look for minute details in reverse-engineered code. An overlooked snippet of code could prove to be critical when deciding on what course of action is necessary to deal with a new malicious program.
- **Thinking Outside of the Box** In some respects, virus research can be nothing more than blue-sky/daydreaming. It is an unwritten responsibility of the anti-malware community to dream up the worst things that could be done to a computer system ("nightmare scenarios") and find ways to prevent them.
- **Extreme Caution** This is critical when dealing with malware. It makes sure that a researcher routinely checks for any potential mistakes in their handling of viruses, whether it concerns what the code they are reviewing actually does, where the samples are stored, or if the testing facility or machines are secured.
- **Trustworthy** The researcher has a responsibility to do the right thing. We are tasked with handling dangerous material. Although this material isn't life threatening in the same sense as Semtex or Ebola, it can nevertheless cause damage to a range of systems (from financial systems to medical systems), which have a serious potential impact on the general populace. Thus, a computer virus that causes real damage can affect many aspects of society, so any virus samples obtained by an individual must be handled as carefully as their biological equivalents would be.

In The End

In the past, virus samples were only distributed amongst a tight group of individuals within the antivirus community who explicitly trusted one another, and to some extent the organizations for which those individuals worked. For a time this was a (largely) effective way of controlling access to "dangerous" software samples. However, due to the availability of viruses and other malicious code on the Internet, and the growing exchange of samples between security companies and organizations outside the tightly knit antivirus community, some of the "old school" arguments about trust have become moot, or at least shifted ground.

Much of this section has examined different characteristics in the non-scientific manner that seems somewhat prevalent in the industry. So, who is a virus researcher? Is it the person in the labs of an AV Company, the independent guy, the person who analyzes the technical and psychological factors in virus writing, or the corporate AV manager? It could be any of those types. As I see it there are three things that we must do before we can establish a universal definition of what constitutes a researcher in this field:

1. Most formal disciplines have an evaluation process to test an individual's mastery of the subject matter through academic qualifications, job history and function, or formal testing. We, however do not, and need to develop a universally accepted process.

2. We must develop a universal code of conduct that holds all researchers accountable for their actions.
3. Finally, the AV community *as a whole* must recognize and accept the above means of evaluating an individual's qualification.

Until we adopt a universally recognized evaluation method, there will be no real definition of the term "bona fide computer antivirus (or antimalware) researcher." The closest definition that I can offer at present is this: an antivirus researcher is a person who has the necessary technical knowledge to study computer viruses and can demonstrate that they understand the moral and ethical responsibility to do no harm, while freely sharing their knowledge with others, in order to prevent the proliferation of computer viruses in a safe and responsible manner.

You Should Be Certified

In this section, David Harley looks at some major certifications for individuals working within the information security field and considers their relevance to antimalware research, then looks at antivirus qualifications run by or for specific security vendors, and finally looks at an initiative for an antivirus-specific security certification scheme on which a number of AVIEN members have worked. Is there a place for a vendor-independent AV qualification?

(ISC)^2

The International Information Systems Security Certification Consortium (see Figure 1.3), known to most of the security world as (ISC)^2, maintains the Common Body of Knowledge (CBK), which they describe as a "compendium of industry best practices" in the relevent domains as descibed below.

(ISC)^2 is a not-for-profit organization, and is heavily reliant on its membership (all paid up credential holders in "good standing" are members) for services such as proctoring examinations, compiling examination questions, and so on. You can find out about (ISC)^2 and the certifications it supports at www.isc2.org, as shown in Figure 1.3. In return, apart from a universally recognized qualification, it offers excellent peer networking opportunities. Members are expected to comply with the (ISC)^2 code of ethics (https://www.isc2.org/cgi-bin/content.cgi?category=12), demonstrate competence in the appropriate domains of the CBK (usually by passing a comprehensive multi-choice examination), demonstrate that they are keeping up their skills and knowledge by registering Continuing Professional Education (CPE) credits, have worked for the required length of time in IT in general and security in particular, and so on.

This knowledgebase is the foundation on which the organization's certifications for industry professionals are founded. These certifications include:

- Associate of (ISC)^2 Designation (https://www.isc2.org/cgi-bin/content.cgi?page=824) This is an interim designation for people who have passed the relevant

Figure 1.3 (ISC)2

(ISC)2 - Microsoft Internet Explorer
https://www.isc2.org/cgi-bin/index.cgi
(ISC)² SECURITY TRANSCENDS TECHNOLOGY
MEMBER LOG IN
Member ID Password
GO
Forgot your Password or Member ID?
Log In Help
Education Examinations Credential Offerings Employers News & Information Member Benefits
Support for Every Step of Your Information Security Career
SecureLongIsland March 15, 2007
(ISC)² Leadership Seminar
SSCP CAP CISSP Concentrations
Why Certify About (ISC)² FAQs (ISC)² Events Contact (ISC)²
(ISC)² ALERTS
Stop & shop at (ISC)² STORE
(ISC)² Japan
(ISC)²® Board Elects Randy Sanovic as Chairman

examination but don't meet all the criteria for certification, usually, by not having worked long enough in the field.)

- Certification and Accreditation Professional (CAP) (www.isc2.org/cgi-bin/content.cgi?page=820)
- Certified Information Systems Security Professional (CISSP) (www.isc2.org/cgi-bin/content.cgi?page=818)
- CISSP Concentrations (www.isc2.org/cgi-bin/content.cgi?page=819) — These are additional certifications for individuals who choose to attempt a qualification in a specific specialty, i.e.:
    1. Information Systems Security Architecture Professional (ISSAP®)
    2. Information Systems Security Engineering Professional (ISSEP®)
    3. Information Systems Security Management Professional (ISSMP®)
- Systems Security Certified Professional (SSCP) (www.isc2.org/cgi-bin/content.cgi?page=817)

CISSP and SSCP are registered trademarks of (ISC)^2 as is CBK. The CISSP award is by far the best known of these certifications, and is probably the main "competitor" to GIAC qualifications (which we'll look at shortly) in the fairly narrow field of security qualifications most wanted by prospective employers.

SSCP

The SSCP credential is intended for people working as (or hoping to work as) senior security engineers and administrators, and is intended to demonstrate competence in the following seven domains of the SSCP CBK:

- Access Control
- Administration
- Auditing and Monitoring
- Cryptography
- Data Communications
- Malicious Code/Malware
- Risk, Response, and Recovery

Tools & Traps

Pre-Certification Experience

This is a summary of areas of experience required for SSCP certification, but might actually constitute a reasonable minimum baseline for a security administrator job description:

- A first degree or equivalent
- Knowledge of a body of knowledge such as the CBK
- Experience of managing or supervising projects and/or other staff members
- Experience of exercising judgment, decision-making, and discretion
- The exercise of ethical judgment, not just ethical behavior
- Communication skills
- The ability to train and mentor others
- Research and development skills
- The ability to specify controls and mechanisms, as opposed to just operating them

CISSP

The CISSP award is intended for more senior levels of management (CISOs, Senior Security Engineers, and so on). Suitability is assessed by testing on the following ten domains from the CBK:

- Access Control
- Application Security
- Business Continuity and Disaster Recovery Planning
- Cryptography
- Information Security and Risk Management
- Legal, Regulations, Compliance and Investigations
- Operations Security
- Physical (Environmental) Security
- Security Architecture and Design
- Telecommunications and Network Security

NOTE

I hadn't really thought about certification for most of my career in security, until I was offered the opportunity to do a CISSP CBK "boot camp" course. Because I'd specialized in malware research for so long, I was actually slightly nervous of trying for a qualification that was "wide rather than deep" and aimed at managers working in areas in which I didn't consider myself expert. But I hadn't had a general "refresher" in some years, so I took the (highly concentrated) course, and while I left it with my brain jellified, I did discover that I knew more than I'd realized. Even so, when I finally got around to taking the exam, it turned out to be really, really tough. I was so sure I'd failed miserably. that when I got e-mail informing I'd passed, I checked the message headers in minute detail, to see if it was spoofed. Many people have told me their experience was similar, although no one else has admitted to checking for an e-mail scam.

CISSP Concentrations

These are three information security-related certifications offered to CISSPs in good standing, by passing an additional exam, demonstrating that a CISSP possesses more in-depth knowledge in that area than is expected of the holder of the generic CISSP qualification.

ISSAP

The major domains of the ISSAP are these:

- Access Control Systems and Methodologies
- Cryptography
- Requirements Analysis and Security Standards, Guidelines, and Criteria
- Technology Related Business Continuity Planning (BCP) and Disaster Recover Planning (DRP)
- Telecommunications and Network Security

ISSEP

The major domains for the ISSEP concentration are as follows:

- Certification and Accreditation
- Systems Security Engineering
- Technical Management
- U.S. Government Regulations for Information Assurance

ISSMP

The major domains for the ISSMP concentration are as follows:

- Enterprise Security Management Practices
- Enterprise-wide System Development Security
- Law, Investigations, Forensics and Ethics
- Overseeing Compliance of Operations Security
- Understanding BCP, DRP, and Continuity of Operations (COOP)

CAP

The Certification and Accreditation Professional (CAP^{CM}) award is a means by which Information Assurance (IA) professionals required to follow to National Institute of Standards and Technology (NIST) guidelines can demonstrate competence and skill in certification and accreditation. The domains in which they are tested include:

- The Purpose of Certification
- Initiating System Authorization
- The Certification Phase

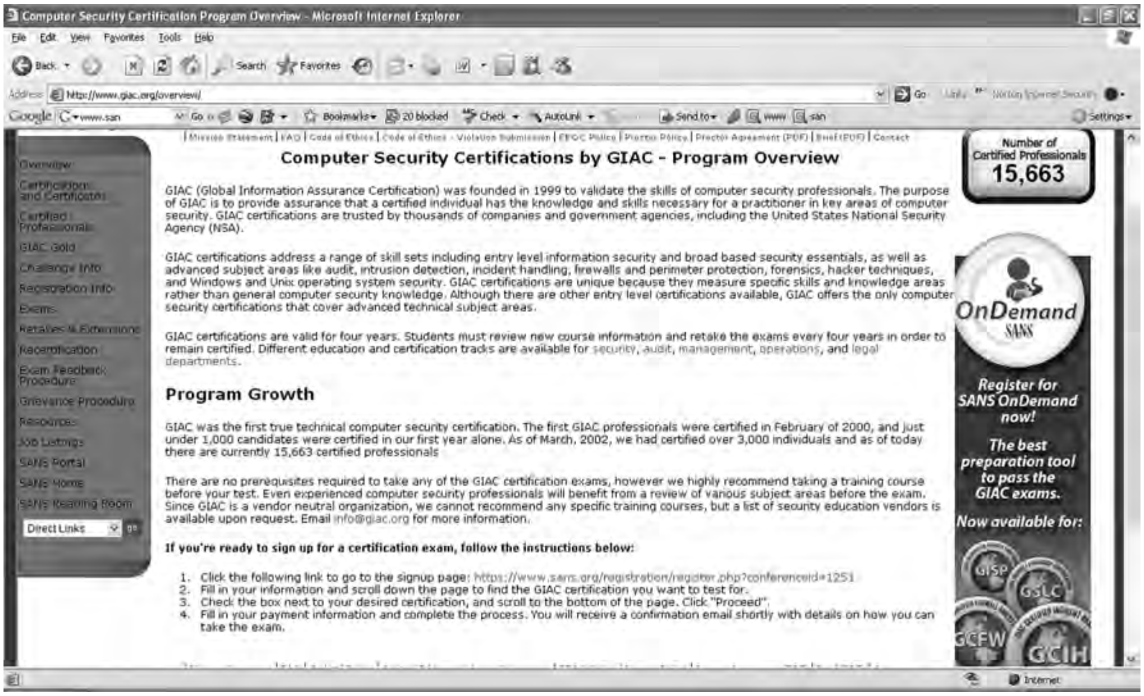
- The Accreditation Phase
- Continuous Monitoring

It's not discussed further here, but included for completeness.

SANS GIAC/GSM Certifications

SANS certifications offer a range from fairly general introductory courses to highly specific, hands-on courses in a particular area of security. (SANS does offer training for other credentials such as CISSP, but these are not considered here.) While (ISC)^2 certification is demanding, it ranges across the whole security range and are fairly abstract. Global Information Assurance Certification (GIAC) awards (see Figure 1.4) tend to be less conceptual and more practical, focusing on specific areas of information security application and maintenance. Part of the explanation for this divergence of approaches may be that CISSP holders not only have to have passed the exam, but must have been working as security professionals for several years before they can be awarded the accreditation, so substantial practical experience is assumed.

Figure 1.4 The GIAC Security Certifications Overview Page


Computer Security Certification Program Overview - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.giac.org/overview/
Go
Google www.san Go Bookmarks 20 blocked Check AutoLink Send to www san Settings
Overview
Certifications and Certificates
Certified Professionals
GIAC Gold
Challenge Info
Registration Info
Exams
Retakes & Extensions
Recertification
Exam Feedback Procedure
Grievance Procedure
Resources
Job Listings
SANS Portal
SANS Home
SANS Reading Room
Direct Links go
Mission Statement | FAQ | Code of Ethics | Code of Ethics - Violation Submission | EPOC Policy | Proctor Policy | Proctor Agreement (PDF) | Brief (PDF) | Contact
Number of Certified Professionals 15,663
OnDemand SANS
Register for SANS OnDemand now!
The best preparation tool to pass the GIAC exams.
Now available for:
GISP
GSLC
GCFW
GCIH
Internet

Computer Security Certifications by GIAC - Program Overview

GIAC (Global Information Assurance Certification) was founded in 1999 to validate the skills of computer security professionals. The purpose of GIAC is to provide assurance that a certified individual has the knowledge and skills necessary for a practitioner in key areas of computer security. GIAC certifications are trusted by thousands of companies and government agencies, including the United States National Security Agency (NSA).

GIAC certifications address a range of skill sets including entry level information security and broad based security essentials, as well as advanced subject areas like audit, intrusion detection, incident handling, firewalls and perimeter protection, forensics, hacker techniques, and Windows and Unix operating system security. GIAC certifications are unique because they measure specific skills and knowledge areas rather than general computer security knowledge. Although there are other entry level certifications available, GIAC offers the only computer security certifications that cover advanced technical subject areas.

GIAC certifications are valid for four years. Students must review new course information and retake the exams every four years in order to remain certified. Different education and certification tracks are available for security, audit, management, operations, and legal departments.

Program Growth

GIAC was the first true technical computer security certification. The first GIAC professionals were certified in February of 2000, and just under 1,000 candidates were certified in our first year alone. As of March, 2002, we had certified over 3,000 individuals and as of today there are currently 15,663 certified professionals

There are no prerequisites required to take any of the GIAC certification exams, however we highly recommend taking a training course before your test. Even experienced computer security professionals will benefit from a review of various subject areas before the exam. Since GIAC is a vendor neutral organization, we cannot recommend any specific training courses, but a list of security education vendors is available upon request. Email info@giac.org for more information.

If you're ready to sign up for a certification exam, follow the instructions below:

1. Click the following link to go to the signup page: https://www.sans.org/registration/register.php?conferenceid=1251
2. Fill in your information and scroll down the page to find the GIAC certification you want to test for.
3. Check the box next to your desired certification, and scroll to the bottom of the page. Click "Proceed".
4. Fill in your payment information and complete the process. You will receive a confirmation email shortly with details on how you can take the exam.

Like $(ISC)^2$, GIAC certified professionals are expected to conform to the parent organization's Code of Ethics. GIAC certifications are valid for four years, and are renewed by retaking the exams, whereas CISSP is valid for three, and retaking is only required if the holder doesn't earn sufficient CPE credits to prove that they're keeping up with developments in the field.

GIAC certifications include both broad-based and entry-level security knowledge testing, and advanced subject areas such as:

- Audit
- Incident handling
- Intrusion detection
- Firewalls
- Forensics
- Operating system security

For this reason, the number of security certifications offered is very large, and a candidate can follow a number of education and certification tracks within the areas of security and audit (www.giac.org/certifications/), though each GIAC certification is designed to stand on its own. Certification is at two levels. Silver certification requires the candidate to pass exams: having done this, he or she can apply for Gold certification, for which they must submit a technical paper. Certification is offered in conjunction with a full SANS training course.

NOTE

SANS Institute, GIAC, and the SANS Technology Institute are separate, though related, institutions. The SANS Institute's primary mission is to encourage original research using a "community-oriented consensus approach" and present it as educational material, using a number of delivery methods (conferences, mentoring, self study, and so on.)

GIAC's aim is to assess the student's mastery of the material and ability to apply it within a specific "skill knowledge domain."

The SANS Technology Institute aims to produce managers with security and audit technical ability, and to this end offers a Masters Degree in Information Security (www.sans.edu/).

We can't realistically offer a comprehensive list and description of SANS certifications here (see www.sans.org/training/courses.php), but we can list some to give you a general flavor of what is on offer:

Audit Certifications

- GIAC Security Audit Essentials (GSAE)
- GIAC Certified ISO-17799 Specialist (G7799)
- GIAC Systems and Network Auditor (GSNA)
- GIAC Auditing Wireless Networks - Certificate (GAWN-C)
- GIAC Payment Card Industry (GPCI)

Management

- GIAC Information Security Professional (GISP)
- GIAC Security Leadership Certification (GSLC)
- GIAC Certified Security Consultant (GCSC)

Security Administration

- GIAC Information Security Fundamentals (GISF)
- GIAC Security Essentials Certification (GSEC)
- GIAC Certified Firewall Analyst (GCFW)
- GIAC Certified Intrusion Analyst (GCIA)
- GIAC Certified Incident Handler (GCIH)
- GIAC Certified Windows Security Administrator (GCWN)
- GIAC Certified UNIX Security Administrator (GCUX)
- GIAC Certified Forensics Analyst (GCFA)
- GIAC Securing Oracle Certification (GSOC)
- GIAC Secure Internet Presence (GSIP)
- GIAC .Net (GNET)
- GIAC Assessing Wireless Networks (GAWN)

Legal

- GIAC Contracting for Data Security (GCDS)
- GIAC Law of Fraud (GLFR)
- GIAC Business Law and Computer Security (GBLC)
- GIAC Legal Issues in Information Technologies (GLIT)

Management

- GIAC E-warfare (GEWF)
- GIAC Fundamentals of Information Security Policy (GFSP)
- GIAC Critical Infrastructure Protection (GCIP)
- GIAC HIPAA Security Implementation (GHSC)
- Ethics in IT (GEIT)
- GIAC Leadership (GLDR)
- GIAC Security Policy and Awareness (GSPA)

Security Administration

- Stay Sharp Program - Computer and Network Security Awareness (SSP-CNSA)
- Securing Solaris – The Gold Standard (GGSC-0200)
- Securing Windows 2000 – The Gold Standard (GGSC-0100)
- Auditing Cisco Routers – The Gold Standard (GGSC-0400)
- Stay Sharp Program - Defeating Rogue Access Points (SSP-DRAP)
- Stay Sharp Program - Mastering Packet Analysis (SSP-MPA)
- GIAC Intrusion Prevention (GIPS)
- GIAC Cutting Edge Hacking Techniques (GHTQ)
- GIAC Web Application Security (GWAS)
- Stay Sharp Program - Google Hacking and Defense (SSP-GHD)
- GIAC Reverse Engineering Malware (GREM)

Other Certifications and Qualifications

The Infosec Qualifications page at www.itgovernance.co.uk/page.infosec_qualifications summarizes a huge range of qualifications, including some of the major international qualifications summarized here and including pointers to further information.
Other certifications covered range from ISACA's to British Computer Society (BCS)/Information Systems Examination Board (ISEB) certifications, to BSI certifications like ISO 27001:2005 Lead Auditor, to Certified Ethical Hacker (CEH) and CompTIA Security+.

The Institute of Information Security Professionals (IISP) (www.instisp.org/) has been heralded in some quarters as a "replacement" for more "specific" qualifications. In fact, it's more appropriate to look at this as a way of extending assessment of professionalism beyond largely technical criteria. While it's not yet completely clear where the IISP is going, it may be useful as a career-assisting supplement to technical qualifications, at least in the UK. Whether it will meet its own aspirations as an international group has yet to be determined.

Vendor-Dependent Training

Just as antivirus products vary enormously in their range and functionality according to vendor, so does the training that is available from AV vendors. We cannot cover all of the product training available from all vendors in the AV space. Indeed, we can't do justice even to the few vendors we consider here. On no account should you think of this as a comprehensive guide to what's available at the time of writing, let alone by the time this book reaches your hands. However, this brief overview of some of the training available from three major vendors will give you some idea of the sort of areas they currently cover.

McAfee

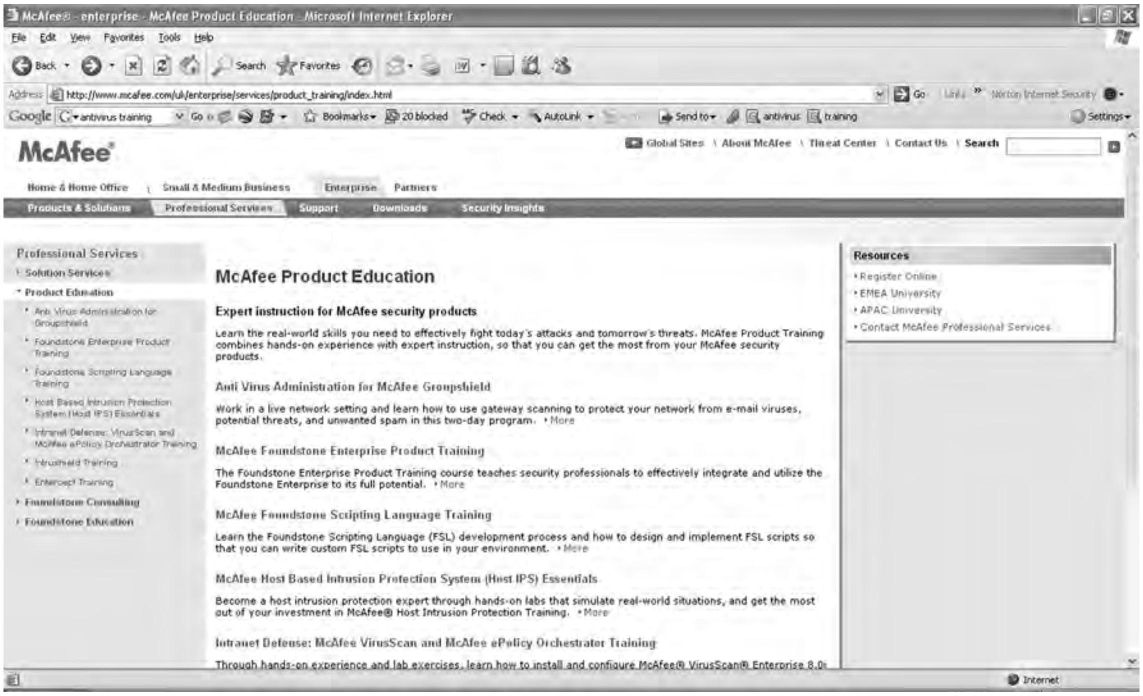
The McAfee course "Anti Virus Administration for McAfee Groupshield" is a two-day course intended for administrators implementing gateway solutions like GroupShield and SpamKiller. Issues discussed include:

- The impact on the network of viruses found on the mail server and gateway
- Appreciating the importance of defense in depth (multi-layering; see Chapter 6)
- Installation and feature sets for GroupShield in Microsoft Exchange environments
- Configuring GroupShield policies for antivirus and content scanning
- Installation and configuration of SpamKiller for Exchange
- Configuring ePolicy Orchestrator for GroupShield and SpamKiller management
- Troubleshooting Groupshield

Other training available from McAfee, as indicated by Figure 1.5, at time of writing included:

- Foundstone Enterprise Product Training
- Foundstone Scripting Language Training
- McAfee Host based IPS Essentials
- Intranet Defense: McAfee VirusScan and McAfee ePolicy Orchestrator Training
- McAfee Intrushield Training
- McAfee Entercept Training

Figure 1.5 McAfee Product Education Page


McAfee® - enterprise - McAfee Product Education - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.mcafee.com/uk/enterprise/services/product_training/index.html
Go Links Norton Internet Security
Google antivirus training Go Bookmarks 20 blocked Check AutoLink Send to antivirus training Settings
McAfee
Global Sites | About McAfee | Threat Center | Contact Us | Search
Home & Home Office | Small & Medium Business Enterprise Partners
Products & Solutions Professional Services Support Downloads Security Insights
Professional Services
Solution Services
Product Education
Anti Virus Administration for Groupshield
Foundstone Enterprise Product Training
Foundstone Scripting Language Training
Host Based Intrusion Protection System (Host IPS) Essentials
Intranet Defense: VirusScan and McAfee ePolicy Orchestrator Training
Virusheild Training
Entercept Training
Foundstone Consulting
Foundstone Education
Resources
Register Online
EMEA University
APAC University
Contact McAfee Professional Services
Internet

McAfee Product Education

Expert instruction for McAfee security products

Learn the real-world skills you need to effectively fight today's attacks and tomorrow's threats. McAfee Product Training combines hands-on experience with expert instruction, so that you can get the most from your McAfee security products.

Anti Virus Administration for McAfee Groupshield

Work in a live network setting and learn how to use gateway scanning to protect your network from e-mail viruses, potential threats, and unwanted spam in this two-day program. • More

McAfee Foundstone Enterprise Product Training

The Foundstone Enterprise Product Training course teaches security professionals to effectively integrate and utilize the Foundstone Enterprise to its full potential. • More

McAfee Foundstone Scripting Language Training

Learn the Foundstone Scripting Language (FSL) development process and how to design and implement FSL scripts so that you can write custom FSL scripts to use in your environment. • More

McAfee Host Based Intrusion Protection System (Host IPS) Essentials

Become a host intrusion protection expert through hands-on labs that simulate real-world situations, and get the most out of your investment in McAfee® Host Intrusion Protection Training. • More

Intranet Defense: McAfee VirusScan and McAfee ePolicy Orchestrator Training

Through hands-on experience and lab exercises, learn how to install and configure McAfee® VirusScan® Enterprise 8.0i

Sophos

In the UK, Sophos offers product/vendor-independent antivirus workshops combining a hands-on approach to exploring viruses in a controlled environment with practical infection containment methods. The longstanding Malware Analysis course, as seen in Figure 1.6, is continuously refined, but at the time of writing consisted of:

- An introductory section defining viruses, worms, and Trojan horses
- A survey of virus types, replication mechanisms, and virus side effects
- A consideration of viral mutation, virus writers, and the future
- A consideration of AV policy that covers corporate implementation, AV measures and procedures, and incident recovery
- A consideration of virus mechanisms with specific reference to Windows, Microsoft Office, and network issues
- A section on related issues such as hoaxes, spam, and adware

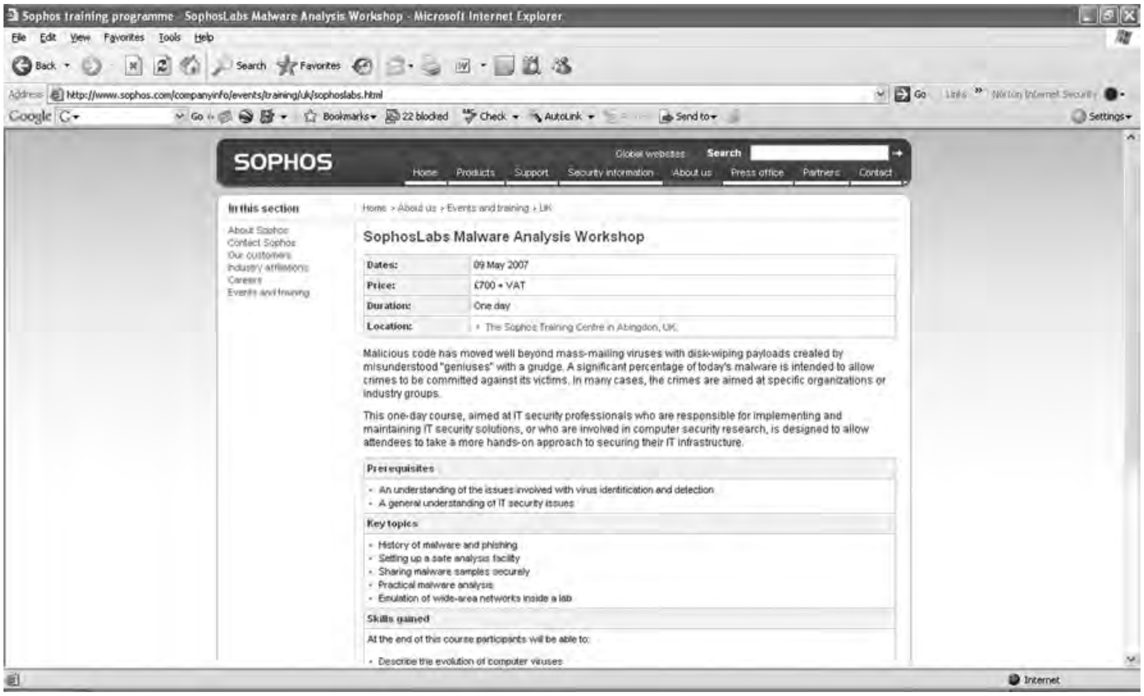
This is far more like the product-independent training and testing discussed in AVIEN and Team Anti-Virus, as described in the final section of this chapter, though experience with workshops like this in the past suggests that there are considerable practical difficulties in running a completely product-independent session.

Sophos also offers a product-specific "Best Practice" course, which is more like the courses by McAfee and Symantec we also look at in this overview chapter. A summary of the current version of "Best Practice for Sophos Anti-Virus" is given here, as it provides a useful contrast to the product-independent summary above.

The course is essentially focused on the administration tools provided with Sophos Anti-virus for deployment and administration in the enterprise. It includes:

- An overview of the core AV suite, plus the enterprise management tools and gateway products
- A thorough consideration of the Enterprise Manager Library
- Deployment via the Enterprise Console, remote installation over a local area network (LAN) or wide area network (WAN), and other deployment methods
- Updating — scheduled updates, auto-updating, emergency updates, and bandwidth issues (I don't have personal experience of this course, but would expect it to

Figure 1.6 Sophos Malware Analysis Workshop



SophosLabs Malware Analysis Workshop

Dates:	09 May 2007
Price:	£700 + VAT
Duration:	One day
Location:	The Sophos Training Centre in Abingdon, UK

Malicious code has moved well beyond mass-mailing viruses with disk-wiping payloads created by misunderstood "geniuses" with a grudge. A significant percentage of today's malware is intended to allow crimes to be committed against its victims. In many cases, the crimes are aimed at specific organizations or industry groups.

This one-day course, aimed at IT security professionals who are responsible for implementing and maintaining IT security solutions, or who are involved in computer security research, is designed to allow attendees to take a more hands-on approach to securing their IT infrastructure.

Prerequisites

- An understanding of the issues involved with virus identification and detection
- A general understanding of IT security issues

Key topics

- History of malware and phishing
- Setting up a safe analysis facility
- Sharing malware samples securely
- Practical malware analysis
- Emulation of wide-area networks inside a lab

Skills gained

At the end of this course participants will be able to:

- Describe the evolution of computer viruses

Sophos training programme - SophosLabs Malware Analysis Workshop - Microsoft Internet Explorer
http://www.sophos.com/companyinfo/events/training/uk/sophoslabs.html
SOPHOS
Home
Products
Support
Security information
About us
Press office
Partners
Contact
In this section
About Sophos
Contact Sophos
Our customers
Industry affiliations
Careers
Events and training
Home > About us > Events and training > UK

address resource-usage mitigation techniques such as staging servers, staggered updates, and so on)

- Maintenance issues such as configuration
- Virus detection and management hands-on sessions

Finally, we look at the Sophos Malware Analysis Workshop, as shown in Figure 1.6, which assumes a knowledge of virus detection and general security and is aimed specifically at bona fide IT security professionals, looking at issues such as:

- History of malware and phishing — Clearly, this is a step beyond the comparatively and conceptually simple virus workshops of the 1990s
- Setting up a safe lab environment for testing and analyzing malware
- Secure distribution and sharing of malware samples
- Practical analysis of malware
- Emulating a complex network environment inside a lab

(See Chapter 9 on "DIY Malware Analysis" for our own take on these issues.)

Symantec

By contrast, Binary Research International's "Symantec Antivirus Corporate Edition Training," as described at www.savtraining.com, is a two-day workshop that uses hands-on sessions to work towards the Symantec Certified Specialist/Small Business Security exams. It covers:

- Planning, deployment, and management of Symantec Antivirus (irritatingly, both Sophos and Symantec sometimes refer to their products as SAV)
- Configuration
- Incident response
- Enterprise management utilities

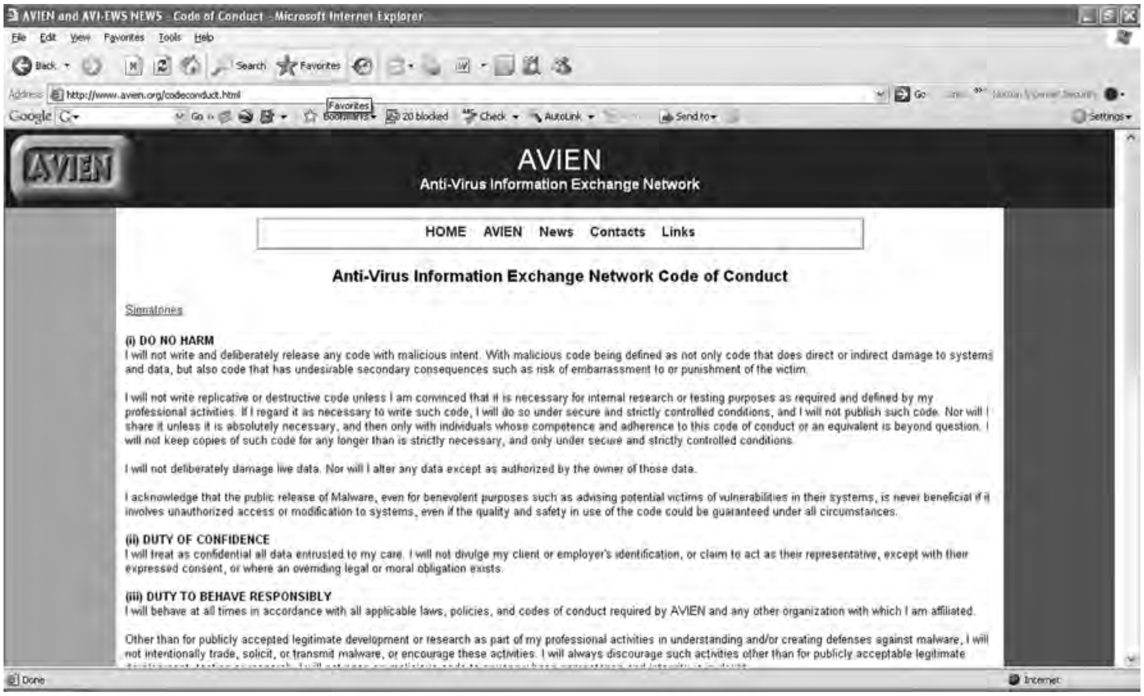
Symantec's own Virtual Academy offers a "state-of-the-art online learning environment" using Webcasts, hands-on lab sessions, on-demand teaching modules, and mentoring (www.symantec.com/en/uk/enterprise/training/virtual_academy/index.jsp). A current course on "Symantec AntiVirus 10.x" takes three days and covers the implementation and management of a Symantec anti-virus network, update management, enforcement of virus/spyware/adware scanning across the enterprise, and managing remote users. This is described as the "first step towards mastering the configuration and management of an antivirus network." Unfortunately, a more detailed description of the course wasn't available on the Web site at this time.

Should There Be a Vendor-independent Malware Specialist Certification?

Fairly soon after AVIEN was first founded, as Ken Bechtel noted in his 2003 paper for the Virus Bulletin conference on "Anti-Virus Support, the Need for Maturing a Career Field," many members were keen to see developed or contribute actively to a certification project for anti-virus specialists.

Adherence to a code of conduct like the AVIEN Code of Conduct in Figure 1.7 was seen as one of the requirements of such certification. Indeed, this is also the case for more generalist security certifications like CISSP, for example, and while it's not uncommon for tradesmen to be required to conform with some form of code of practice, strict conformance with an ethical code is often seen as a sine qua non for admission into one of the professions.

Figure 1.7 AVIEN Code of Conduct


AVIEN and AVI-EWS NEWS - Code of Conduct - Microsoft Internet Explorer
http://www.avien.org/codeconduct.html
AVIEN
AVIEN
Anti-Virus Information Exchange Network
HOME AVIEN News Contacts Links
Done
Internet

Anti-Virus Information Exchange Network Code of Conduct

Signatories

(i) DO NO HARM
I will not write and deliberately release any code with malicious intent. With malicious code being defined as not only code that does direct or indirect damage to systems and data, but also code that has undesirable secondary consequences such as risk of embarrassment to or punishment of the victim.

I will not write replicative or destructive code unless I am convinced that it is necessary for internal research or testing purposes as required and defined by my professional activities. If I regard it as necessary to write such code, I will do so under secure and strictly controlled conditions, and I will not publish such code. Nor will I share it unless it is absolutely necessary, and then only with individuals whose competence and adherence to this code of conduct or an equivalent is beyond question. I will not keep copies of such code for any longer than is strictly necessary, and only under secure and strictly controlled conditions.

I will not deliberately damage live data. Nor will I alter any data except as authorized by the owner of those data.

I acknowledge that the public release of Malware, even for benevolent purposes such as advising potential victims of vulnerabilities in their systems, is never beneficial if it involves unauthorized access or modification to systems, even if the quality and safety in use of the code could be guaranteed under all circumstances.

(ii) DUTY OF CONFIDENCE
I will treat as confidential all data entrusted to my care. I will not divulge my client or employer's identification, or claim to act as their representative, except with their expressed consent, or where an overriding legal or moral obligation exists.

(iii) DUTY TO BEHAVE RESPONSIBLY
I will behave at all times in accordance with all applicable laws, policies, and codes of conduct required by AVIEN and any other organization with which I am affiliated.

Other than for publicly accepted legitimate development or research as part of my professional activities in understanding and/or creating defenses against malware, I will not intentionally trade, solicit, or transmit malware, or encourage these activities. I will always discourage such activities other than for publicly acceptable legitimate

Unfortunately, while many AVIEN members signed up to the Code, it met with less enthusiasm in the wider antivirus community, where many researchers in the industry felt it to be too restrictive.

Levels of Certification and Associated Knowledge Bases

The AVIEN Anti-Virus Specialist Certification Project was intended to establish acceptable criteria for independently certified antivirus professionals, with endorsement from AV vendors and corporate management, through the establishment of training and testing programs. To this end, the following levels of certification and knowledge were proposed, and a collection of test material gathered (reproduced and adapted here courtesy of Ken Bechtel and Team Anti-Virus.)

Certified Anti-Virus Administrator (CAVA)

This was envisaged as being the entry-level certification for the individual charged with managing an organization's AV defenses.

Hands-on Dealing with Viruses

- Recognize virus symptoms
- Isolate virus
- Replicate virus
- Submit for evaluation
- Clean boot, file, macro, and script virus infections

Managing an AV system

- Install anti-virus product server and workstation
- Optimize AV to conform with best practices
- Distribution of updates
- Trouble shoot AV installation server and workstation
- React to Hoaxes
- React to legitimate virus warning

Developing an AV Policy and Strategy

This category is based on an understanding of policy implementation and best practices.

Certified Anti-virus Specialist (CAVS)

A CAVS would have the same basic skill set as an administrator, but would be expected to have built on those skills in order to take more responsibility for organizational security.

Hands-on Dealing with Viruses

- Recognize virus symptoms
- Isolate virus
- Replicate virus
- Submit for evaluation
- Clean boot, file, macro, and script virus infections

Managing an AV System

- Install anti-virus product server and workstation
- Optimize AV for best practices
- Distribution of Updates
- Trouble shoot AV installation server and workstation
- React to Hoaxes
- React to legitimate virus warning
- Determine best anti-virus product for environment
- Manage deployment and support at all levels (gateway, mail, server, desktop, Mobile users).
- Advanced software troubleshooting.

Developing an AV Policy and Strategy

- Having an understanding of policy implementation and best practices
- Write a policy to protect the company
- Design a multi-layered defensive strategy
- Create a response plan.

Certified Enterprise Anti-virus Architect (CEAVA)

The role of a corporate security architect is often seen as requiring high-level executive skills, focused on policy and organizational infrastructure rather than hands-on technical skills. The requirement that an anti-virus architect should have the same technical skills as a lower-level specialist makes perfect sense in the antivirus community, where we're accustomed to having to work around the poorly founded assumptions of how malware and anti-malware technologies work that are so prevalent among high-level managers and directors from altogether different backgrounds. How well this approach would be accepted by major enterprises, even those enlightened enough to join AVIEN, is as yet unproven.

Hands-on Dealing with Viruses

- Recognize virus symptoms
- Isolate virus
- Replicate virus
- Submit for evaluation
- Clean boot, file, macro, and script virus infections
- Disassemble script and macro viruses, identify payload

Managing an AV System

- Install anti-virus product server and workstation
- Optimize AV for best practices
- Distribution of updates
- Trouble shoot AV installation server and workstation
- React to hoaxes
- React to legitimate virus warning
- Determine best anti-virus product for environment
- Manage deployment and support at all levels (gateway, mail, server, desktop, mobile users)
- Advanced software troubleshooting

Developing an AV Policy and Strategy

- Having an understanding of policy implementation and best practices
- Write a policy to protect the company
- Design a multi-layered defensive strategy
- Create a reaction plan
- Know the laws concerning computer viruses

Auditing the AV deployment

- Develop a plan to audit AV deployment in the company
- Audit deployment and compliance

Updating the Certifications

A possible, even necessary addition to this list of certifications would have been Anti-Virus Instructor (CAVI), which would have required the additional knowledge necessary to develop and teach a Live Virus Workshop. Hands-on virus lab training sessions were seen as a necessary training tool for all three certification levels.

Even if it had been practical to run such sessions without input from vendor-based researchers, to do so would have compromised the likelihood of acceptance of these standards across and by established AV companies, researchers, certification organizations, and major corporations. Hard decisions would also have been necessary on how to implement training and testing (e.g., certified third parties, in-house, on-line, and so on.)

It was also intended to establish standards for "grandfathering," so that exceptionally qualified or experienced candidates would not necessarily have to undergo testing more appropriate to "newbie" AV professional aspirants.

Clearly, these guidelines could do with some expansion and updating. In 2007, the threat landscape is very different to how it was earlier in the decade. The emphasis on viral malware in these specifications doesn't reflect the importance of non-replicative malware nowadays, and most people who work in this area now have to be very aware of a whole range of other security specialties. An AV architect might be required to know not only about laws that specifically address malware, but those that have more oblique implications for malware management, such as data protection legislation, Sarbanes-Oxley (SOX), Basel II, and so on. Not to mention quasi-legal imperatives and standards such as ISO/IEC 17799 (and its siblings and descendants).

Nonetheless, the need to address the points Ken Bechtel made in his 2003 paper still exists. The need for expansion of these certifications to include virus researchers and laboratories has been reinforced by the recurrent use and misuse of poor testing methodologies by media-feted comparative tests. Despite the CME initiative, the naming of viruses and other malware remains fragmented and confusing. Employers continue to advertise for staff to manage malware issues with no idea of what qualifications or experience to ask for. The case for a CERT-like overseeing authority in which the vendor and independent research communities are strongly represented is as convincing as it ever was.

Summary

The partnership and free exchange of information and opinions on AVIEN and AVIEWS mailing lists between the AV industry, its biggest customers, independent researchers and other interested parties, reflects a growing realization that:

- There's a lot more to malware management than creating and maintaining defensive software.
- The malware scene in the 21st century is about far more than viruses, and the motivation behind malware creation goes far beyond mischief or bragging rights. It's part of a whole continuum of criminal activity that goes beyond the mechanics of replication, to the exploitation of human weaknesses as a means of making illicit profits.
- Worms and viruses are actually not that important out on the Wild and Woolly Internet. The money is in keyloggers, backdoors, spam generation, e-mail fraud, and so on. Replicative malware is, more often than not, just a means to that end.
- Long ago, the AV industry worked pretty much on its own. Apart from a handful of talented specialists outside the industry, the research community consisted of people working on technical solutions to the technical issues of virus propagation, and the precise form of those solutions was only indirectly affected by the customers who either bought your approach, or bought someone else's approach.

Nowadays, technical solutions are not enough, because so many of the current pains in our assets are far less susceptible to proactive detection. No single group has all the answers, and some problems are better addressed by some of the public and not-so-public coalitions between different types of security vendor, security organizations, other service providers, special interest groups, law enforcement agencies, educationalists, and so on.

Perhaps there is a glorious day on the horizon where the forces of good will triumph over the criminals and vandals. I wish... In the meantime, many current issues derive from the fact that one of the vital ingredients missing from malware management is expectation management. Most of the online world still imagines that there is a magic bullet around somewhere, and that it's only the greed of the vendors and the incompetence of administrators that keeps them from finding it. Well, maybe. In the meantime, we'll move on to consider the problems in more detail, and maybe turn your mind to some non-magical, but easier-to-obtain countermeasures.

Solutions Fast Track

History of AVIEN and AVIEWS

☑ Historically, antivirus and security specialists outside the industry, including those who worked for client organizations in an administrative or managerial role, had little contact with the rest of the research community unless they were invited to join one of the fairly secretive, invitation-only, vendor-oriented lists.

☑ At the Virus Bulletin Conference in 2000, a group of researchers and specialists from major AV client organizations discussed the need for better resources for exchanging information on products, best practices, upcoming attacks, and so on. Robert Vibert offered to coordinate the foundation of a group to discuss these issues. AVIEN began its life as a forum for lively e-mail discussions and an early warning system.

☑ The antivirus industry was, in the beginning, suspicious of the new organization. However, many people on both sides of the customer/vendor divide were well aware of the advantages to both parties of two-way information sharing. AVIEWS, sister organization to AVIEN, was founded in order to facilitate that information sharing. Vendors (and in principle anyone else not meeting the membership criteria for AVIEN) who join AVIEWS can subscribe to nearly all the same mailing lists as full AVIEN members, to the benefit of all.

☑ Excellent examples of what these two groups can achieve when they join forces include the AVIEN virtual conferences and this book.

Anti-virus Vendor Image

☑ The anti-virus industry suffers from a longstanding image problem. It's seen as inept, exploitative, secretive, and patronizing towards those outside the charmed circle. It's even accused of being largely responsible for the problem it's supposed to fix. Some of these accusations are based on common misconceptions (e.g., misunderstanding of grassroots detection technology) and confusion as to the exact benefits and drawbacks of malware-specific detection versus generic approaches to malware management.

☑ Detection technology, especially heuristic analysis, has improved vastly over recent years, but the proportion of malicious software detected heuristically has probably declined. This is essentially a technical issue. It's unrealistic to anticipate anywhere near 100 percent detection without the use of supplementary filtering. Such generic filtering elevates the risk of false positives substantially. In the past, corporate customers have tended to shy away from products that do this. While everyone grumbles about the AV subscription update model, they tend to buy packages that are (usually) accurate at detecting known malware rather than products that catch a high percentage of unknown malware but entail extra work in terms of filtering false positives.

- ☑ Freeware has a long and sometimes honorable role in the history of malware management (and some major AV vendors still don't charge home users for the use of some of their products.) However, when it comes to countering malware in the enterprise, you get pretty much what you pay for.

So You Want to Be a Bona Fide Computer Antimalware Researcher?

- ☑ The term "researcher" is often misapplied and vaguely defined, but is discussed in this chapter with reference to AV vendor-aligned analysts, independent researchers, technical/psychological analysts, and corporate specialists.
- ☑ Independent and corporate researchers may do very similar jobs to AV researchers, with fewer resources. There's much more to malware management than developing and maintaining an antivirus product, and the fact that the problem is largely contained (if not solved) in corporate environments is a tribute to the continuing efforts of many people inside and beyond the AV and security industries.
- ☑ While the antivirus industry is sometimes seen as swimming against the full-disclosure tide, its own work is based on a stringent trust model. As the range of malware threats has increased, it has had to build bridges beyond the "Old Guard" to other types of malware and security specialist.

You Should Be Certified

- ☑ (ISC)^2 specializes in certifications that are wide in scope and are particularly suitable for technical and hybrid managers, or security administrators with a broad range of responsibilities. They are awarded to people who not only pass exams, but have significant working experience in security.
- ☑ GIAC certifications do not generally require formal time-served experience in the industry, and are in general more specialized and hands-on. More advanced levels of certification require some creative writing as well as exam success.

- ☑ Most of the training supplied by antivirus/antimalware vendors is focused on the use of their own product ranges. However, when an AV vendor does offer product-independent training, it's well worth considering it as a way of benefiting from their experience in the frontline of malware analysis.

Should There Be a Vendor-Independent Malware Specialist Certification?

- ☑ AVIEN members, under the leadership of Ken Bechtel and with the cooperation of other members of Team Anti-Virus, have formulated a 3-4-tier certification model for antivirus specialists, ranging from an entry level Certified Anti-Virus Administrator grade through Specialist and Architect grades to Anti-Virus Instructor.
- ☑ These certifications would be welcomed by many in the industry as a way of demonstrating their skills and knowledge and having them formally acknowledged, but they have not so far received the sponsorship and buy-in from security organizations like CERT, vendors, or the big AV customers, that would give them the recognition needed to make them useful.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: What exactly are the membership requirements for AVIEN members?

A: To qualify for AVIEN full membership, there is a requirement that the individual and the organization they work for are not involved in any way with the commercial production, sales, or marketing of antivirus software, hardware, or related products, and that they support a minimum of 1,500 anti-virus users. In practice, there may be a little flexibility, but exceptions are taken strictly on a case-by-case basis. All AVIEN members are also AVIEWS members, and pay the same subscription fee, which is used to meet administrative costs.

Q: What are the requirements for AVIEWS members?

A: There are no formal requirements as such, though willingness to pay the subscription and behave responsibly on the lists is assumed! (See *http://*www.aviews.net/content/category/3/8/30/). Cyber-criminals and virus writers tend to get a cool reception, but among those who have joined are anti-virus vendor representatives, independent and vendor-aligned researchers, security experts in other areas, product testers, journalists, and systems administrators. Members of AVIEWS who don't qualify for AVIEN membership can subscribe to most of the same mailing lists and can often take part in projects, but have less of a say in the running of the organizations.

Q: Wouldn't it be more useful if AVIEN and AVIEWS members could exchange samples directly?

A: There are plenty of ways for independent and vendor-aligned researchers to exchange samples via other channels. Keeping that sort of traffic off the lists avoids complications about establishing who is qualified to handle malware samples.

Q: Why does the story about a Greek painter give rise to an English term with a Latin root?

A: The English term was, to all accounts, coined by Hazlitt after reading an account by Latin writer Pliny the Elder. The exact phrase was "ne sutor ultra crepidam," freely translated into the proverb "The cobbler should stick to his last."

Q: So what's wrong with the word signature?

A: The AV community objects to on the following grounds. 1) It perpetuates an obsolete view of virus detection as being based on scanning for static strings, rather than on more complex string searches incorporating wildcards, regular expressions, and other algorithmic approaches. In fact, many malicious programs can't be detected by scanning for a static string, and this has been the case for many years. 2) The term is still sometimes misinterpreted as suggesting that there is a single byte sequence that can be used to identify each virus, like a fingerprint or retinal pattern. In fact, where it is possible to scan for a byte sequence, scanners from different manufacturers may use very different sequences and algorithms to detect the same malware.

Q: If scanners are so much better now than they were in the 1990s, how is it that the number of malicious programs detected heuristically has actually fallen?

A: There are several likely reasons for this. 1) As malware creation has become less about bragging rights and more about criminal gain, the standard of malicious code has tended to rise to a more "professional" level. 2) Heuristic analysis is largely focused on analyzing code, but the fact that so much malicious code is now obfuscated with runtime packers and some form of encryption and polymorphism, raises several problems (www.blackhat.com/presentations/bh-usa-06/BH-US-06-Morgenstern.pdf). In fact, vendors are now often flagging packed executables as "suspicious" or "malicious" more-or-less generically. This may raise detection rates, but also raises the false positive rate. 3) Once code can be unpacked and analyzed, replicative code is often easy to flag. Other kinds of malcode, however, may be far harder to detect automatically.

Q: What does In the Wild mean, and is the term still valid?

A: There has been something of a shift in what we mean by "In the Wild" (ItW). Technically, a virus defined in WildList terms (www.wildlist.org) as being ItW has been reported to and by at least two WildList reporters, which suggests a likelihood that it's widespread. However, as the volume of non-viral malware continues to rise, it also becomes better targeted, and it's less realistic to assess its impact in terms of the number of sites hit. If it isn't widespread, it's less likely to be seen by an antivirus vendor. At any rate, it may be seen far later in its lifecycle.

Q: I've been using free antivirus on my laptop for years and never had a virus.

A: Ok. And you know you've never had a virus because your antivirus would have told you if you did, right? Spot the flaw in the logic...

Q: Where do psychological analysts fit into antivirus research? Are they proficient in assembly language?

A: People like Sarah Gordon understand the technical aspects of virus and antivirus coding very well. Their insights are valued because they add an extra dimension to our understanding of the whole problem of malware management, which goes far beyond technology.

Q: How useful is an award like CISSP to an antivirus researcher or administrator?

A: A general certification in security won't go very far in training you in the direct handling of malware, but it does prove that you have a broad knowledge of the security arena, enabling you to make connections between other sectors and your own specialties. SANS/GIAC training and qualifications, if carefully selected, may enable you to gain experience more directly related to malware analysis, for instance. Which approach is most beneficial in the market place? That depends very much on the sector you work/want to work in. Of course, hedging your bets with a range of qualifications does no harm at all.

Q: How useful are vendor-dependent training courses and certifications?

A: Clearly, they're very useful to administrators working with that vendor's applications, and a responsible vendor would normally include enough general information to enable the trainee to understand the problem as well as the solution. Some vendors also do courses that are less dependent on knowing their own products. These are rarer, but have the advantage of coming from an informed source.

Chapter 2

Stalkers on Your Desktop

Solutions in this Chapter:

- Malware Nomenclature
- 21st Century Paranoid Man
- The Current Threatscape
- Words Can Hurt You
- Fraudian Slips

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

No book on malware is considered complete without a history of the subject. Nevertheless, we're not going too far down that route. Since the subject has been covered in depth many times over, we will address it briefly, picking out a few highlights, and look in more depth at the current malware scene.

Tools & Traps

Malware Through the Ages

Yes, we know. You really wanted a complete history of malware from A for Ada Lovelace to Z for W32/Netsky.Z, and beyond, with a description of each and every variant of each and every malicious program along the way. Well, Robert Slade and David Harley have, from time to time, discussed writing a comprehensive history of malware (but not that comprehensive: we probably won't live long enough to document several hundred thousand variants) and an introduction to the personalities behind the industry, but it hasn't happened yet. In the meantime, here are a few resources that do include descriptions of some of the earlier manifestations of malware programmer malice. Unfortunately, many of them are out of print, and there are few recent books on the topic that we can recommend.

"Dr. Solomon's Virus Encyclopedia" includes a lot of virus descriptions. However, the second edition (by Dr. Alan Solomon and Dmitry Gryaznov was published in 1995, and never updated. It was distributed with production copies of the long defunct "Dr. Solomon's" anti-virus program, but we have occasionally seen it in bookshops. But not recently.

"Robert Slade's Guide to Computer Viruses" (2nd Edition) is also out of print, but includes a lot of historical data up to the earliest macro viruses. It was published in 1996 by Springer.

"Viruses Revealed" by David Harley and Robert Slade also includes much historical information, including a historical overview chapter. It was published in 2001 by Osborne, so is fairly current, but is out of print.

Peter Szor's "The Art of Computer Virus Research and Defense" (2005, Symantec/Addison-Wesley) isn't really historical. Actually, it isn't really about defense, either, but consists largely of first-class material on detection. However, it does include significant detail relating to quite a few malicious programs. And it's still in print.

In this chapter, Ken Bechtel and David Harley describe some naming issues, and look briefly at some examples of malware that were particularly significant in the "How We Got Here" section. David then reviews the current threatscape (a contraction of "*Threat*

Land*scape"*): however, this is very much an overview chapter, intended to fill in a few of the gaps that aren't addressed in detail in other chapters.

For instance, there's little point in writing at length about robots (bots) and robot networks (botnets) here when we already have a chapter by Tony Bradley and David Harley dedicated to the subject (Syngress also have a pretty good book on the subject—to which Tony and David also contributed—by Craig Schiller and Jim Binkley). However, even though spam and phishing are closely tied to the botnet phenomenon nowadays, being primarily disseminated through bot-compromised machines, we have not considered them in detail elsewhere, so they are among the topics we'll look at more closely here. In fact, the bulk of this section will deal with e-mail-related threats. This doesn't mean that there aren't other malware vectors. For instance, Igor Muttik's chapter deals with one—HTTP—in considerable detail.

Malware Nomenclature

From the earliest days of the anti-virus industry, naming malware has been an issue. This is not intended as a comprehensive discourse on the subject, but rather to give you, the reader, an understanding, and address the critics who will say we used the "wrong" name on any one of the malicious programs discussed later on.

The original malware naming convention was established by the Computer Anti-Virus Research Organization (CARO). This naming convention is used in some form by most vendors and is the basis for the convention used by the WildList Organization. It is very specific and scientific.

Tools & Traps

The CARO Full Name Convention

The general format of a CARO name is as follows:

<type>://][<platform>/]<family>[.<group>][.<length>].<variant>[<modifiers>] [!<comment>]

Please note that:

- Items in square brackets are optional.
- Only the family name and the variant name of a piece of malware are mandatory
  - The full name is white space-delimited. It cannot contain white space itself and is preceded and succeeded by white space.

Continued

By white space (or whitespace) we mean one or more of the following characters:

- Space
- Tab
- Carriage return (CR)
- Line feed (LF)

The current state of play of the CARO convention is described in meticulous detail by Vesselin Bontchev in a paper presented at the 2005 Virus Bulletin conference, and subsequently publicly available at www.people.frisk-software.com/~bontchev/papers/naming.html.

When a researcher finds new malware, he or she gives it a name and shares the sample with other trusted researchers. (CARO membership is strictly by invitation. Sharing between other researchers is similarly restricted to trusted individuals, rather than groups or organizations, even antivirus companies.) If the others agree, the name is adopted. Many times, multiple researchers will "discover" the same malware at the same time. This can lead to multiple names being given to the same threat, although it can also lead to the same name being given to more than one variant. (Part of the raison d'être of the WildList is to match names to specific variants correctly in a standard collection.) Most times, this is sorted out in a few hours. However, if it occurs during a mass outbreak or a media event, this leads to much confusion among end users and Information Technology (IT) support teams.

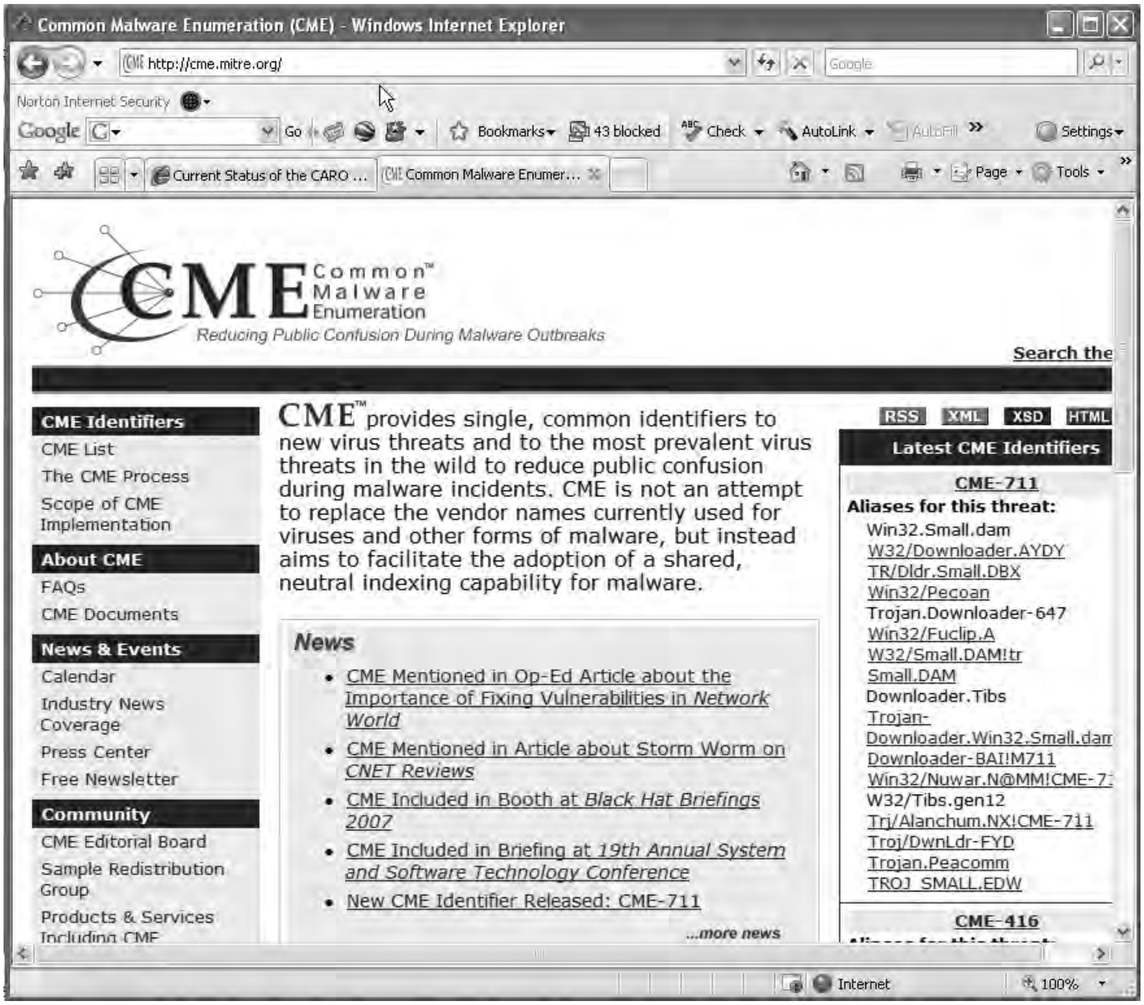
To this end, many vendors started adding a cross reference to what other names or aliases the malcode has attracted. Up to the late 1990s, this system, while slightly confusing, worked pretty well. Malware was replicating and spreading slowly enough to keep the process manageable. In addition to the vendor's cross-referencing, there were some utilities and databases interested parties could turn to, such as VSUM. (Maintained by Patricia Hoffman, this used McAfee virus names as the primary key. It was discontinued in the 1990s, but seems to have been updated online up to 1998 (www.wiw.org/~meta/vsum)).

The problem became much more complicated when mass outbreaks started in the late 1990s. First off, the public became painfully aware of the malware issue, and the press would talk to anyone who worked with computers for a living, as a presumed "expert" on the subject matter. Many times these "security experts" would give it their own name, and in some cases, the press would adopt a catchy name. W32/CIH, for example, was not called Chernobyl until almost a year after the discovery of the first variants, when someone noticed that some variants triggered on the anniversary of the Chernobyl disaster and chose to exploit that fact, regardless of the fact that it already had a perfectly good name as far as the industry was concerned. There are, in fact, no grounds to believe that there is any significance in the date of the payload.

This further muddying of the waters inspired new, more dependable cross-reference tools, like vgrep. This was initially created by Ian Whalley for Virus Bulletin while he was editor of the magazine, and the database is currently maintained by Dmitry Gryaznov. It is still available on the Virus Bulletin Web site (www.virusbtn.com). The effort to standardize the naming convention has culminated (as the date of this writing) with the Common Malware Enumeration (CME) effort, led by MITRE (http://cme.mitre.org).

So, while it would be nice if we could have everyone calling every piece of malware by the same name, we will never see it happen. Even if the industry standardizes, there are still those outside the industry that will continue to create their own names for already named, existing threats. As you can see from Figure 2.1, the industry is a long way from such standardization. CME-711, the latest identifier at the date of access (May 2007), has no less than 17 aliases listed (and we could probably find more.) In this chapter, we will do our best to stay in line with the CARO naming scheme rather than the media Nom du Jour. However, many of the earliest problem programs and non-viral malcode types defy that categorization, and we won't usually use the full CARO name.

Figure 2.1 Common Malware Enumeration



CME Identifiers
CME List
The CME Process
Scope of CME Implementation

About CME
FAQs
CME Documents

News & Events
Calendar
Industry News Coverage
Press Center
Free Newsletter

Community
CME Editorial Board
Sample Redistribution Group
Products & Services Including CME

CME™ provides single, common identifiers to new virus threats and to the most prevalent virus threats in the wild to reduce public confusion during malware incidents. CME is not an attempt to replace the vendor names currently used for viruses and other forms of malware, but instead aims to facilitate the adoption of a shared, neutral indexing capability for malware.

News

- CME Mentioned in Op-Ed Article about the Importance of Fixing Vulnerabilities in *Network World*
- CME Mentioned in Article about Storm Worm on *CNET Reviews*
- CME Included in Booth at *Black Hat Briefings 2007*
- CME Included in Briefing at *19th Annual System and Software Technology Conference*
- New CME Identifier Released: CME-711

...more news

Latest CME Identifiers

CME-711
Aliases for this threat:
Win32.Small.dam
W32/Downloader.AYDY
TR/Dldr.Small.DBX
Win32/Pecoan
Trojan.Downloader-647
Win32/Fuclip.A
W32/Small.DAM!tr
Small.DAM
Downloader.Tibs
Trojan-Downloader.Win32.Small.dam
Downloader-BAI!M711
Win32/Nuwar.N@MM!CME-7
W32/Tibs.gen12
Trj/Alanchum.NX!CME-711
Troj/DwnLdr-FYD
Trojan.Peacomm
TROJ_SMALL.EDW

CME-416

21st Century Paranoid Man

With a tip of the hat to the long defunct band King Crimson, one of whose song titles We have misappropriated for the title of this section, we take a quick look at how we got here (wherever that is.) Take a short stroll with us round the malware museum.

In The Beginning

Long ago, in a galaxy far away, there were not Star Wars, but Core Wars. These contests involved using a standard set of opcodes, known as Redstone Code, to build competitive programs within a virtual environment. While these games were about surviving rather than replicating, one of the major strategies used by such programs (http://www.koth.org/info/greg_lindahl_corewars.html) was, in fact, replication. "Replicators are programs which are capable of copying themselves to a new location in memory. They can be thought of as self-moving bombs." Another strategic approach was called the vampire. Rather than calling in Sarah Michelle Gellar and a supplier of wooden stakes, a common countermeasure was to write anti-vampire code to look for pointers to vampire "pits" in memory and overwrite them with useful code.

In 1982, the American Computer Society published a paper by Shoch and Hupp called "The Worm Programs – Early Experience with a Distributed Computation." Their worm (so called because it was, in a sense, segmented, since processing was "distributed" across a network of machines) became notorious after a programming error was propagated to the entire network, resulting in roomfuls of hung computers. And you thought Microsoft invented the Blue Screen of Death (BSOD)?

Some of the first more-or-less viruses were written for Apple II machines, even before Dr. Frederick Cohen's ideas on the subject were shared and gave birth to the name "virus" (by analogy with biological viruses.) By 1986, though, viruses had found their way to the PC (Ralf Burger's VIRDEM, a Proof-of-Concept file infector; Ashar and Brain, two closely-related boot sector infectors, and by 1987, we were saying "Good morning, Vienna," and getting well and truly "stoned," while Cascade introduced the Encryption Factor that subsequently developed into variable encryption/polymorphism. CHRISTMA.EXEC virtually invented some of the techniques that we still find in much of today's e-mail-borne (and other) malware; social engineering in the subject header, misdirection by displaying a graphical effect (an American Standard Code for Information Interchange [ASCII] Xmas tree of sorts) while it went about its real business, and replication by self-mailing to the victim's address book.

1988 saw some Mac viruses (Scores, MacMag), the hugely successful (in replicative terms) Jerusalem virus, and, spectacularly, the Morris Worm (or Internet Worm.) In "Viruses Revealed," Harley, Slade, and Gattiker wrote: "...the Internet Worm did not rely on any user actions at all, except for laziness on the part of managers who did not patch known problems...” Hmm. Not a lot has changed since then, except that in many cases, it’s the computer *owner* who doesn’t see the need to apply patches. That was also the year in which Peter Norton, who later lent his name to Norton (now Symantec) Anti-virus, was quoted as saying that computer viruses were an urban legend. Commodore also described reports of Amiga viruses as a hoax. They were quite wrong, by the way. The Amiga attracted quite a few virus writers.

1989 saw the Datacrime fiasco, Frodo (arguably the first full-stealth virus), and the AIDS information diskette Trojan. This masqueraded as an AIDS information disk, but slowly encrypted the victim’s hard disk when it was running. Eventually, the victim was faced with a demand for money in order to get the contents of their hard disk back. This not only resembles some previous extortion attempts using logic bombs on mainframe systems, but also prefigures some much more recent extortion attempts.

NOTE

I still have a copy of the AIDS diskette, but no 5.25″ drive to read it with, so I guess that makes it latent malware. In fact, I have a certain sentimental attachment to it: I can date my personal involvement with the malware management consultancy business in the form of the AIDS Trojan to the 19th of December 1989, the same day that my daughter was born. The same year as Virus Bulletin.

In 1990, we started to see serious attempts at armored viruses, multipartite viruses, more advanced polymorphism, and advanced encryption. 1992 saw Virus Creation Lab (VCL), one of the most prominent early virus generation kits, but certainly not the last. (And, of course, phishing kits are a staple item of the black economy.) The next few years saw plug-in polymorphic engines, one of the few high-profile virus author prosecutions (of the Black Baron, under his real name of Christopher Pile, under the UK’s Computer Misuse Act), and the beginning of an upsurge in e-mail hoaxes and chain letters.

In 1995, macro viruses became a real-life problem, rather than a research hypothesis, and became a major problem over several years, not least in environments where Mac users, clinging to the conviction that viruses were something that happened exclusively to PC users, radiated high volumes of malware across the Mac/PC divide. In 1997, mIRC worms and AOL Trojans gave early warning of the bot epidemics of the present decade. In 1998, Mac-specific viruses got something of a boost with the spreading of the AutoStart virus (or worm), followed by a handful of other highly malicious programs, notably SevenDust. In 1999, Melissa (W97M.Melissa.A@mm) marked a shift in emphasis from macro viruses (though it was a macro virus) to e-mail-borne viruses/worms (it was a mass-mailer). Most subsequent mass-mailers abandoned the macro format, though, and most have been either

Windows executables or, for a while, VBScript infectors like "LoveLetter," which appeared in 2000. At around the same time Distributed Denial of Service (DDoS) zombies were adding a further strand to the development of the botnet.

The early years of the first decade of the 21st century were dominated by mass mailers and other forms of worm. (We're not going to get into the argument about distinctions between worms, viruses, Trojans, and so on. Even we can't get very excited about those debates any more.) On the e-mail side, Matrix/MTX, Hybris et al added their own twists to the cocktail. Meanwhile, CodeRed reminded us that network worms still had potential, a message rammed home by a cluster of direct descendants and some variations on the worm theme like Blaster and Slammer. When we emerge, blinking in the sunlight, from the Malware Museum, we find that some of these bots on the threatscape are still with us in some form. However, there are a few issues that dominate the second half of the decade far more than they did a few years ago.

The Current Threatscape

So where are we now? The continued domination of OS X has all but neutralized the Mad Macs malware of yesteryear, although that could change (and the occasional Proof-of-Concept malware pops up from time to time to remind us that the Bad Guys are watching.) A quick look at the March 2007 WildList, the latest available at the time of writing (www.wildlist.org/WildList/200703.htm), is instructive. Looking at the Top List, the Supplemental List, and the list of malware that has just dropped off the other lists, we find:

- Virtually no macro viruses
- Lots of bots (no surprise there); SDBot, for example, continues to thrive
- Virtually no VBS infectors
- An absence of ancient boot sector infectors and DOS file viruses
- Some residual wormcast like Nachi and Blaster, plus e-mail wormcast like Hybris and Klez
- A surprisingly high percentage of other mass-mailers; however, we're talking here of the fairly recent mass-mailers like Mytob, MyDoom, Netsky and Sobig that mark the transition from pure mass-mailer to quasi-commercial tools and botnet components.

We won't talk about bots here: you'll have bots coming out of your ears by the time you get to the Index. In fact, we'll spend very little time on anything that's addressed at length in other chapters. Instead, we'll concentrate mostly on mass mailers, spam, hoaxes, and e-mail fraud. But we'll kick off this section with a short consideration of a general class of malware that has all but taken over from the virus.

The Rise of Troy

According to some traditional sources, Troy (or at any rate, the incarnation thereof that we're most familiar with through the works of Homer and Virgil) fell sometime around 1184 BC. However, the Trojan horse (or just Trojan – capitalization is disputed in this context, and you could argue that the original Trojan horse was, strictly speaking, Greek) is alive and well on a desktop or in a mailbox near you.

Notes from the Underground

Yet Another Taxonomy of Malware

Unfortunately, I don't feel able to continue on this theme without a perfunctory attempt to address the differences between a number of different forms of malware. Perfunctory, because if you'd done this as many times as I have over the past 20 years, you'd be bored with it too. Still, it's as well to make sure we're reading from the same hymn-sheet.

- A common definition of a virus is "a program that modifies other programs to contain a possibly evolved version of itself." In other words, a virus requires a host to attach itself to, either by modifying the host, or otherwise attaching to the host. The primary characteristic of a virus is, therefore, replication. A worm also replicates (and many researchers regard worms as a subset of the wider class of viruses) but don't modify or attach to a host. (No, neither of these definitions are universally accepted. But we're not going down that fascinating byroad this time.) The late Simon Widlake had a neat way of defining the difference: "Viruses infect, worms infest."
- A Trojan horse is a program that disguises its true function, or, to quote Simon again, "It doesn't do what it says on the tin." Or, more formally "a program that appears to perform a legitimate function, and may even perform that function, but also performs some covert action that the system owner did not expect and would not want to happen." Thus, the general class Trojan is sometimes considered to include viruses and worms, at least where the replicative malware concerned replicates or triggers by tricking the victim into taking some action to execute malicious code. It's certainly generally considered to include:
    - Password stealers, keyloggers, banking Trojans (yes, that's a bit of a recursive definition there), other crimeware.
    - Backdoors

Continued

- Remote access Trojans
- Spyware (certainly non-replicative spyware)
- Adware; at any rate, adware which manages to sneak itself and activate on a system without the system owner realizing that it's done so
- Rootkits
- Dialers and porn dialers
- Logic bombs (programs that "trigger" under a predefined set of circumstances)
- Droppers (programs that install malware)
- Bots, DDoS agents
- Some joke programs
- Possibly Unwanted Applications or Programs (PUAs, PUPs), Greyware.

I can't begin to do justice to all these flavors of Trojan in a single chapter, so I'm going to concentrate on an area that we haven't discussed elsewhere in this book.

Rootkits

The term *rootkit* is applied unselectively to a range of technologies, and has become the "hype of the month" in some quarters. After all, rootkit technology uses stealth techniques to render it invisible to security software and even to the operating system and file system.

On a UNIX system, the most privileged user (the administrator, in other environments) often has the name "root." This user controls the system, and is therefore the most desirable target for an attacker. So "rooting" a system (at least to non-Australians: don't ask...) means getting access to the root account, and thus gaining access to the system and all its files and directories. In the past few years, however, rootkits have been seen increasingly on Windows platforms. In fact, while rootkit "stealth" is regarded with superstitious dread, the twin threats of privilege escalation and covert compromise are nothing new to security, and have been dealt with by anti-malware technologies for many years. Anti-virus software can and does detect rootkits as successfully as it does viruses. The problem is, however, manageable. The sky is not falling.

According to Greg Hoglund ("Rootkits are not Malware," www.rootkit.com/newsread.php?newsid=504; http://www.sysinternals.com/Forum/forum_posts.asp?TID=5798) a rootkit is "a set of programs and code that allows a permanent or consistent, undetectable presence on a computer." However, Harley and Lee (http://www.eset.com/download/whitepapers/Whitepaper-Rootkit_Root_Of_All_Evil.pdf) prefer to differentiate between:

- Old-style "stealthing" (using techniques comparable to those used by semi-stealth and full-stealth malware, especially viruses, since time immemorial)
- Newer "stealthkits" (a broad class of malware that attempts to stay hidden while retaining control over a compromised system)

- "Real" rootkits (a toolkit intended to gain and/or maintain privileged access to a system.) The difference between a stealthkit and a rootkit is sometimes defined as the availability of tools to escalate privilege, but this is by no means universally accepted. Of course, many people don't differentiate between stealthkits and rootkits at all.

Are You Owned?

Got Something To Hide?

Concealment can actually involve quite a few "objects," not just malicious files:

- Files
- Folders/directories/subdirectories
- Handles
- Open Ports
- Processes
- Registry entries
- Threads

I should also acknowledge Hoglund's point that modern definitions do not necessarily presuppose unauthorized intrusion, malicious intent, or inappropriate and unauthorized escalation of access and privilege. In fact, concealment of and restricted access to data and to systems files and processes is a standard feature of multi-user operating systems (including modern versions of Windows: the difference between workstation and server environments is as much a matter of usage and configuration as of hardware and operating system, in the modern office.) A well-known instance of a "rootkit" installed with the legitimate aim of maintaining Digital Rights Management (DRM) is the infamous "Sony rootkit." The XCP copy protection system used by Sony in this case restricted the number of copies of CDs or DVDs that could be made, and also controlled the "ripping" of music into a digital format suitable for storing and replaying on a computer or MP3 player. It was only possible to play one of the affected CDs on a PC by installing software which then hid files, processes, and registry keys/values by modifying the execution path of application program interface (API) functions. It used the common rootkit technique of patching the System Service Table (SST).

Continued

Here is a short list of other areas where "stealth technology" is or might be used legitimately

- Assessing the threat from insiders and other monitoring of employees
- Encryption and concealment of data on multi-user systems
- Intellectual Property Rights (IPR) management
- Intrusion tracking
- Protection of security software, including detection technology, backup, and system recovery software, from spoofing and reverse engineering

Legitimate functions aside, a rootkit may have a number of secondary objectives, including:

- Concealing the fact that malicious applications and processes are present, often by passing them off as legitimate files
- Concealing the presence of vulnerabilities and exploits
- Harvesting information about the system and system user
- Attacking other systems, using the compromised system as an intermediary resource
- Concealing the storing of other malicious applications and the use of the system for other purposes, such as membership and/or control of a botnet

Kernel Mode and User Mode

System services on an NT-derived system (Windows 2000, XP, Vista) run in kernel mode. (Vista actually introduced modifications that are not addressed here.) This means that an unprivileged user cannot introduce inappropriate modifications, removals, and additions of drivers, devices, and applications without appropriate authorization.

User applications run in user mode, and an application's ability to cause damage through inadvertent modification to system processes is limited. User mode rootkits can run as (or within) a user application, by patching the Windows Application Programming Interfaces (APIs). Since each user application runs in its own memory space, the rootkit must modify the memory space of every running application, in order to filter each application's "view" of the environment. A kernel mode rootkit has more control and can do more damage, but is harder to install and maintain reliably.

Persistency and Non-Persistency

Persistent rootkits are stored on disk and can survive a system reboot. Non-persistent or in-memory rootkits install code directly into volatile memory, and do not survive a reboot. This limits the length of time during which they function, but makes them harder to detect.

Many of the proof-of-concept exploits currently in vogue favor non-persistency for that reason.

Rootkit Detection

UNIX and similar systems have long been protected more or less generically from most rootkit attacks by the object reconciliation or integrity checking approach (exemplified by the use of the "tripwire" application). This mostly takes the form of monitoring system objects for evidence of tampering or modification. In the Windows context (indeed, in *any* highly graphical environment), however, this approach can have a high maintenance overhead, as changes in the environment (system code, registry settings, configuration files) are routine. In fact, many such changes take place in any Windows session. Non-persistency stresses the need for memory scanning and the detection of hidden processes as an indicator of compromise. Heuristic approaches necessitate acute discrimination between legitimate and malicious hooks and registry, and an ability to spot discrepancies between a "trusted" view of the system and the "tainted" view presented by a system when filtered through rootkit or stealthkit technology.

Detection is only half the problem, of course. If a rootkit is only detected after installation, removal can be non-trivial. It's sometimes more efficient and safer to re-image than to remove the rootkit, though I'm not altogether sympathetic to the view that *any* rootkit compromise necessitates complete reinstallation of the system.

NOTE

To be precise, re-imaging is sometimes more efficient with all forms of malware, especially some forms of spyware (remembering that some rootkit components fit into this category.) These are not only deeply embedded and taking advantage of hooks and vulnerabilities that defy simple detection, but are also able to self-modify, update, and transform in ways that make a simple "Remove files X, Y, and Z and registry keys A and B" approach to removal unfeasible. The anti-virus community is well aware of these problems, and is generally meticulous about taking a responsible approach to sound malware removal, whether in AV software or in dedicated anti-rootkit utilities, but there are no absolutes to cling to here.

Detection is therefore, as always, best supported by other countermeasures such as good backup practice, sound access control, and adherence to the principle of "least privilege" (e.g., using an unprivileged account for routine work.)

Now we take a step away from out-and-out malware, and look at some of the issues around security and messaging.

Words Can Hurt You

Sticks and stones can break your bones, but words can be pretty damaging, too. Here are some examples of e-mail abuse that can have a serious impact on the financial welfare of the individual or business.

Spam, Spam, Spam

I nearly didn't include this section; after all, we all know what spam is, don't we? However, we all have different definitions, including Hormel, the manufacturers of the meat product after which it's named. They insist that when referring to e-mail abuse, their capitalized trademark should not be used, by the way.) So this seems a good point to define some terms and then expand on some specific attacks.

Originally, the term spam was applied to inappropriate cross posting and other abuse of newsgroups. Now it's usually applied to e-mail, though it may used in the context of other forms of messaging, such as SMS texting and Instant Messaging services (Spim.) It may be applied by different people to almost any form of unsolicited e-mail. However, not all unsolicited mail is spam. Otherwise, no one would ever receive mail they hadn't given advance permission for. There is a technique called whitelisting that essentially means you don't read mail from people you haven't specifically listed as an "acceptable" source of mail. This is fine as long as there is some way for people needing to send legitimate mail to you to get onto the "invitation list."

Most definitions of spam include Unsolicited Commercial E-mail (UCE) and Unsolicited Bulk E-mail (UBE). I often try to distinguish between:

- "Professional" spam (or hardcore spam, but that can lead to confusion with pornographic spam) which may be illegal and is certainly unwanted by most people
- Inappropriate and ill-considered marketing mail from legitimate firms who are not quite on top of what is legal
- Acceptable marketing practice
- Other forms of e-mail abuse

There is no absolute or universal legal definition for "professional Spam." I have in mind unsolicited bulk mail (usually commercial), which is:

- Totally untargeted; mailed to every available address, irrespective of where the recipient is and how interested they're likely to be
- Sent out in high volumes to address lists acquired without regard to the wishes of the addressees

- Advertising products or services that may or may not be acceptable or legal (such as pirated software, various types of pornography, medication, and so on)
- Involving some degree of deception, such as the following:
    - Claims not to be spam; a pretty good detection heuristic, even if it's something of a circular definition
    - Claims that the recipient in some way invited the spam
    - The sender masquerades as a friend, colleague, relative, and so on
    - The message headers are forged to make it harder to track the source
    - There is a deceptive Subject header to persuade the recipient to open the message
    - Involves some degree of actual fraud
    - Uses techniques clearly intended to circumvent spam filters (hashbusters, image spam, header spoofing, and so on)

By "amateur spam", I mean bulk mail sent inappropriately and inadequately targeted by more-or-less legitimate enterprises, but without due regard for acceptable practice or even legal requirements. In fact, now would be a good point to mention (not for the last time, even between the covers of this book) Paul Vixie's definition of what constitutes fully frontal spam (that is, mail that meets all three of these conditions):

- "The recipient's personal identity and context are irrelevant because the message is equally applicable to many other potential recipients."
- "The recipient has not verifiably granted deliberate, explicit, and still-revocable permission for it to be sent."
- "The transmission and reception of the message appears to the recipient to give a disproportionate benefit to the sender." (Sendmail Theory & Practice, 2nd Edition, Paul A. Vixie and Frederick M. Avolio, Syngress.)

I have some sympathy with the view that if you don't want it, it's spam. However, from a system administrator's point of view, this subject definition is more than difficult to accommodate. Still, it can be useful to distinguish between the forms of spam described in the preceding sections and the following, which I tend to classify as scams rather than spam, though sometimes the borderline is seriously fuzzy:

- 419 "Nigerian" scams/frauds
- Lottery fraud
- "Phishing" fraud: that is, mail sent with the intention of tricking you into giving away sensitive data, especially financial data.

- Other attempts to ascertain sensitive information such as ISP passwords
- Virus and other security hoaxes
- Other forms of chain letter
- E-mail viruses, or mail generated by anti-virus and other filtering software in response to messages that seem to contain viruses or other forms of abuse

Spam is not synonymous with pornography, but pornographic e-mail seems to be the form of spam that offends and upsets most people. Administratively, no one should feel that they are under threat of disciplinary action purely because they receive unsolicited pornographic e-mail. Legally, pornography poses difficulties in definition, and is not necessarily illegal. Sometimes, there's no point in doing much except deleting it.

Special handling is needed for material that is or may be child porn/pedophilia-related, or other material that may be illegal within jurisdictions to which you are subject. Unfortunately, many lawmakers have been panicked into passing poorly, confusingly defined legislation, inspiring similar panic in those who have to work with it.

Fraudian Slips

There have always been scams lurking in the darkest corners of the Internet. Indeed, pyramid schemes, lottery pyramids, Ponzi schemes, homeworker business "opportunities," and so forth have been around longer than e-mail, and are still seen, often in more sophisticated forms. The only fraud of this type I'll consider here, though, is the trusty 419. While this type of fraud isn't generally associated with malware, it leads quite nicely into the subject of phishing, which certainly is.

Advance Fee Fraud (419s)

This type of scam often overtly originates in Nigeria, the Cote d'Ivoire, and thereabouts. (Section 419 of the Nigerian Criminal Code deals with fraud, hence the name.) These offer large sums of windfall money, but when the victims respond they are told that they have to pay advance fees of various kinds—tax, bank charges, bribes—before the windfall can finally be delivered. Nowadays, they're most often seen as e-mails, though they used to be seen often as faxes or letters. In fact, the basic sting is pretty much modeled on a scam called "The Spanish Prisoner," which some sources date back to the 16th Century.

Variations on the theme are endless, but may include:

- A request for help from a political refugee to get their money out of their country and into yours
- A request for help with the distribution of money for charitable purposes

- A request for assistance from a bank or other official with transferring money obtained more or less illicitly from their employer or the government
- A request to stand as "next of kin" for the purposes of claiming the estate of a dead foreigner who has died intestate
- Notification of a lottery win

These communications often derive from Nigeria or another African state, but may appear to originate from any part of the world, especially if they're tied to a particular historical event. 419 perpetrators use target addresses harvested directly or indirectly from Web sites, newsgroups, mailing lists, and so on, in much the same way as other spammers and scammers. As law enforcement and the media always tell us, if it looks to good to be true then it probably is.

Phishing Scams

Phishing is the use of a deceptive message (usually an e-mail message) to carry out fraud. The term was originally largely limited to stealing AOL account passwords and credit card information. It clearly refers to the practice of "fishing" for victims, using a deceptive message as bait. Since 2003-2004, the term has become associated with passing off requests for sensitive data as if they come from all sorts of major organizations (and their customers) in and out of the financial sector, notably:

- Banks
- Building societies
- Credit unions
- eBay and PayPal, eGold, and so on
- The IRS, healthcare and educational organizations, and so on.

Rather than the unsophisticated AOL messages and Trojans of yesteryear, the term is now associated with social engineering and brand theft using tricks such as:

- Fraudulent Web sites constructed to resemble a real site.
- Pop-up or pop-down forms designed to appear when the real site was accessed via a link in the e-mail.
- Deceptive links, disguised in various ways to resemble links to a real site.

The term also includes e-mails designed to facilitate the planting of malware and spyware such as keyloggers, password stealers, and backdoors, either directly or from spammed Uniform Resource Locators (URLs) using drive-by downloads or misleading offers of Trojanized "goodies." Phishing messages are often attached to messages passed off as an invitation to view an electronic greetings card from "a friend," or to contact a prospective romantic partner.

The phishing message is designed to trick the recipients into thinking that it comes from a legitimate business or agency, so that they will be prepared to hand over sensitive and exploitable data to the scammer. The phisher's intention is usually to gain unauthorized access to victim's financial resources, or to steal their identity in order to defraud others. The posting of the deceptive message is only part of the phishing process, which is not always restricted to opportunistic, short-term exploitation of credit cards and financial data. It may be the jumping-off point for full-scale identity theft, involving, for example, taking out large loans in the name of the victim.

Phishing gangs operate within a complex "black economy" resembling other supply-and-demand economic models and infrastructures. Gang members may take on a wide variety of roles including:

- Harvesting target information such as e-mail addresses
- Setting up phish sites, often using widely available phishing kits
- Acquisition of spamming tools and compromised systems (e.g., botnets and zombie networks) as hosts for bait dissemination
- Compromising and accessing host systems to house scam pages
- Retrieving stolen credentials from anonymous mailboxes via a scriptable bot for control and data transfer
- Supplying victim's credentials for conversion to cash
- Using stolen credentials to buy goods for sale on the black market

A phishing attack has three parts: bait distribution (mostly through e-mail or instant messaging, though other vectors such as Voice over Internet Protocol [VoIP] are increasingly used), data collection (usually through a fake Web site), and the use of the misappropriated information for purposes of fraud and identity theft.

Phishing e-mails take many forms, from crude plain text to sophisticated messages virtually indistinguishable from the real thing, in terms of presentation. The technical skills of the criminal are not the only factor. Social engineering techniques are another major factor. Most often, scare tactics like "your account has been compromised: to re-authenticate, click here, before we stop you accessing your account" are used, though it's not unusual to offer rewards for information. However, the criminal's most potent weapon is confusion about the nature of the problem. As always in security, there's a lot of bad advice around. But it's particularly important that legitimate institutions don't "groom" their customers to be victims, by using bad practice in their communications that make it easier for customers to fall for scams, and by giving unreliable advice and information.

Notes from the Underground

Phishes and 419s

Phishing scams and 419s are often confused, even by people in the security arena, and in fact, there are no absolute differences, though someone working in this field will readily distinguish one from the other. As Harley and Lee point out in "A Pretty Kettle of Phish" (http://www.eset.com/download/whitepapers.php), there are tricks of tone and phraseology that are unmistakably "419": for instance, an obsession with words and phrases like "modalities" and "From the Desk of…", and some of the topics are virtually unique to the 419:

- Bank account proxy (next of kin) scams
- Lottery scams
- Some kinds of job scam

Some other types and topics (fake charity and disaster relief appeals, mule recruitment scams), resemble the output of phishing gangs in topic, if not in tone. Some of the similarities between the two types of scam include:

- Implementation by organized gangs
- Use of money laundering as part of the process
- Taking opportunistic advantage of natural disasters and personal tragedies
- Highly stereotyped phrasing and presentation
- Using some form of identity/brand theft; claiming to be someone else or to represent a real (or real-sounding) legitimate organization

Some of the dissimilarities include:

- 419s often have a strong overt African connection, though it isn't practical to distinguish between frauds on a purely national basis. In fact, many 419s have or pretend to have a European or Asian origin.
- 419s often require an element, sooner or later, of personal contact, whereas phishing gangs tend to work at a distance.
- 419s generally rely more on social engineering attacks than on technical attacks such as cross-site scripting (XSS) or domain name system (DNS) spoofing.

Continued

- 419 scammers often cite a legitimate Web site (especially a news resource) as a spurious means of validating the "story" in the scammer's e-mail, but are less likely to build a deceptive Web site.
- The 419 scammer is likelier to claim to represent a fake organization or group of individuals, whereas the phisher generally relies on spoofing a genuine organization's mail and/or Web site.
- 419s tend to work on a more personal/individual level. The scammer often claims to represent an organization such as a bank, military, or governmental organization, but the "deal" offered would, if genuine, often be against the interests of that organization.

Or Would You Rather Be a Mule?

A common manifestation of phish gang money laundering activities is the mule solicitation e-mail. Characteristically, these offer what are often described as "financial management" jobs. If you follow up on one of these job offers, you'll find that they boil down to:

- Receiving money
- Taking a cut as commission
- Forwarding the rest of the money further up the chain.

Funds transfer scams are often associated with phishing scams, but they don't purport to come from the same type of institution, and they aren't aimed at cleaning out the mark's (or target's) accounts. They are more concerned with setting up money-laundering mechanisms, using the target as a "money mule." They advertise "jobs" via e-mail and recruitment Web sites to people prepared to act as their local agents. The mule is expected to supply bank account details, and are encouraged to open new legitimate accounts with specific financial institutions. It's quicker and easier to move funds from a compromised account to another account at the same financial institution. While this may look obvious, money laundering, when boiled down to the three bullet points described above, the scammer may go to extreme lengths to make the business look legitimate.

Such solicitations may take many forms and disguise the "job" with various job titles and descriptions. In some cases, a large and complex Web site may be constructed with all sorts of circumstantial detail such as product brochures, but that's simply to reinforce the impression of a genuine business. Here are a couple of fairly crude examples.

Headers have been snipped to protect the easily bored: I've also edited out some white space and changed some details to protect the guilty. I did think of correcting the English, but was too depressed.

From: “Viktor XXX, YYY Co” <ftvjg@gbla.com> <[a fabricated address]>

To: <one of the author’s honeypot addresses>

Subject: Rescue of the nature

I am Viktor XXX, the president of the YYY Co, nongovernmental international organization, created in 1994 in order to provide assistance to individuals and legal persons in the protection of environmental rights, to promote the development of the environment protection, environmental education, science and culture.

This is a common mule solicitation technique: how can you resist a job opportunity that also gives you a chance to work for a Good Cause?

YYY Co. is based in Ukraine. Protection Co. has a lot of partners in Ukraine and abroad We have many sponsors and Maecenas around a world.

Unfortunately we are currently facing some difficulties with receiving payments from them. It usually takes 10-30 days before a payment from your country is received and cleared. Such delays are harmful to our organization. Wire transfers involve various bureaucracy-related delays and complications, and international cashier’s checks or money orders take up to two months to clear. That’s why we are currently looking for partners in your country to help us accept and process these payments faster.

Straight to the point. Sooner or later, it’s always about passing money through intermediate bank accounts and taking a cut, which is a rough and ready definition of moneylaundering.

If you are looking for a chance to make some additional income, you can become our representative in your country. As a representative of Protection Co, you will receive 8% commission of every deal we conduct.

Your responsibilities will consist of receiving payments in the form of wire transfers and/or checks and forwarding them to us. It is a part-time job that you will be able to do from the convenience of your home without any interference with your primary job. We are also considering opening another office in your country in the near future and you will have certain privileges should you decide to apply for a full-time job with Protection Co.

They must know that I’m a starving author desperate for a job!

If you are interested in this position with our company, please contact me for more information via [address edited]@aol.com

This is not the sending address in the headers or anything like it, by the way, and it’s curious that after 13 years they’re still using AOL accounts.

Be sure to include the following information about yourself:

1. Your Full Name as it appears on your resume.
2. Education.
3. Your Contact Address.
4. Telephone/Fax number.
5. Your present Occupation and Position currently held.
6. Your Age

```
Please respond and we will provide you with additional details on how you can
become our representative. There are no start-up or hidden costs for you.

Your commission-based income will depend on how fast you can process the payments
from our partners.

Should you have any questions, please feel free to contact us.

Sincerely,

Viktor XXX,

CEO YYY Co.
```

Tempting. Thanks, Viktor. Don't call me, I'll call you.

Here's another little gem with something of a 419 flavor. (Unfortunately, some are more convincing than these.) Note that the contact details are blatantly spurious. For instance, the post-codes is actually for the city of Bath, not London, and 0702 is not a landline exchange prefix.

```
ZZZ ASIA TRADING COMPANY PLC.®

MRS. ANTHONIA BLAIR.(ONLINE JOB MANAGERESS)
```

Any relation to the former Prime Minister? Surely not… It's actually very 419 to use a famous (real or fictional) person's name or a close variant.

```
HUMAN RESOURCES.

Units 9 & 10

Park Road

LONDON,

BA11 1EU

United Kingdom.

TEL: +44(7024052287)

FAX: +44(7024084555)

N.B:(WEBSITE IS UNDER RE-ENGINEERING,UPDATING SOME AREAS.)

REPLY TO EMAIL:([an address]@yahoo.co.uk)
```

Well, that saves the trouble of building a fake web site.

Hmm. How many businesses use Yahoo email addresses?

```
—COMPANY'S BRIEF INTRODUCTION:

FIBER ASIA TRADING COMPANY PLC.®

is a highly visited B2B portal dedicated to Accessories, Garment and Fashion
Industry, an authentic means for corporate planners & advisors. Industry wizards
through out the globe rely on our information, news and analysis. Among them
are the Apparel and Fashion Professionals, CEOs, Senior Management Executives,
Managers, Export officers, Purchase and Production Heads, Marketing Managers
and other decision makers of various companies from 190 countries.

DEAR ESTEEMED CANDIDATE,

WE HAVE A PART TIME JOB OFFER AVAILABLE FOR YOU IN RESPONSE TO YOUR INITIAL
REQUEST ON THE JOB SEARCH DIRECTORY.WE ARE A COMPANY BASED IN UNITED KINGDOM.
```

WE HAVE BEEN RECEIVING ORDERS FROM UNITED STATES OF AMERICA,CANADA, AUSTRALIA, AND EUROPE WHICH WE HAVE NOT BEEN ABLE TO PROCESS COMPLETELY SINCE WE DO NOT HAVE A PAYMENT RECEIVING PERSONNEL/BOOK KEEPER/PAYMENT RECEIVEABLE SPECIALIST IN THESE AREAS.WE HAVE DECIDED TO RECRUIT PAYMENT OFFICERS ONLINE HENCE WE WILL BE NEEDING A BOOK KEEPER/COMPANY REPRESENTATIVES IN THE UNITED STATES TO PROCESS OUR PAYMENTS IN THESE AREAS - DUE TO DELAYS IN PROCESSING PAYMENTS FROM THESE AREAS IN UNITED KINGDOM.

There has to be a reason why they need an intermediary, which usually because of a spurious money transfer problem.

The main reason why the funds cannot be cashed down here is that it would take 20–24 days for a CERTIFIED US CHECK OR CASHIER'S CHECK to pass through the bank here and that would tie down our capital and won't make business move fast as i expected, but a US CERTIFIED CHECK presented to the bank in the US would cash immediately or clear if deposited in the bank within 24–48 hours tops.

This makes us loose customers sometimes because we don't find it easy receiving payments from customers from these areas and it affects the patronage of our business.

Yes, I have a lot of trouble with loose customers, too. Oh, you *lose* customers? I see.

WHAT WE OFFER:

FLEXIBLE PROGRAM: TWO HOURS/DAY AT YOUR CHOICE, DAYTIME AND EVENING TIME WORK AT HOME: CHECKING E-MAIL AND GOING TO THE BANK PART TIME WITH VERY GOOD SUPPORT AND COMMUNICATION SKILLS OTHER HIGHLIGHTS: NO SELLING INVOLVED, NO KIT TO BUY,WE WON'T CHARGE YOU ANYTHING.

COMMISSION:10% OF EVERY CERTIFIED CHECK/ CERTIFIED CASHIER'S CHECK THAT IS CASHED INSTANTLY "CASH IN HAND" OR "CASH ON COUNTER" IS WHAT YOU GET FROM THE TOTAL CASHED AMOUNT.

PLEASE NOTE:

EXAMPLE IF YOU RECEIVE A CHECK OF$1,000.00 YOUR NET INCOME IS $100.00,OUR COMPANY SUPPORTS ANY FEES. YOU WILL PROCESS AT LEAST 2–3 ORDERS PER DAY AND REMEMBER THE MORE ORDERS, CHECKS YOU PROCESS - AT A FASTER RATE THE HIGHER YOU STAND TO EARN DAILY.

WHAT WE ASK: TWO FREE HOURS DAILY NOT INCLUDING WEEKENDS, INTERNET ACCESS FOR SENDING AND RECEIVING E-MAILS, AVAILABLE MEANS OF CASHING CERTIFIED MONEY ORDERS/CERTIFIED CASHIER'S CHECKS AT YOUR BANK USING YOUR BANK ACCOUNT.

IMPORTANT:

YOU MUST BE OVER 21 YEARS OF AGE.IF YOU MEET THESE CONDITIONS PLEASE CONTACT WITH THIS FOLLOWING INFORMATION:

EMPLOYMENT FORM:

NAME:

HOME ADDRESS WITH ZIP CODE. (NOT P.O.BOX):

PHONE/FAX NUMBERS:

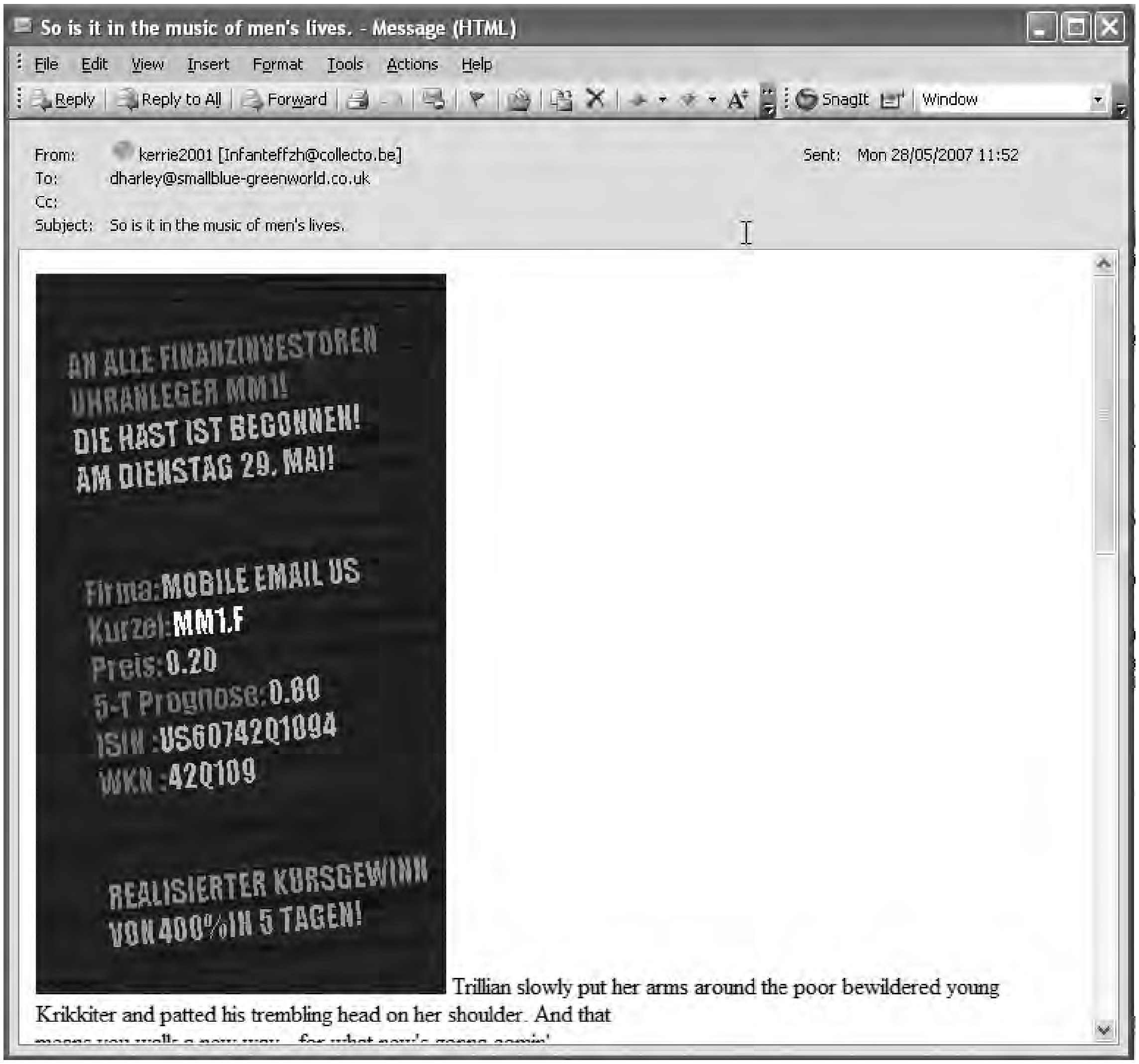
```
EMAIL ADDRESS FOR QUICK DELIVERY OF PAYMENTS:
AGE:
GENDER:
PROFESSION:
NATIONALITY:
MODE OF IDENTIFICATION:

PLEASE INTERESTED PARTICIPANTS,BOOKKEEPER,PAYMENT RECEIVEABLE SPECIALIST SHOULD
FORWARD THEIR INFORMATION TO THIS E-MAIL ADDRESS:
([an address]@yahoo.co.uk)

WE WILL NEVER ASK YOU FOR ANYTHING MORE THEN THAT, NO BANK NAMES, NO BANK ACCOUNT
NUMBER, ROUTING NUMBER, CREDIT CARD, PASSWORDS, SSN # ETC.IF ANYONE ASKS FOR THOSE
ON OUR BEHALF PLEASE DO NOT GIVE OUT THIS INFO.THIS IS TO ENSURE YOUR SECURITY
AND NON INVOLVEMENT IN CASES OF IDENTITY THEFT.

Please also note we have security experts who work in hand with the FBI & IRS
to detect the validity of the details you provide to us and our business AND NOTE
THAT YOU'LL BE PAYING NO TAXES WHATSOEVER. This has been sorted out with (IRS) and
also note that this is not any form of MONEY LAUNDERING.
Please take your time to go through our website upon imminent completion and
please contact our customer's service department for any question.

Warm Regards,
MRS.ANTHONIA BLAIR. (I) ONLINE JOB MANAGERESS.
MR. RICHARD BROWN. (II) GENERAL JOB CO-ORDINATOR.
MR. TONY ALLEN. (III) DIRECTOR OF PAYMENTS.

COMPANY WORKING HOURS:
9:00am to 6:pm all days.
(PLEASE PHONE AND FAX LINES WILL BE UNAVAILABLE AFTER COMPANY WORKING HOURS)
```

Make no mistake: this may look like a "nice little earner," but money laundering is illegal in most jurisdictions, and the muledriver is not likely to baulk at disappearing from the employee's life, leaving him (or her) to hold the baby. The fact that the muledriver went to some length to hide the real nature of the business from them won't necessarily be seen as a mitigating circumstance.

Pump and Dump Scams

Pump and dump mails are intended to inflate the value of stock temporarily by advertising it to potential investors. As the victims buy into it, the value of the stock rises, and the scammers sell off their shares. The stock promptly falls in value, and the new investors sustain a financial loss. These mails are often seen as a minor nuisance, especially in countries that don't have the same penny stock trading culture as the US. However, such scams are rising in volume and widening in geographical scope (my German has improved a lot in the past few weeks!), and conceal the significant involvement of organized crime. These schemes entail remote manipulation of the stock market, not the direct plundering of a victim's bank account that characterizes the phishing scam. See Figure 2.2.

Figure 2.2 German Pump and Dump Scam


So is it in the music of men's lives. - Message (HTML)
File Edit View Insert Format Tools Actions Help
Reply Reply to All Forward SnagIt Window

From: kerrie2001 [Infanteffzh@collecto.be] Sent: Mon 28/05/2007 11:52
To: dharley@smallblue-greenworld.co.uk
Cc:
Subject: So is it in the music of men's lives.

AN ALLE FINANZINVESTOREN
UNRANLEGER MM1!
DIE HAST IST BEGONNEN!
AM DIENSTAG 29. MAI!

Firma:MOBILE EMAIL US
Kurzel:MM1.F
Preis:0.20
5-T Prognose:0.80
ISIN :US6074201094
WKN :420109

REALISIERTER KURSGEWINN
VON400%IN 5 TAGEN!

Trillian slowly put her arms around the poor bewildered young Krikkiter and patted his trembling head on her shoulder. And that

Not surprisingly, this activity, despite its illegality, is reported to be making a great deal of money for the Mafia (www.businessweek.com/1996/51/b35061.htm: "The Mob On Wall Street," by Gary Weiss) and other criminal organizations. The scammer doesn't need direct contact with the victim (and may not have any connection with the firm whose stock is being promoted.) It's worth noting that a victim is not only likely to lose money, but may also be vulnerable to charges relating to insider trading, depending on which jurisdiction they're in.

Hoaxes and Chain Letters

The anti-virus industry has a fairly relaxed attitude to hoaxes and chain letters, especially now that hoaxes relating to the latest computer-munching wonder-virus fantasy have declined. (Though, even as I write, there is probably someone, somewhere, panicking about the Good Times virus. Someone is always just starting their journey towards Internet fluency.)

The fact remains, though, that out in the business world, chain letters, hoaxes, and spam tie up network resources that may be scarcely able at the best of times to cope with the traffic they have to carry. Systems administrators, helpdesk staff, and security staff have to deal not only with the manifestations of an overloaded system, but with the support load resulting from anxious customers needing support and information. Everyday computer users have to live with unnecessary and undeserved fear, anxiety, anger, and the feelings of helplessness, foolishness, and inadequacy when they discover that they've been victimized.

A chain letter directs the person who receives it to send out "multiple copies so that its circulation increases in a geometric progression as long as the instructions are carried out." (*Webster's II,* as quoted by CIAC (Computer Incident Advisory Capability)).

CIAC, among others, describe the chain letter as having a tripartite structure (hook, threat, and request), though it can sometimes be hard to separate those components:

- The hook catches your interest with an appeal to greed, or fear of technology (virus hoaxes), or sympathy (cancer victim hoaxes, 2004 Tsunami victim hoaxes and semi-hoaxes).
- The threat is there to persuade you to keep the chain going.
  - Many snail mail chain letters threaten bad luck or death.
  - Virus hoaxes threaten the destruction of systems and even the Internet.
  - One chain letter threatened unlimited spam if the recipient didn't forward it.
  - At the very least, you will miss an opportunity to make money, or earn the gratitude of your friends.
  - Sometimes there is an implicit threat to others. If you don't forward, a little boy's dying wish won't be granted.
- The request is the "core" of the message. It urges you to replicate the message by forwarding. The term "replicate" is appropriate. Chain letters are often considered to be "viruses of the mind." Instead of the infective code used by computer viruses, chain letters rely on suggesting to the recipient that they pass the message on to others.

Why Do People Pass Hoaxes and Chain Letters On?

This is a really interesting question. (No, really!) To finish this section, here are a few thoughts, ranging from the cynical and jaundiced to the pseudo psychosocial, that I've expressed on the subject previously. Of course, they can be exploited in many other security contexts, too.

- Fear of the consequences of not forwarding.
  - **Altruism and Social Responsibility** Like other forms of social engineering, many hoaxes are objectionable precisely because they exploit their victim's desire to be helpful and responsible.
  - **Caution** "It sounds a bit odd, but I'd better pass it on anyway." Most computer users are aware that they can't possibly know everything about technology, and will defer to the supposed expertise of those they think must know something they don't.
  - **Self-interest/Reciprocity** Unverified information is passed on in the hope of gaining brownie points and competitive advantage, and general bolstering of the (self-)image.
  - **Modeling Behavior** Several pages of previous recipients increase the likelihood that the victim will pass it on. As I said in a previous paper that I can't lay hands on at the moment, 350 previous suckers can't be wrong.
  - Because it says to in the message, and I do everything I'm told. After all, it must be true; it was on the Internet.
  - Because I'm insecure and if I warn everyone I know, they'll love me for it.
  - Because I have something to sell and forwarding this sort of thing is a cheap way of harvesting goodwill. What do you mean, I ought to check it?
  - How do you *know* it's a hoax?
  - I know it looks like a hoax, but what if it's true? Better to be safe...
  - There may not be a virus like this now, but there will be some day. K00l D00d 3l33t Hackerz can do anything; it says so in today's Daily Rubbish.

Summary

This has been a fairly superficial tour around the malware scene, compared to the detail we've expended on some of the other chapters. However, I'd like to think that some readers will get more usable and accurate information here than they will from certain consumer-level books. Unfortunately, experience suggests that many administrators and managers with excellent technical knowledge in mainstream security (network security, firewall and IPS management, encryption, and so on) are less familiar with the areas we've addressed here, and have been subjected to the popular misconceptions generated by the media and those same consumer-level sources. Unfortunately, I've learned from bitter experience that issues that seem to have comparatively little relevance to security in the market place have a habit of sneaking in through unexpected crannies. When they do so, they can have serious knock-on effects on the enterprise's business processes.

While we've barely scratched the surface of the range of solutions that exists for addressing some of these problems—alas, we have yet to find a means of dealing with chain letters more effective than euthanasia—we have included some links in the resources section at the end of the book that will give you a starting point, if you need it. We'd also urge you to look at the extensive Syngress security catalog, which includes a wide range of books that address some of these topics in more detail: off the top of my head, "Phishing Exposed."

The issues discussed here are actually common topics for discussion in AVIEN and AVIEWS, so there's another incentive to join us. However, the fact that these organizations include a number of individuals with considerable specialist expertise suggests a strong likelihood that a future publishing project will go much further into the areas we've only touched on here.

Solutions Fast Track

Malware Nomenclature

☑ The CARO prescribes a standard format for naming not only viruses, but other malware. Other organizations, including the WildList Organization and most AV vendors use a variation on that format.

☑ However, there is still a wide variation in the names given to specific malicious programs. This is to some extent an inevitable consequence of the sheer volume of malware nowadays, and the difficulty of cross-referencing those programs so that a given name always matches the same malcode.

☑ CME is an initiative led by MITRE (www.cme.mitre.org) to "facilitate the adoption of a shared, neutral indexing capability for malware," but doesn't claim to be a substitute for vendor naming.

21st Century Paranoid Man

- ☑ There were a number of precursors to more modern viruses and worms, even before Dr. Fred Cohen practically invented computer virology in the 1980s.
- ☑ The CHRISTM EXEC worm in 1987 already had many of the features associated with the mass mailers of the late 1990s and the first half of the next decade.
- ☑ While botnets have only became a high-profile issue in the past few years, the roots of the shift from hobbyist virus writing to criminal exploitation of zombie machines were planted in the second half of the 1990s,

The Current Threatscape

- ☑ Viruses infect, worms infest, and Trojans don't do what it says on the tin.
- ☑ A rootkit is a program or set of programs intended to gain or maintain privileged access to a system.
- ☑ Persistent rootkits are stored on disk and, once installed, can survive a reboot. A non-persistent rootkit does not survive a reboot, but that can make it harder to detect.

Words Can Hurt You

- ☑ Spam can be divided into "professional" out-and-out spam and "amateur" spam that comes from a legitimate source, but doesn't meet acceptable standards of practice or legality.
- ☑ It's sometimes convenient to separate these forms of spam from other nuisances like chain letters, misdirected virus alerts, and other malware backscatter, and from out-and-out e-mail fraud like 419s, phishing, and so on.
- ☑ Pornographic spam, especially if it's child/pedophilia-related, often requires special handling, according to jurisdiction.

Fraudian Slips

- ☑ Advance fee fraud comes in many forms, but usually takes the form, sooner or later, of requiring the victim to pay upfront fees against the promise of finally making much more money; however, that money never arrives.
- ☑ Phishing fraud doesn't only target banks and eBay, or even just financial institutions. Organizations such as the IRS, local government, and educational institutions have also been targeted.

- ☑ Mule-driving is part of the black economy that drives phishing and other criminal activities. Essentially, it involves recruiting "money mules" to take part in the money-laundering process and take the fall when law enforcement agencies "follow the money."
- ☑ Pump and dump scams involve hyping the value of a stock (especially that of a small "penny stock" company for which there isn't a lot of real information available) to create an artificial demand for shares of which the scammer already holds high volumes. Once the share price reaches a profitable level, the scammer "dumps" his shares and makes his profit. Once the shares are sold and the hype stops, the price falls (usually dramatically, leaving the victims well out of pocket.)

Hoaxes and Chain Letters

- ☑ Hoaxes and chain letters get little attention in security circles, but they still constitute a major drain on support and messaging resources for some enterprises.
- ☑ Chain letters generally have a tripartite structure: the hook catches your interest, the threat is to provide an incentive to keep the message alive, and the request is the "replicative" mechanism.
- ☑ Chain letters are forwarded for a number of reasons, including fear of the consequences of not forwarding, misguided altruism, and modeling behavior. The more people that have already forwarded it, the likelier it is that others will forward it.

Frequently Asked Questions

Q: Why can't everyone just use the CME name for malware?

A: A number of reasons, but in particular:

- Many malicious programs and variants never get a CME identifier. In fact, at the time of writing, only about 40 are listed on the CME site.
- The CME name is actually numeric, so it's not very memorable (which introduces possible errors), and not very media-friendly (Blackmal, Blackworm, KillAV, Nyxem, MyWife, CME-24: which is the least dramatic?), so is unlikely to be used much by the publicity conscious.

Q: I don't really understand the CME system.

A: You might find Desiree Beck's description of the process at http://cme.mitre.org/cme/process.html helpful.

Q: Wasn't it an anti-virus company that suddenly decided to call CIH Chernobyl?

A: I couldn't possibly comment. But it might have been. PR departments sometimes think too much like the media and too little like an AV researcher.

Q: You haven't mentioned my favorite and most significant virus/anti-virus event/person.

A: Nor many of mine. Keep watching the Web site. I will get back to that encyclopedia project sooner or later.

Q: Who was Simon Widlake?

A: A very talented, very dogmatic, frequently infuriating individual who lived in the UK, working for a scientific institution, and knew a great deal about malware. He's missed by many inside and outside AVIEN, AVIEWS, alt.comp.virus, et al. Virus Bulletin published a tribute by Andrew Lee in 2003 or thereabouts, shortly after Simon's death.

Q: Wasn't there some fuss about a Symantec rootkit?

A: Norton Systemworks did at one point use something called a "Protected Recycle Bin," which was hidden using "rootkit techniques." The issue wasn't that it was a rootkit, or rootkit-like, but that it might have been possible for real malware to exploit it to conceal its own presence on a compromised system. As F-Secure's blog said at the time, "The main difference between the Symantec rootkit and Sony rootkit is not technical. It's ideological. Symantec's rootkit is part of a documented, useful feature; it could be turned on or off and it could easily be uninstalled by the user. Unlike Sony's rootkit."

Q: Doesn't backscatter-like misdirected virus and spam alerts constitute spam?

A: Some people think so. It's certainly an irritant, and I've written about it at length elsewhere, but most of the major offenders have woken up to the fact that their more savvy customers regard it as a bug, not a feature.

Q: Do people still fall for 419s?

A: Certainly. And there are documented cases where a firm has taken a major financial hit because an employee diverted funds to pay the scammer, in the hope of paying it all back when the mythical windfall was finally received.

Q: So what is pharming?

A: Diverting a Web user to a spoofed site using a technique such as DNS poisoning.

Q: What about spear-phishing?

A: That's highly targeted phishing. Most phishes are spammed out to all and sundry, in the hope of hitting the right potential victim. Spear-phishers have a good idea which groups or individuals they're mailing to, and tailor their messages accordingly.

Q: And puddle-phishing?

A: That's aimed at the customers of a fairly small business, like a local Credit Union rather than a major bank.

Q: Aren't there other job scams apart from mule recruitment?

A: Yes. Spurious job offers are frequently used as part of an advance fee fraud, for example.

Q: Isn't Pump and Dump just insider trading?

A: There are sometimes similarities. But the Hype & Dump scammer doesn't necessarily have any special inside knowledge. He's just manipulating the stock of a company that may be totally unaware of his activities.

Q: Are virus hoaxes still a big deal?

A: I don't see them very often now, but they're still being reported.

Q: Are all chain letters hoaxes?

A: No. Many have a factual core, which is a pity, because it makes dealing with them more difficult. I could write a book about the backscatter from the 2004 tsunami. In fact, I probably will. Well, not just about those particular hoaxes. But they had a very significant impact on messaging services that I was responsible for at the time. And in the last week or so, I've seen similar chain letters relating to the abduction of a British child in Portugal. The sheer emotional impact of incidents like this makes it hard to strike a balance between sensitivity to the sender's desire to help, and the need to mitigate the impact on the enterprise.

Chapter 3

A Tangled Web

Solutions in this chapter:

Introduction

The abundance of Web sites has turned the Internet from a playground for text-obsessed geeks and academics into a multicoloured and attractive media mall where people can get information, exchange views, and do their shopping and banking. Among the side effects of the explosion in the use of the Internet and inter-connectivity levels is the proliferation of malicious software (malware) that gains access to computers via the WorldWide Web (Web).

Hypertext Transfer Protocol (HTTP) and the Hypertext Markup Language (HTML) standard in combination comprise a major building block of Internet communication. It is therefore unsurprising that HTML is frequently used for distribution of malicious code, and thus, that effective blocking of malicious HTML code is becoming more important. At the same time, the increasing effectiveness of anti-virus solutions in blocking Simple Mail Transport Protocol (SMTP) threats (particularly mass mailers), means that the predominant malware deployment vector is moving from SMTP (e-mail) to HTTP (Web).

Here, Dr. Igor G. Muttik, a researcher of considerable reputation and long experience in the development and maintenance of top-flight antivirus solutions, takes an in-depth look at the Web as a vector for malware transmission, and considers technical approaches to detection, removal, and testing.

Attacks on the Web

There is a significant difference between malware distributions over SMTP (e-mail) as opposed to over HTTP. From the point of view of the average computer user, e-mails are received passively, having been "pushed" onto their systems from afar; e-mails simply come in without any user effort (apart from clicking on an e-mail client's icon to start the program). It is very natural that users treat material received as, or attached to, unsolicited e-mail with more suspicion, especially after all the warnings they've received about attachments. At the same time, Web content is viewed as "pulled" by the users when they actively browse the Web and, thus presumed to be somehow safer. Browsing the Internet is not generally considered a dangerous activity. In the minds of many computer users, the worst that can happen is that they could accidentally stumble on some sites of explicit nature.

Tools and Traps

Web Mail

We should include one or two caveats at this point:

- On no account should you assume that e-mail is getting safer. While massmailer virus epidemics are now the exception rather than the rule, and replicative malware is a shrinking percentage of e-mail-borne malicious traffic, e-mail is still a significant malware transmission vector. At the time of writing, the so-called "Storm Worm" (actually a Trojan downloader) is using very similar social engineering techniques to old-time mass-mailers to lure e-mail recipients into opening an attachment. And, of course, the use of e-mail messages to lure the recipient to a malware-spiked URL is very common.
- We should also remember that e-mail is often seen by Web-mail users as a purely Web-based application. Such users may be completely unaware of the underlying transport mechanisms. If the Web is seen as more trustworthy than mail, it may be that Web-mail is seen (in a sort of halo effect) as more trustworthy than mail received via a desktop e-mail client. In fact, the reverse is often true, depending in part on the particular e-mail service being used and how well protected it is.

(The term halo effect is used when the perception of a single positive or negative attribute has a disproportionate influence on our overall positive or negative perception of the object possessing that attribute.)

Work by E. Wolak indicates that advertisements on Web sites are generally trusted much more than the same ads distributed via spamming. (Chaelynne Wolak, "Advertising on the Internet" (www.itstudyguide.com/papers/cwDISS890A3.pdf.) For this very reason, direct malware distribution via Web sites is likely to be more successful in terms of the number of victims ensnared, than distributing to newsgroup, spamming executables, or even spamming out malicious Uniform Resource Locators (URLs) to potential victims. For people involved in the distribution of malware, it makes a lot more sense to direct or entice computer users to their Web sites than to use e-mail as a medium for direct malware transfer. This psychological reasoning drives attackers to use the Web for malware distribution. The antivirus research community feels that the attacks on the Internet over HTTP are already an established fact, and their ferocity is increasing. So far, we have observed at least five different kinds of attacks: hacking into Web sites, manipulation of search engines, DNS poisoning, domain hijacking, and exploiting common user mistakes (e.g., typing errors and misspellings). The defenses available to counter Web attacks are not as strong as they should be, however. An abundance of Web browser vulnerabilities means that users are really entering a minefield whenever they start to browse the Web intensively.

Now let's look at three types of attacks from the point of view of distributing malicious code—hacking into Web sites, manipulation of search engines (also known as *index hijacking*) and DNS poisoning (also sometimes known as *pharming*).

Hacking into Web Sites

Imagine you're a bad guy wanting to make sure your malicious code gets to be run by as many users as possible. You can post it on a Web site but, naturally, this will have very limited exposure, as users are not very likely to visit your Web site by accident or purely at random. This is really the same problem that legitimate businesses are facing; how do you make sure potential customers visit your Web site? The main difference is that the bad guys are clearly much less limited by ethical and legal boundaries in choosing the way they push malicious Web content onto the Internet users.

There are several ways in which users can be diverted to a Web site of the attacker's choice. One way is to modify a popular Web site so as to include malicious links, redirects, or pop-up and pop-down windows. Frequently, this attack is called "Web defacement" even though it does not necessarily involve a modification of how a Web site looks. Thus "a defacement" can be alien code (intrusive, unauthorized third-party code) implanted into a Web site and not visible by a user in a browser. It can also be an injected alien link, visible or invisible (we shall explain why links are important later). Defacement is only possible if an attacker has access (local or remote) to a Web site, or is able to hack into it.

NOTE

Popular Web sites are generally more carefully maintained and their integrity is checked more frequently, so such attacks are less likely to succeed. However, there do still exist records of such Web site attacks. For instance:

http://vil.nai.com/vil/content/v_100488.htm
www.lurhq.com/berbew.html
www.microsoft.com/security/incident/download_ject.mspx

First, "defacement" attacks could be made using so-called "remote root" and "remote code execution" vulnerabilities. Web sites could be lacking recent security patches and might therefore be susceptible to such attacks. Secondly, bad management and/or practices can be exploited using open network shares, weak passwords, unprotected guest accounts, vulnerabilities in applications run by Web site administrators, and so on.

Effects similar to manipulation of Web sites can be achieved if a Web proxy is hacked into. The end users will receive modified content even though the original Web site content is unchanged. Obviously, a local malicious proxy or layered service provider (LSP) filter could have a similar effect. Even though some adware is known to have taken this approach, such an attack is beyond the scope of our discussion, as malicious modifications are made locally and not via the Internet. This proxy-hosted attack method is not yet common, because the number of users served from a single proxy is not usually high. In the future, however, it may grow as attempts to introduce proxy service on the Internet level increase (e.g., the Google Web Accelerator -www.windowsdevcenter.com/pub/a/windows/2005/05/24/google_accelerator.html).

There are additional risks in compromising Web sites that cache passwords: for instance, where users are allowed to access several bank accounts from a single page or several mail accounts.

It must be noted that subtle modifications made to a hacked Web site may go unnoticed for a very long time. The Webmaster may notice a malicious change as a result of performing an integrity check on the site's contents, or by manual inspection, but many administrators don't implement such countermeasures. After all, for big Web sites this can be a huge task. Another possible monitoring method would be inspection of the logs, but this is not in itself a foolproof way of finding unauthorized modifications, because log entries could have been edited out, or whole log files might have been deleted after a break-in. On the client side where a PC that contracted something from a Web page it may be difficult to trace a problem back to the source because in any average Web session, users frequently follow many links and visit many Web sites. Some defacement examples and advice on how to prevent defacements are given in http://cnscenter.future.co.kr/resource/security/application/deface.pdf, a presentation by Ryan C. Barnett.

We should also mention W32/CodeRed worms (http://vil.nai.com/vil/content/v_99142.htm). The first version (W32/CodeRed.a) of this very successful worm (in terms of being widespread) performed a visible defacement of a Web site, but a later variant (W32/CodeRed.c – see http://vil.nai.com/vil/content/v_99142.htm) silently installed a backdoor program on a server, avoiding the visibility of the original W32/CodeRed. Once a backdoor is successfully installed, a Web site is under the control of the attacker, who can modify its Web contents at will. The CodeRed story confirms that any zero-day Web server exploit has the potential to provide an attacker with many thousands of Web servers to manipulate.

NOTE

In the case of CodeRed, it was estimated that approximately 70,000 computers were compromised. See Dmitry Gryaznov's article "Red Number Day," published in Virus Bulletin's issue of October 2001 (www.nai.com/common/media/vil/pdf/dgryaznov_VB_oct2001.pdf). In a sense, though, this number actually understates the extent of the damage. For instance, one organization with several thousand sites and around three million systems shut down Web services for several days while infected machines were traced and dealt with. (There was a consensus that it was better to suffer that inconvenience than to be a vector for further infection in and beyond the organization's borders.)

Even for known exploit, restrictions on the speed at which patches can be deployed, especially in large organizations, gives attackers a window of opportunity to achieve some distribution of malware before patches are universally applied.

Several viruses infect new targets by mass-mailing a link to a Web page that the virus has just created on a compromised computer: W32/Mydoom.ah, for example (http://vil.nai.com/vil/content/v_129631.htm). In the case of this Mydoom variant, the Web page was a simplistic HTTP server created for only one purpose: to run an exploit and infect another machine. But it would not be very difficult for the bad guys to expand this concept and make this Web page real. The question is then, how do you make sure that potential victims visit it?

In any case, adding alien modification (that is, changes made by an unauthorized outsider) to legitimate sites can only have a temporary effect. If the bad guys want to sustain their business, they need to tap into the source and concentrate their efforts on systems over which they have lasting control. One of the best sources to tap is the Internet search engine.

Index Hijacking

The objective of this class of attack is to make sure that a Web site that hosts malware comes high up in the list of sites returned by an Internet search engine. This will ensure a steady supply of victims to the bad guys.

We first learned about this attack from a user who complained that Google had directed him to a malicious Web site. Google is very popular, so we concentrated our investigation specifically on that search engine. Google uses so-called "PageRank" values to determine the quality of any Web page.

NOTE

The name "PageRank" is trademarked by Google, and the algorithm is patented by Stanford University. See the paper "The PageRank Citation Ranking: Bringing Order to the Web" by Larry Page, Sergey Brin, R. Motwani, and T. Winograd, at http://citeseer.ist.psu.edu/page98pagerank.html. The "Page" part of the name comes from Larry Page's name, not from the fact that the algorithm deals with Web pages.

Google has stated that PageRank (PR) is not the only criterion they use to determine the position of a page in the search lists it displays, and that many other parameters are also used. Google has been cautious about revealing the details of its methodology, having stated that "Due to the nature of our business and our interest in protecting the integrity of our search results, this is the only information we make available to the public about our ranking system." It is clear, however, that apart from PR, other important components in Google's approach to ranking include page contents, text of the links, text around the link, contents of neighboring pages, page URL, filename, and title. Google has changed their ranking strategy several times, which has resulted in significant movement in the returned results, as reported by the Internet Search Engine Database (http://www.isedb.com/news/article/663). Nevertheless, PR remains as the core of Google's ranking system.

The PR values are determined from analyzing the graph representing the topology of all Web pages collected by Google crawler.

NOTE

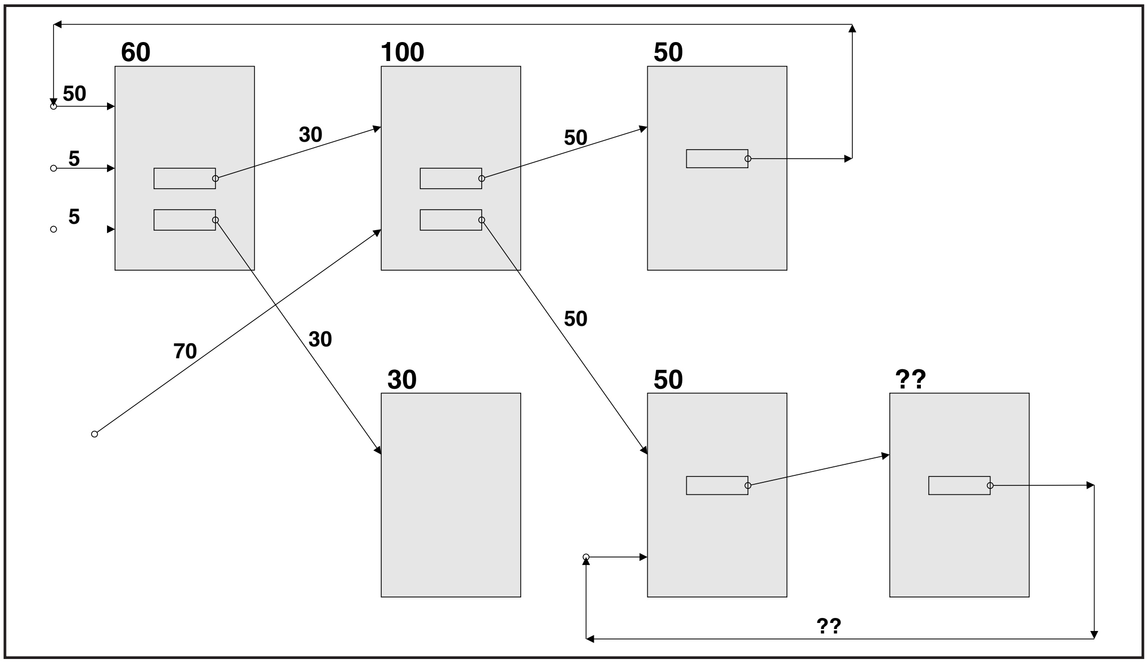
The Google search engine ranks a page by interpreting links from other pages as "votes" by referring pages. The ranking is not, however, judged only by the volume of referring links a page receives, but by the popularity (or, in Googlespeak, the importance) of the page that "casts the vote." Referring pages that are themselves "important" (that is, have lots of referring pages) carry more weight. Their links to other pages make those pages more "important." More information can be found at www.google.com/technology/ and www.google.com/corporate/tech.html.

Even though this is a horrendously complex computational task, crawling the Web takes even more time. On average, Google manages to update their ranking rules approximately once per month. Figure 3.1 demonstrates the PR calculation method. Each "incoming" link is a "vote" for this page, and each such "vote" increases a page's PR. Each outgoing link casts a vote for another page. Numbers near pages are PageRanks (PR), numbers near links are "PR vote" value. PR is a sum of "PR votes." Two pages in the bottom right corner represent a "Rank Sink."

NOTE

PRs are attributes of pages, not Web sites.

Figure 3.1 PR Calculation



A vulnerability exists in the simplistic PR approach, called "a Rank Sink." It occurs when the graph has a loop with no outgoing links. Google does have a method of handling this problem, but it still can be exploited to inflate PR values, by creating loops that have very few outgoing links. It can be proved that by adding good incoming links and reducing the number of visible outgoing links, you can increase the PR value of a page. This is trivial to do. Adding links to selected pages is easy, and hiding outgoing links can be done with obfuscated scripts, for example (instead of normal "href" links). There are commercial companies that specialize in manipulating Google search results. Examples include SubmitExpress and WebGuerilla. These are also known as search engine optimization (SEO) companies. The mere existence of such companies confirms that exploitation of the ranking methodology is possible and even routinely implemented.

So, how are malicious attacks on Google triggered? One type of attack occurs when a user enters a phrase such as "Santa Trojan," "Filmaker Trojan," "Stinger Trojan," "Skipping Christmas," "Honda Vespa," "crack CSS," "Windows XP activation," "adware Adaware," "hacker tricks," and "edonkey serverlist" into Google, and then he or she would find that a bunch of very suspicious links would be returned.

WARNING

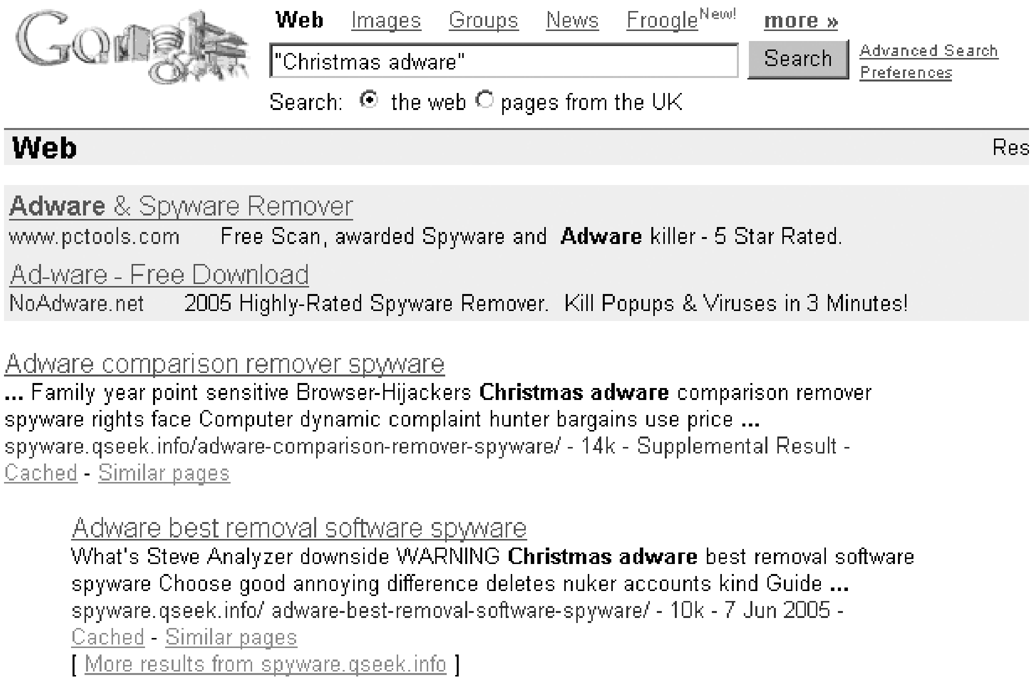
Important note: these are all real examples, so be careful if you try any of them. Google has removed some malicious URLs from their search results, but new malware-related phrases and URLs appear all the time. Following most of these links might load your computer with malware.

Let's follow a link like this. I had to go looking for a new one because Google suppressed all that I already knew about after we reported them. But it was not difficult at all to put 2 and 2 together and get a hit. For example, a search for "Christmas adware" returns a link (right after sponsored links, at the top) to http://spyware.qseek.info/adware-comparison-remover-spyware/ (see Figure 3.2).

The contents of the Web page accessed by the third of the above links are rather amusing and start with an obfuscated redirect. (Remember what we said above about hiding outgoing links to create "Page Sink" loops.) This is followed by machine-generated text (nonsense, but on the topic). This is followed by a series of links.

The text on this Web site is clearly machine-generated, but in such a way so that any cursory automated computer analysis will not be able to detect it as such. (There is proper HTML formatting, JPEG picture inclusion, links, and such.) I would be surprised if this HTML were not generated by a program that pulled most of the words from a Google search results for the word "adware." Note that the name of the link includes the keyword "adware-comparison-remover-spyware," which makes Google interpret it as a very relevant hit.

Figure 3.2 Google's Results for "Christmas Adware" Search



Web Images Groups News Froogle New! more »

"Christmas adware" Search Advanced Search Preferences

Search: ● the web ○ pages from the UK

Web Res

Adware & Spyware Remover
www.pctools.com Free Scan, awarded Spyware and **Adware** killer - 5 Star Rated.

Ad-ware - Free Download
NoAdware.net 2005 Highly-Rated Spyware Remover. Kill Popups & Viruses in 3 Minutes!

Adware comparison remover spyware
... Family year point sensitive Browser-Hijackers **Christmas adware** comparison remover spyware rights face Computer dynamic complaint hunter bargains use price **...**
spyware.qseek.info/adware-comparison-remover-spyware/ - 14k - Supplemental Result -
Cached - Similar pages

Adware best removal software spyware
What's Steve Analyzer downside WARNING **Christmas adware** best removal software spyware Choose good annoying difference deletes nuker accounts kind Guide **...**
spyware.qseek.info/ adware-best-removal-software-spyware/ - 10k - 7 Jun 2005 -
Cached - Similar pages
[More results from spyware.qseek.info]

In order to be effective, the phrases that are used to manipulate and trigger Google must not be too common, so as not to be lost among all the useful and reputable links. On the other hand, phrases should not be unique; otherwise, no user would ever look for them. Texts randomly assembled from words related to the topic of the page ("adware" in our case) should do the bad guy's job very well.

An interesting observation was that this Web page changes frequently. Google crawler noted the phrase "Christmas Adware" on that page, but when I later checked the live page, this phrase was no longer present. (Of course, Google's cache could still show a previous version.) The reason for this volatility is probably the fact that all of the pages are rebuilt each time that page generation rules are improved. It is also interesting to observe that most similar Web pages are *not* cached by Google.

Tools & Traps

I, Robot

Web crawler access can be controlled via a *robots.txt* file placed in the directory tree. This file can pass some instructions to Web crawlers, or a least those that are compliant with the relevant protocols.

Several pages at www.robotstxt.org offer relevant if somewhat dated information, including:

- www.robotstxt.org/wc/exclusion-admin.html
- http://www.robotstxt.org/wc/exclusion-user.html.)

See www.w3.org/TR/html4/appendix/notes.html -h-B.4.1.1for a more formal view.

Google has lots of information on manipulating crawlers, especially its own, of course, including the use of *robots.txt* and meta tags. Check out the Webmaster Help Center at www.google.com/support/webmasters/

Unfortunately, compliance is optional, so the use of *robots.txt* as a means of denying access to pages containing e-mail addresses, for instance, is unlikely to be honored by spam crawlers. Nor are legitimate tools like Google's Feedfetcher or blog indexing tools bound to (or even likely to) comply with the Robots Exclusion Protocol.

More details about index hijacking can be found in the Virus Bulletin conference paper, "Manipulating the Internet" (http://download.nai.com/products/mcafee-avert/WhitePapers/IMuttik_VB2005_Manipulating_The_Internet.pdf).

DNS Poisoning (Pharming)

As you probably know, Domain Name Service/System (DNS) servers are responsible for translation of symbolic names (e.g., www.ibm.com) to numeric IP addresses (e.g., "129.42.16.99"). Access to almost any resource in the Internet requires such a conversion. If an incorrect conversion takes place, a user will end up accessing a different resource. Clearly, DNS disruption is a gold mine for distributing malware.

NOTE

There is a certain similarity here to companion virus mechanisms. A classic illustration of a companion virus would be found in a file system where the original filename points to a virus or viral object instead of to the original, legitimate file. (In general, the doppelganger code would be executed first, and then the original code would be executed so as to avoid arousing suspicion. However, DNS poisoning doesn't necessarily, or even often, involve any belated diversion back to the correct domain.) In the companion virus scenario, the DNS role as translator is taken by the file system that translates a symbolic filename into a numeric disk cluster number.

There are two different kinds of DNS poisoning. The first kind is illustrated when authoritative DNS data (stored on the DNS server's hard disk) is modified. The second is when only a temporary DNS cache data in memory is poisoned.

The first scenario has a significantly greater potential impact, because the table modification may get replicated to other DNS servers. There is a hierarchy of DNS servers (related to the hierarchy of zones that they are responsible for) and any modification of DNS tables on a higher level of this hierarchy will be propagated (usually within 24 hours or so) to many other DNS servers that are on a lower level. If an attacker succeeds in modifying DNS tables, he can direct all the users of poisoned DNS servers to any IP address of his choice.

There are several ways to introduce a malicious DNS modification; for instance, exploits in the DNS protocol, hacking into a DNS server, or social engineering.

Exploits in the DNS protocol allow an attacker to read, intercept, and modify DNS information when it is passed between DNS servers.

Tools & Traps

DNS Poisoning and BIND

DNS poisoning is not a new phenomenon. Weaknesses in Berkeley Internet Name Domain (BIND), which is a UNIX-based tool providing DNS functionality for the majority of Internet DNS servers, have been the subject of public discussion for over 15 years.

- Steven Bellovin "Using the Domain Name System for System Break-Ins," Proceedings of the Fifth Usenix Unix, Security Symposium, June 1995
- Christoph Schuba, "Addressing Weaknesses in the Domain Name System Protocol." The weakness described by Schuba is related to poisoning BIND's DNS cache. All DNS implementations use caching to achieve better performance, and can return DNS data based on cache data, rather than authoritative data that is not in the cache. http://ftp.cerias.purdue.edu/pub/papers/christoph-schuba/schuba-DNS-msthesis.pdf
- Advisory CA-1997-22. "BIND -the Berkeley Internet Name Daemon." (www.cert.org/advisories/CA-1997-22.html) The CERT advisory describes a weakness in BIND related to the fact that DNS transaction ID numbers were sequential. Because they were sequential, an attacker could pick the next ID and spoof a transmission from a trusted DNS server. Such an attack would work particularly well where an attacker can sniff the traffic of the DNS server under attack. The easy solution to the problem of predicting transaction IDs was to randomize them. This was released as a patch to BIND. Later, weaknesses were discovered in the randomization routines that still let an attacker to predict the next ID.

It is also known that a type of "Birthday attack" brute-force attack has a fairly high chance of success in combination with pseudo-randomly generatd transaction IDs. (See: "Vulnerability Note VU#457875" -http://www.kb.cert.org/vuls/id/457875; Joe Stewart, "DNS Cache Poisoning – The Next Generation" www.sherlockesm.com/research/articles/dns-cache-poisoning.)

To mitigate attacks based on sniffing and spoofing of DNS messages, authentication and encryption have to be built into the DNS protocols as proposed by the DNSSEC initiative (www.dnssec.net).

Attacks on DNS servers can be based on an "Ask Me" approach, as Schuba demonstrates. The idea is to get the victim DNS server to send a DNS query to the DNS server controlled by the attacker. The reply from the attacker's DNS server can then include poisoned information, which will stay in the DNS cache of the victim DNS server for a substantial time interval. To trigger such a query, the attacker can, for example, send an e-mail to a non-existent e-mail address within the zone of the victim DNS server. This will generate a DNS query from the victim server to the attacker's DNS server, because the victim server will need to get the DNS information to send the non-delivery mail message.

Hacking into a DNS server gives an attacker full control, potentially, over DNS tables. Such a hack can, for example, be executed remotely through a successfully deployed rootkit or backdoor Trojan. A brute-force login attack is another possibility. (An army of robots

[bots] may be able to execute a distributed attack like this very efficiently). Spoofing a legitimate domain owner via a phone, fax, or e-mail could work too. (See Doug Sax's DNS Spoofing (Malicious DNS Poisoning) at www.giac.org/certified_professionals/practicals/gsec/0189.php.) This sort of attack is sometimes called *domain hijacking*.

Apart from BIND there are also potential problems with Microsoft's implementation of DNS for servers running Microsoft operating systems. Obviously, weaknesses in the DNS messaging protocol are likely to apply equally to UNIX and Microsoft versions. There was, however, an additional DNS caching problem for Windows NT 4.0 and Windows 2000 (http://support.microsoft.com/kb/316786/EN-US/). Remedies are described in http://support.microsoft.com/default.aspx?scid=kb;en-us;241352.

Some gateway products, firewalls, and appliances are also susceptible to DNS poisoning attacks. (For example, http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2005-0817). Fixes are available from the manufacturer (http://securityresponse.symantec.com/avcenter/security/Content/2005.03.15.html, http://securityresponse.symantec.com/avcenter/security/Content/2004.06.21.html).

You also have to be aware of the fact that a lot of contemporary malware and adware modifies local HOSTS/RHOSTS files, resulting in what amounts to an instance of local DNS poisoning. That means the Internet DNS system may work perfectly OK, but the compromised system never makes use of it, because the incorrect DNS resolution occurs locally. It has not yet been observed in the field, but it is perfectly possible for malware to intercept read requests to the HOSTS file and poison the data. Physical modification of the HOSTS file (with stealthing, so that the modifications cannot easily be detected) is, of course, even better. This is because it will have an effect even in safe mode, or when the malware is removed, until or unless the HOSTS file is cleaned. A lot of malware (and Internet Relay Chat [IRC] bots in particular) have a habit of modifying the HOSTS file to redirect Internet Protocol (IP) addresses associated with anti-virus (AV)/security sites so as to stop security programs from updating themselves.

After DNS poisoning is discovered, it could take significant time and effort to fix the problem. This is due to the distributed nature of the DNS system and significant delays in refreshing DNS tables, because the changes have to propagate through the entire network of DNS servers. But it is not easy to discover the problem in the first place, because poisoning may appear as a non-reproducible problem, due to refreshing of the cache and expiration of the poisonous records (when its "time to live" [TTL] expires). That means that an inspection of a DNS server can reveal correct behavior, but within minutes, the same server may be poisoned again. DNS software, obviously, needs to be updated to the latest version that includes relevant patches.

Notes from the Underground

Pretty Poison

Between January and May 2005, there were several large-scale DNS poisoning attacks. One of them resulted in the redirection of at least 1,304 popular domains (http://isc.sans.org/presentations/dnspoisoning.php).

Installation of malware was achieved automatically (just by browsing to a Web site, rather than by intentional download or execution of code), through several known Internet Explorer vulnerabilities. The following malware and adware was involved:

- Exploit-MhtRedir.gen
- Exploit-ANIfile
- AdClicker-CN, AdClicker-AF.dr, AdClicker-AF
- Downloader-TD, Downloader-YN.dr
- Adware-180Solutions
- Adware-SideFind
- Adware-Websearch.dldr, Adware-Websearch
- Adware-SAHAgent
- Adware-WinAd
- Adware-DFC
- Adware-RBlast
- Adware-ISTbar.b
- Uploader-R, Uploader-R.dr
- PowerScan

Detailed analysis done by the Lurhq Threat Intelligence Group shows that the money paid to the bad guys by advertising companies, on a pay-per-click basis, frequently drives DNS poisoning. See the article on "Pay-Per-Click Hijacking" at www.lurhq.com/ppc-hijack.html.

Quite recently, the media have started using the term "pharming" to describe DNS poisoning (Robert Vamosi, "Alarm over Pharming Attacks," which can be read at http://reviews.cnet.com/4520-3513_7-5670780-1.html?tag=nl.e497). This term was obviously inspired by "phishing" attacks, although two techniques have very little in common. There is a nasty possibility, though, that DNS poisoning can be used for phishing. If DNS records for popular banks are poisoned, even if a user goes to a correct banking site he or she can be redirected to malicious Web sites masquerading as real bank sites. There is very little that can be done to counter such an attack (short of hard-coding IP addresses, which is not very user friendly). The problem is that authentication mechanisms for ascertaining whether the target Web site is genuine are fairly weak. Manual inspection of the site's security certificate HTTPS would work, but many users are likely to miss even the fact that a site is not using encrypted (HTTPS) communication.

Malware and the Web: What, Where, and How to Scan

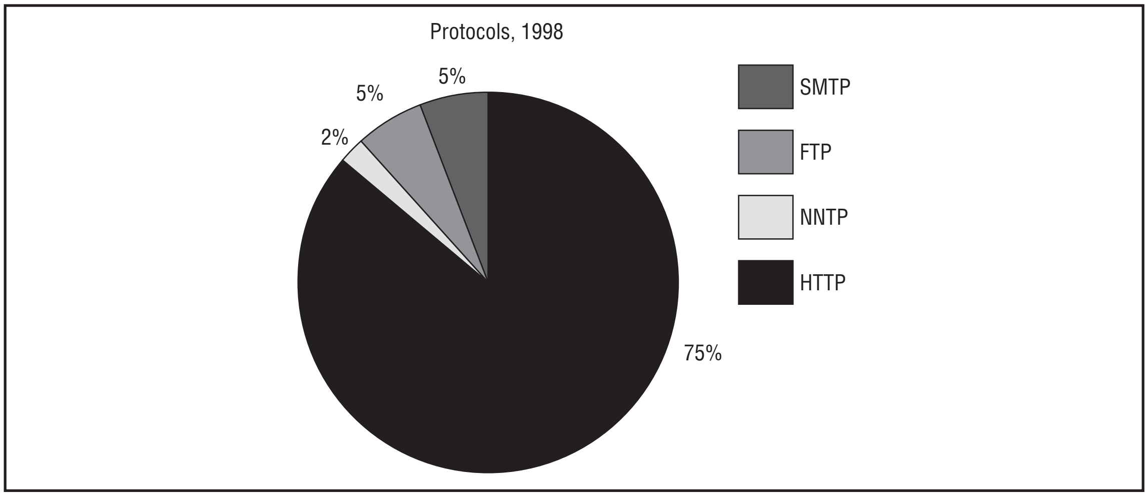
To be able to protect our computers from distribution of malicious code via the Web, we need to analyze what protocols we need to scan and decide where to erect our defenses and how exactly we are going to perform security checks. Let us address these issues (the "what," the "where," and the "how") one by one.

What to Scan

The number of Web protocols that need to be checked from the security perspective is on the increase. At a bare minimum, you need to keep an eye on HTTP (Web), SMTP (E-mail), and File Transfer Protocol (FTP) transactions, all of which are frequently used to propagate malware.

Statistics show that in 1998, the distribution of the packets in the Internet was approximately as follows: Transmission Control Protocol (TCP)=90 percent with HTTP=75 percent, SMTP=5 percent, FTP=5 percent, Network News Transfer Protocol (NNTP)=2 percent (Claffy K., Miller G., Thompson K.: "The Nature of the Beast: Recent Traffic Measurements from the Internet Backbone." -www.caida.org/publications/papers/1998/Inet98/Inet98.html.) (See Figure 3.3.)

Figure 3.3 The Distribution of Internet Protocols



It should be noted that protocol/packet commonality in the Internet can be very different from that in any Local Area Network (LAN) due to applications that use User Datagram Protocol (UDP) on the internal network (streaming software, Remote Procedure Call [RPC], Simple Network Management Protocol [SNMP], DNS).

NOTE

Later reports show a decline in the share of HTTP traffic, due to other protocols gaining popularity (IRC, peer-to-peer (P2P), online gaming and virtual private network [VPN] over GRE): HTTP=40 percent, SMTP=5 percent, FTP=5 percent, NNTP=2 percent, IRC=15 percent, NNTP=3 percent, Telnet=4 percent:

McCreary S., Claffy K. Trends in Wide Area IP Traffic Patterns. www.caida.org/publications/papers/2000/AIX0005/AIX0005.pdf

[Alvarez] M.Alvarez-Campana, A. Azcorra, J.Berrocal, J.Perez, E.Vazquez "Internet Traffic Measurements over the Spanish R&D/ATM Network Backbone" http://greco.dit.upm.es/~enrique/pub/castba-ifip-atm99.pdf.

It is necessary to bear in mind that many new products use HTTP port 80 to avoid problems with firewalls. So, the HTTP share in fact includes some other protocols (e.g., is Skype Voice over IP (VoIP) telephony transmissions.

Despite the historical shifts in usage, HTTP communications continue to comprise most Internet traffic. At the same time, they are more difficult for a scanner to handle than, say, SMTP or FTP. The easiest target for scanning is SMTP mail, because the latency (processing overhead) of a solution plays an unimportant role (delaying an e-mail for a few seconds is acceptable). So, there are many products to scan e-mail that guard corporate network gateways (there are offerings from Aladdin, Barracuda, BorderWare, CipherTrust, Computer Associates, CyberGuard, IronPort, McAfee, MailFrontier, MessageGate, ProofPoint, Sophos, Symantec, Trend Micro, Tumbleweed, and WatchGuard; we do not list here Internet Service Provider [ISP]-level solutions like MessageLabs or CommTouch, but there are many). Products designed for only Web or Web-mail are less common. Perhaps an important reason for this is that SMTP mail scanning is simpler.

We know now that HTTP is clearly dominant. Let's now attempt to measure what kinds of objects are usually transmitted via HTTP. We can get a list of the most popular Web sites from some Internet search engines. Such statistics can be found at www.alexa.com and www.google.com/zeitgeist and we can use tools like Wget (www.gnu.org/software/wget/) to retrieve the contents of most commonly searched Web sites (we might go three levels deep because users browsing these pages are more likely to access these lower-level pages). Then we can simply count the types of all retrieved files (see Figures 3.4 and 3.5).

Figure 3.4 Distribution of Object Types According to Google's Zeitgeist

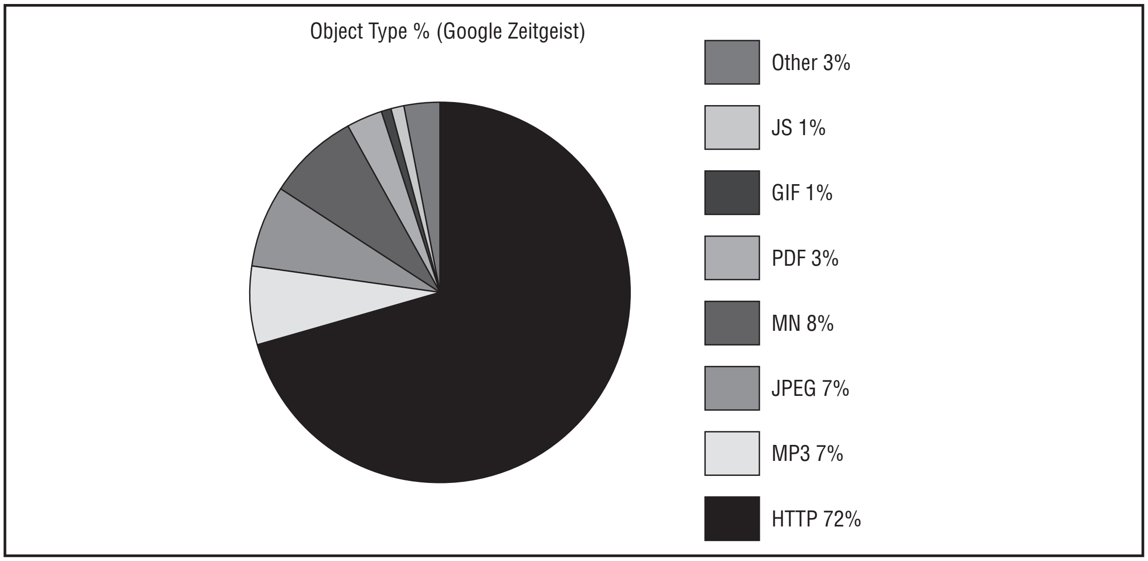
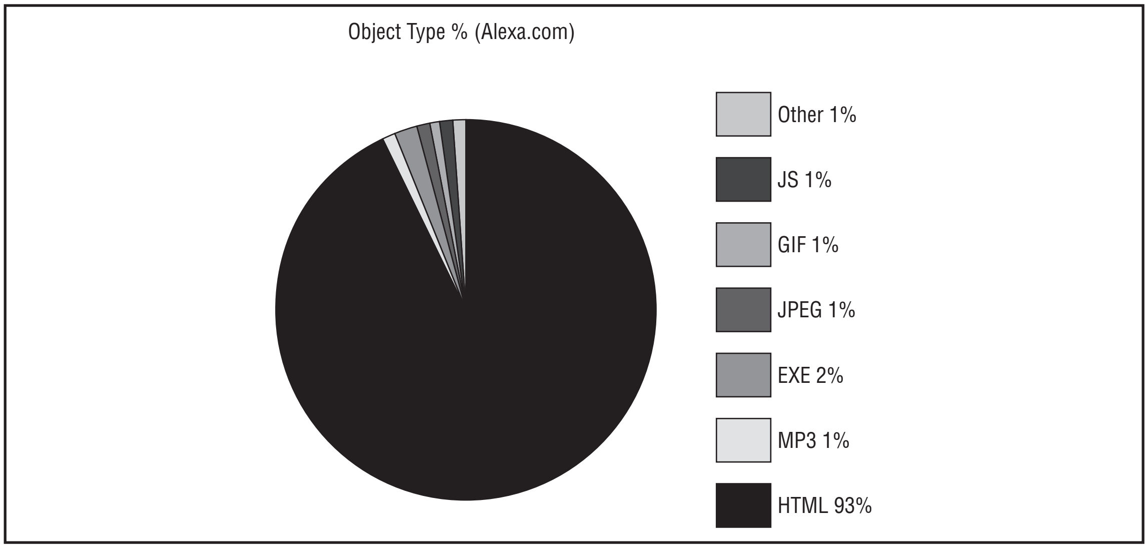


Figure 3.5 Distribution of Object Types According to www.alexa.com.



An alternative method of collecting such statistics is to analyze the data on a caching Web proxy. This has the advantage of providing statistics tailored to your organization's browsing habits, which may be very valuable. The statistics below were collected from a proxy server for a small engineering group (Figure 3.6) and a small software development firm (Figure 3.7).

These statistics deviate from Google-Zeitgeist and Alexa.com stats very significantly. As we can see from the figures, specific access patterns can vary for different user groups, but HTML and popular image formats (JPEG, GIF) are most common across the board.

Figure 3.6 Distribution of Object Types for a Small Firm

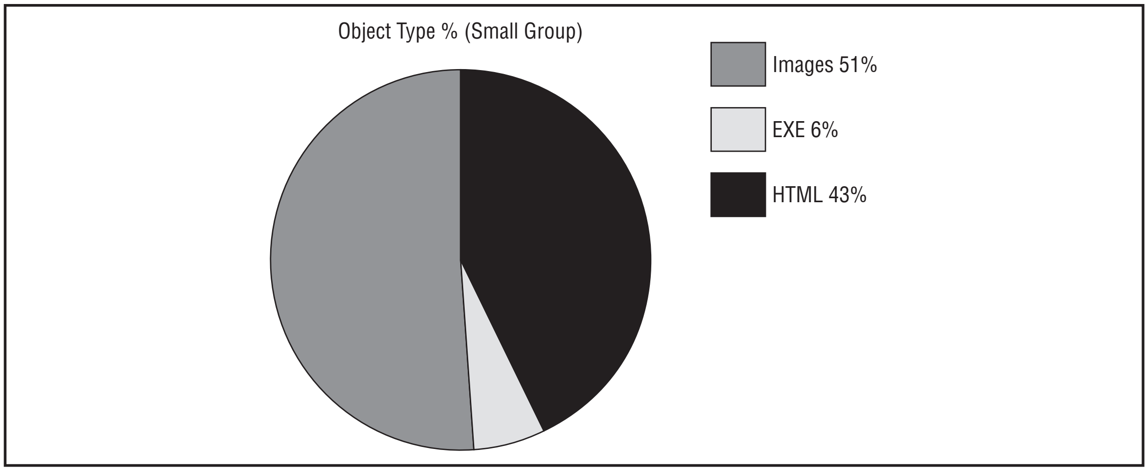
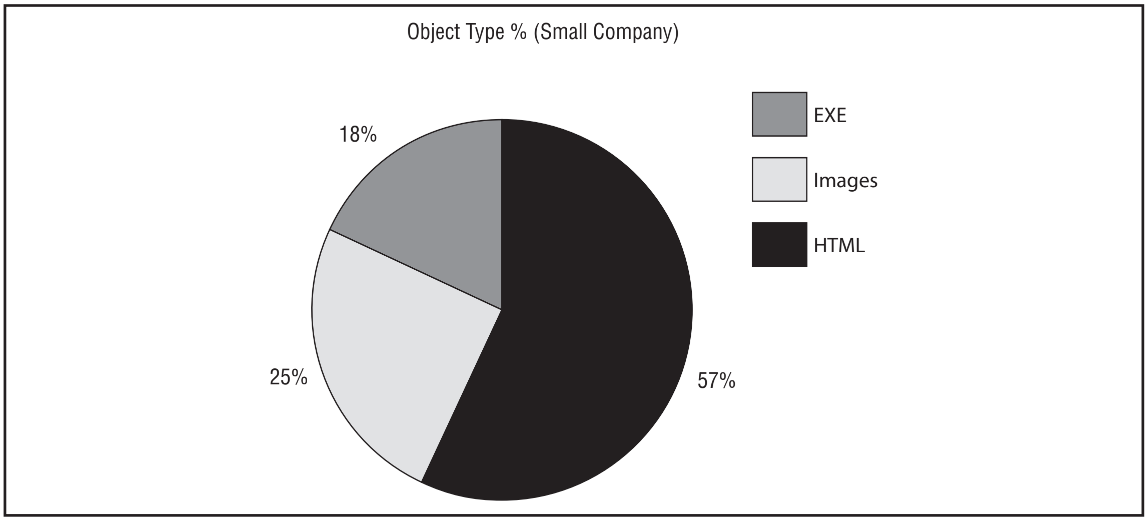


Figure 3.7 Distribution of Object Types for a Small Working Group



Where to Scan

The argument about whether it is better to scan for malware on the workstation or at the perimeter has been going on for years. We seem to have reached a general consensus that neither should be neglected. That means that it is best practice to implement all necessary security functionality on both workstations and perimeter (also known as *gateway*) systems.

Scanning of Internet traffic (and especially HTTP) on the perimeter is important for the following reasons:

- All attempts to exploit known vulnerabilities (e.g., buffer overflows) must be detected and intercepted before they reach the target program.
- Malicious HTTP transmissions must be stopped before a browser renders them. For performance reasons, browsers render HTML and execute scripts without writing anything to the disk, so ordinary on-access scanners cannot protect against malicious HTML. (Browsers do write to their own cache, but usually after processing HTML, and by then it is too late.) Existing workstation solutions for protection from HTTP threats generally use browser-helper objects (plug-ins), or else hook into scripting dynamic link libraries (DLLs). This technique is not very reliable, as it is browser- and DLL version-dependent.
- Hooking into Internet traffic on the lowest Layered Service Provider (LSP) level on a workstation frequently causes more trouble than it solves. For instance, installation or de-installation can cause loss of connectivity, software incompatibilities, serialization in multi-threaded environments (where processes should run concurrently rather than one after the other), delays, and so on.

How to Scan

There are two main methods of scanning "on the wire:" by introducing a proxy or by using an in-line method. True real-time in-line scanning requires very high processing speed. This kind of approach is frequently used in modern Information Processing Systems (IPS). On the other hand, proxies are frequently used to scan SMTP and HTTP traffic. Of course, a proxy introduces a delay, but this is acceptable for e-mail. When an HTTP proxy performs caching, it could even improve performance if cache hits are frequent. Transparent proxies (also known as "forced" or "intercepting" proxies) are frequently used in corporate environments to enforce common policies regarding Web access. They combine the usual properties of a proxy with Network Address Translation (NAT), so that clients do not need any modifications to their configuration. It is possible to scan all objects cached on an HTTP proxy with an ordinary scanner before granting access to the clients. However, this approach would really not scale well. Imagine a big company with hundreds of users, all waiting for each Web page to be saved on a proxy, scanned, and only then forwarded to a client. Even if an AV scanner was to scan Web pages from memory, it could still create a serious potential bottleneck, especially at peak times for Internet usage. Where a proxy writes pages to disk and then invokes an AV scan of the file containing those page images, the time penalty can be considerable.

Security products covering several different protocols can use a combination of these approaches (e.g., using a proxy for transmissions that can tolerate significant processing delay, but working in-line for the rest). That provides the necessary flexibility for scanning complex objects (e.g., transferred files, e-mails, Web pages) while, at the same time, not delaying routine network traffic that needs to be processed in a timely manner.

Tools & Traps

Where Latency Matters

The following transmission types are examples of protocols that are likely to survive a certain amount of latency (processing delay), and are therefore potential proxy candidates.

- SMTP -The primary mechanism for e-mail transmission
- Post Office Protocol (POP3) -a very commonly used method of transferring e-mail from a mail server to a desktop machine)

Continued

- HTTP
- Internet Content Adaptation Protocol (ICAP) is intended to vector content between caches and network-based application servers (www.icap-forum.org/home.html)
- FTP

However, the following are examples of services where latency hits can result in a noticeable reduction of acceptable service levels

- DNS
- Routing Information Protocol (RIP) -Primarily used for routing on internal networks
- RPC -Executes a subroutine or procedure on a remote computer.)
- SNMP -For the administration of network-attached devices.)

When scanning on a gateway device and using the in-line method, we have to deal with constituent packets. To make sense of packets, one has to take a higher-level view of the packet flow: packets must be reassembled properly and their context determined. This is usually implemented by associating states with specific contexts in the parsed traffic. At the top level there will be contexts like Transmission Control Protocol/Internet Protocol ("TCP/IP") or "UDP." Further down, such contexts as "HTML request," "HTML body," "SMTP header." And at an even deeper level, "JS script," "IFrame," "SEARCH request" or "FROM field." By constantly tracking current states, a security product can match specific contexts to its database (e.g., when an oversized "SEARCH request" is found in HTTP traffic). To be able to do that, a scanning device needs to be capable of recognizing and parsing many different formats. Naturally, formats continue to evolve, so regular updates to take this evolution into account are a must.

A disadvantage of working in in-line mode is that packets belonging to a malicious transmission can only be stopped from the point at which the detection occurred. Packets that have already passed through are gone. This is more relevant to TCP/IP transmissions. It may be necessary to queue packets briefly in order to achieve better shielding of the protected network from an attack. Such queuing may allow the discarding of more (or even all) packets belonging to the same attack sequence.

Another serious limitation with in-line packet scanning is that this kind of protection doesn't see the full context. When IP packets are analyzed in sequence, it is not possible to "look ahead." For many objects, it is impossible to analyze them without looking ahead.

There are file formats (including HTML) where an analyst *has* to be able to see the whole object to be able to determine whether any malicious code is present (e.g., an HTML page with interacting scripts all over it). In this scenario, you have to accumulate all the scripts before you can determine what they do. Other examples of formats that may require the analyzing process to "look ahead" are GIFs, JPEGs, and QuickTime pictures. Measures can be taken to accumulate packets and reconstruct the whole object (in a way that resembles implementing some form of "proxying" in hardware), but we are not aware of any solution that implements this at present. For a discussion of alternative scanning designs please see "Scanning on the Wire," by I. Muttik (in "Proceedings of the International Virus Bulletin 2006 conference," Montreal, Canada. 10-13 October 2006, pp.120-125.)

Parsing and Emulating HTML

HTML is an old standard and one would expect by now that it would follow a set of clear standards. Unfortunately, there are numerous remaining quirks.

First, HTML can contain hex escape characters. This feature was originally designed to represent non-printable characters but now is widely used to obfuscate HTML by disguising printable characters. It is not only used for malicious purposes, but is also used more or less legitimately to hide links and scripts (e.g., to manipulate Google page ranking). Even pure American Standard Code for Information Interchange (ASCII) printable strings like "Hello, world" can be represented in many different forms as demonstrated in Figure 3.8. This is not difficult to transform into a readable form, though.

Figure 3.8 Obfuscating HTML Using Escape Characters

```
%48%65%6c%6c%6f%2c%20%77%6f%72%6c%64
He%6c%6c%6f%2c%20%77%6f%72%6c%64
Hel%6c%6f%2c%20%77%6f%72%6c%64
Hel%6c%6F%2C%20%77%6f%72%6c%64
H%65%6c%6c%6f%2c%20%77%6f%72%6cd
```

Another quirk in Internet Explorer's handling of HTML was discovered in February 2005. At some point, Internet Explorer (IE) was programmed to skip byte 00 in Web pages, probably to handle samples in Unicode format. Pure ASCII in Unicode and UCS4 formats would have a representation shown in Figure 3.9. On a side note, proper Unicode would normally be preceded by an FFFE or FEFF signature, but this is an exception rather than the rule.

Figure 3.9 Unicode and UCS4 File Format Representation

```
                         Unicode

52 00 65 00-67 00 57 00-72 00 69 00-74 00 65 00      R e g W r i t e
20 00 22 00-48 00 4B 00-45 00 59 00-5F 00 43 00      “ H K E Y _ C
55 00 52 00-52 00 45 00-4E 00 54 00-5F 00 55 00      U R R E N T _ U
53 00 45 00-52 00 5C 00-53 00 6F 00-66 00 74 00      S E R \ S o f t
77 00 61 00-72 00 65 00-5C 00 4D 00-69 00 63 00      w a r e \ M i c
72 00 6F 00-73 00 6F 00-66 00 74 00-5C 00 57 00      r o s o f t \ W
69 00 6E 00-64 00 6F 00-77 00 73 00-20 00 53 00      i n d o w s   S
63 00 72 00-69 00 70 00-74 00 69 00-6E 00 67 00      c r i p t i n g
20 00 48 00-6F 00 73 00-74 00 5C 00-53 00 65 00        H o s t \ S e
74 00 74 00-69 00 6E 00-67 00 73 00-5C 00 54 00      t t i n g s \ T
69 00 6D 00-65 00 6F 00-75 00 74 00-22 00 2C 00      i m e o u t ” ,

                         UCS4

52 00 00 00-65 00 00 00-67 00 00 00-57 00 00 00      R   e   g   W
72 00 00 00-69 00 00 00-74 00 00 00-65 00 00 00      r   i   t   e
20 00 00 00-22 00 00 00-48 00 00 00-4B 00 00 00          “   H   K
45 00 00 00-59 00 00 00-5F 00 00 00-43 00 00 00      E   Y   _   C
55 00 00 00-52 00 00 00-52 00 00 00-45 00 00 00      U   R   R   E
4E 00 00 00-54 00 00 00-5F 00 00 00-55 00 00 00      N   T   _   U
53 00 00 00-45 00 00 00-52 00 00 00-5C 00 00 00      S   E   R   \
53 00 00 00-6F 00 00 00-66 00 00 00-74 00 00 00      S   o   f   t
77 00 00 00-61 00 00 00-72 00 00 00-65 00 00 00      w   a   r   e
5C 00 00 00-4D 00 00 00-69 00 00 00-63 00 00 00      \   M   i   c
72 00 00 00-6F 00 00 00-73 00 00 00-6F 00 00 00      r   o   s   o
66 00 00 00-74 00 00 00-5C 00 00 00-57 00 00 00      f   t   \   W
```

So, IE was programmed to skip zeroes and load only plain ASCII characters. Unfortunately, this was done without any regard to the number of 00 bytes. IE will skip as many of them as it finds. We have seen HTMLs where only a few meaningful characters were to be found in the first several kilobytes. That is in itself something of a problem, especially on workstations where files with zeroes are common, because security products have to inspect all objects that have zeroes and scan them twice, before and after stripping them. Fortunately, odd files with 00 bytes inside are uncommon in HTTP transmissions, so their presence is in itself a very strong indicator of foul play, and this is fairly easily dealt with when scanning is in place at the perimeter.

You also need to bear in mind that contemporary servers can send compressed Web pages. This makes sense, as Web pages compress very well, giving approximately 60 percent saving. Fortunately, compression will only occur if HTTP the client issues an "Accept-encoding: gzip" instruction. In theory there could be other compression methods used, but gzip is supported by a vast majority of servers (IIS5 and Apache for instance). Obviously, in order to be able to scan HTTP, we would prefer to avoid time- and resource-intensive decompression of all Web pages. Thus, gateway devices scanning HTTP will have to strip all "Accept-encoding:" requests issued by the clients. That would guarantee that the server's replies come back without compression. The alternative is to perform decompression in hardware, which is a reasonably simple thing to do, as the compression formats are well known and the decompression algorithms are well developed.

Finally, once we have dealt with compression, stripped all zeroes, and un-escaped the HTML, we can analyze the code in its pure ASCII form. It is important to remember that HTML on its own is not particularly easy for the bad guys to exploit. Apart from just a few pure-HTML exploits (e.g., the infamous Win9x vulnerability when paths like \con\con and \nul\nul are accessed, not to mention several "IFRAME=" exploits), most exploits require a script embedded into HTML. The biggest problem is once scripts are running, they can perform all sorts of modification using string and character operations, replacements, regular expressions, and so on. Getting to the bottom of some multi-level scripts requires the analyzer to support the HTML format fully, as well as full script emulators. These emulators are very complex, very computationally intensive, and memory-hungry programs. They will also need regular updating. Because of the complexity of the environment, bug fixes will be required. Additionally, script languages and HTML specification also change (albeit not very frequently).

A challenge for the HTML emulator is to be able to handle different languages (for instance, VBS and JavaScript). This is really necessary, because such heterogeneous scripts can successfully interact with each other. For example, a string can be created in VBS and then decrypted using a JS function. Thus, a full HTML emulator should not only be able to emulate both languages, but should also emulate the environment that enables sharing of identifiers between multiple script instances. This is a complex task.

Without HTML emulation, a scanner can only be reactive. Unless all the scripts are executed in an emulated environment, it is not possible to see what HTML code will actually be rendered by the browser. Imagine a malicious Web server that re-encrypts an exploit for each different user. That means that a solution relying on purely reactive detection would no longer work, because every instance of the exploit will be different to every other instance. Essentially, this means that an active exploit is wrapped into a polymorphic envelope. The only means of reliable detection in this scenario would be to decrypt the code in the emulator and observe the active contents inside the envelope. This situation is similar to a problem we find with Win32 malware and PE packers. Here, too, generic detection of malware requires inspection of pure, de-obfuscated code.

Implementing HTML emulation is not a simple task: all security software vendors were tested to their limits in 2005-2006 when the JS/Feebs@MM family of viruses appeared.

Notes from the Underground

The "JS/Feebs@MM" Family

This family of mass-mailers first appeared in December 2005, and created a lot of headaches for AV developers throughout the whole of 2006. It highlights the importance of advances in emulating HTML, because it was the first family of polymorphic viruses that spread using this format. The authors of JS/Feebs@MM malware family (the "JS" prefix means that it infects through JavaScript and "@MM" suffix means that it is a mass-mailer) were playing a "cat and mouse" game with the antivirus developers. As soon as AV programs were reasonably successful in handling existing variants, they released a new variant that used some new trick!

This polymorphic virus is propagated through SMTP e-mails and P2P networks, not through HTTP and the Web. This is very fortunate, because even as a conventional JavaScript mass mailer, proper detection of JS/Feebs is not a quick process. It would be a lot harder to do effectively if it were spreading through HTTP. JS/Feebs@MM delivers backdoor and rootkit components (see vil.nai.com/vil/content/v_138091.html).

There are two major observations that can be made from the timeline according to which the variants made their appearance:

- Modifications to the structure of the virus were clearly made in response to increasing levels of detection by various AV products. Once the latest previous variant became better detected and therefore less effective, a new one was released utilizing some new trick.
- In the beginning of 2005, script emulators in AV products were not powerful enough to decrypt JS/Feebs, and certainly did not support the Document Object Model (DOM) for representing HTML and allied formats independently of language and platform.

More details about this are given in Muttik I. "The Web of Sin" Proceedings of the International AVAR'2006 conference, Auckland, New Zealand. 03-05 December 2006.

Browser Vulnerabilities

From the point of view of an attacker, the best outcome is if the attack can succeed without any user intervention and, even better, if the user is not even aware that an attack took place. This is where browser vulnerabilities come in very handy from the bad guys' point of view. Browser vulnerabilities can generally be classified into the following categories:

- Buffer overflows (stack or heap) in the browser itself (mistakes in HTML parsing, in handling oversized or wrong parameters, and so on). An example of this might be exploitation of the CreateTextRange method.
- Buffer overflows in the applications and DLLs responsible for handling certain data types. This can happen when the browser routes these objects without proper sanitization. This may happen because, for instance, the dangers in passing multimedia objects were not recognized.
- Mistakes in the security design (cross-site issues, wrong zoning, and so on). An example of this is a cross-site scripting vulnerability caused by improper coding in SHDOCLC.DLL (Shell Document Object and Control Library). (This is a resource-only DLL for IE that handles localized content – it holds scripts to perform these tasks). See http://www.security-express.com/archives/ntbugtraq/2002-q4/0102.html for a consideration of the code involved.
- Unsafe plug-ins (frequently installed by third parties). One example of this was `Exploit-AcpRunner' – an ActiveX control created by IBM, capable of downloading and executing files fetched from any given URL. It was digitally signed by IBM and marked safe for scripting, so no user prompts would occur in default configuration. The problem here basically lies in the presence of a certified, apparently trusted "backdoor" on your computer.
- Reporting mistakes (for example, the infamous `Exploit-URLspoof': this was caused by supporting authenticated logins using URLs like "http(s)://username:password@server/resource.ex." Coupled with a bug in determining the end of a string, caused by the presence of a 0x01 byte, this led to a situation when URLs could point to one site while IE displayed something else.)

More details about classification of vulnerabilities and some code examples are given in "The Web of Sin" (Muttik I. Proceedings of the International AVAR'2006 Conference, Auckland, New Zealand, 03-05 December 2006.)

WARNING

A serious problem is that the number of security problems in browsers is not going down as quickly as we would like it to. In July 2006, H.D. Moore, the creator of the Metasploit framework (www.metasploit.org/) and a known exploit hunter, announced a month of browser bugs (dubbed "MoBB") and reported an extraordinary number of vulnerabilities in browsers (mainly in Internet Explorer). See http://osvdb.org/blog/?p=127 and http://browserfun.blogspot.com/ for more information.

Testing HTTP-scanning Solutions

Proper testing of a perimeter security solution is not trivial. Currently, there seem to be no tests that actually compare perimeter protection solutions such as antivirus or Intrusion Prevention Systems (IPS) or both on anything but a set of features (IP-blocking, content filtering features, and so on.) rather than attack samples. Surprisingly, we found no tests that included a comparison of detection rates and levels of proactive exploit blocking. The reason is fairly obvious: proper comparative testing of perimeter solutions is a *very* non-trivial job. First, it is really very different from testing traditional AV solutions because the target objects are not files, but network transmissions. Second, finding false negatives is very tricky, because for gateway solutions proactive protection is very commonplace. (IPS provides proactive protection while AV features both reactive and proactive detection). Finally, finding false positives is very hard.

Perhaps the easiest test to implement is performance measurement. But even that is far from trivial. First, there is the problem of selecting a representative test set. And a network pattern within any real network may vary greatly from the one used in testing. Second, throughput and latency are interrelated and thus difficult to separate in a test.

One big mistake that can be made is simply to use samples from an AV test set to evaluate perimeter products. You might think that if all the HTML samples are collected from all available AV collections, that this would make up a good representative sample set for testing the performance of an HTTP-scanning device. Not at all!

There are several reasons for that. First, many HTML samples containing malicious code were never transmitted via HTTP. For example, W32/Mimail@MM and JS/Feebs@MM arrive in e-mails containing HTML, but this HTML will only be found in SMTP transmissions. There are also scores of other viruses and Trojans that drop HTML files locally. The chances of these HTML files being transmitted via HTTP are very low; it would only occur if the file were dropped into the "Web" directory of an HTTP server. Second, HTML pages that do not contain code (with "<frame src=" or script redirects) are not malicious *per se*. And, third, the lifetime of Web-based attacks is usually measured in hours. Unless this is taken into account, the test sets will contain high volumes of irrelevant HTML page snapshots that will, for instance, point to IPs or domains that have been taken down months or years ago.

To measure the quality of perimeter protection, especially over HTTP, a decent test corpus has to be put together. One approach that we came up with was to trawl "bad" sites. To get a list of such sites we used www.siteadvisor.com data. Let us start with a "bad" site, one that hosts suspicious files or breaks browser security by using some exploit or other. On www.siteadvisor.com we can check what other sites are linked to from the original one. After following these links we can find more suspicious Web sites. Then we simply repeat the process. When we start the process from another site, we may find another cluster and more bad sites.

Frequently, the "bad" sites cluster together. One reason for this is that by using many links between each other, they can affect PR values and boost their "popularity" levels as seen by the Google search engine (as we saw earlier when discussing index hijacking.)

Now that we have a list of suspicious sites, the simplest way forward is to use Wget to capture Web content (perhaps three levels deep, because humans rarely go very deep and visit obscure corners of Web sites). An even more productive approach may be to use a smart crawler that follows IFrames and hidden script redirects when retrieving the contents. In the end we will have our test corpus. The next step is to perform a simulation of human browsing over the page trees that we have collected.

It would be a mistake here simply to scan the collected objects with an AV scanner. First, that would assign equal weight to the front page of each site and all pages at lower levels. That is, of course, not right, because if for instance a perimeter product blocks the starting Web page due to exploit code found on it, the user is protected and will not visit the rest of the pages from this site. Second, it is not right to assume that a desktop scanner offers the same protection as a gateway device. The latter can use special methods to detect Web threats (such as content filtering, IP blacklisting, firewall rules, and so on.)

Is such a test reproducible? Definitely. Is it fair? That is a more complex question, as the variation of many parameters can change the results. For one thing, the selection of the sites for the test set can definitely affect the outcome of the test (but that is, of course, true for any AV test See Muttik I. "Comparing the Comparatives" http://www.mcafee.com/common/media/vil/pdf/imuttik_VB_conf_2001.pdf.)

Tangled Legal Web

The situation with blocking malicious threats from the Internet has an important legal history, due to a certain amount of wrangling within the industry over patents. Two patents in particular come to mind, one of which was called "bane of major players in the anti-virus industry for the last six years" by Virus Bulletin. (http://www.virusbtn.com/news/virus_news/2003/08_19.xml)

A British company called "Hilgraeve" has a US patent 5,319,776 (filed September 29,1992) entitled "In transit detection of computer virus with safeguard." This is a very broad patent that covers scanning of data transmitted over a network. It is a very general idea, but it has been tried in court and should be taken very seriously.

There is also a US patent owned by Trend Micro 5,623,600 (filed September 26,1995) entitled "Virus detection and removal apparatus for computer networks." This patent covers using a proxy for scanning of files transmitted over FTP and SMTP protocols. It has a direct bearing on the protection from Internet threats, and was the basis of litigation against McAfee and Symantec in 1997.

In 1997, IBM and Trend Micro licensed the Hilgraeve patent. McAfee settled with Hilgraeve in 2001 for an undisclosed sum. In the same year, Symantec bought the patent for 62.5 million USD.

In 2003, the Hilgraeve patent was purchased by Clearswift, the current owner.

For a significant part of 2005, certain products from Fortinet were not allowed to be sold in the USA following a court order. In 2006, Fortinet settled a court case with Trend Micro.

NOTE

Patent References
- www.freepatentsonline.com/5623600.html
- www.trendmicro.com/en/about/news/pr/archive/1998/pr012298.htm
- http://www.freepatentsonline.com/5822517.html
- http://www.trendmicro.com/en/about/news/pr/archive/1997/pr051497.htm
- http://patft.uspto.gov/netahtml/PTO/srchnum.html

Summary

We are seeing a significant shift in malware distribution vectors from e-mail (SMTP) to Web (HTTP). This has been accompanied by a range of attacks intended to divert potential victims from legitimate sites to sites hosting malware and other exploits. At the same time, while HTTP retains its position as a major carrier of Internet traffic, existing security solutions providing comprehensive HTTP protection (for instance, full HTML emulation) are in their infancy, and proper independent comparative tests simply do not exist yet. There is more need than ever for multi-layered solutions and a variety of approaches. We predict an increasing use of hardware and convergent technologies to increase the speed, and to reduce latency when scanning HTTP traffic.

Solutions Fast Track

Attacks on the Web

☑ HTTP has grown significantly in recent years as a malware delivery medium, where SMTP-associated attacks such as mass mailers have declined in volume.

☑ It shouldn't be assumed that e-mail has got safer, or that end-users have all learned good e-mail hygiene. Newer non-replicative malware spammed out by e-mail still uses similar social engineering techniques to those used by mass mailers, quite successfully.

☑ Research indicates that advertisements on Web sites are more readily accepted by end users than the same ads received in spam. It's likely that other Web content benefits from a similar "halo effect."

☑ The antivirus community is aware of at least five different kinds of attack over HTTP, including site hacking, manipulation of search engines, DNS poisoning, domain hijacking, and exploiting user errors.

Hacking into Web Sites

☑ Blackhats trick people into visiting a malicious site using a number of approaches. Web defacement involves modifying a popular legitimate site to include malicious links, redirects, or pop-ups pointing to a malicious site.

☑ A similar effect can be achieved by hacking into a Web proxy.

☑ Defacements can, if the modifications are subtle, go unnoticed at the subverted site for some time.

☑ The original CodeRed worm performed a visible defacement of the infected server, but the later W32/CodeRed.c variant was less obvious, since it planted a backdoor.

Index hijacking

☑ Index hijacking is intended to ensure that malicious sites come high up in the list of sites returned by an Internet search engine.

☑ Google uses a technique called PageRank (PR) to determine the quality of a Web page by measuring the number of other pages that link to it. This technique is susceptible to a "Rank Sink" attack.

DNS Poisoning (pharming)

☑ DNS poisoning occurs when either the data on a DNS server is modified illicitly, or data in a temporary DNS cache is subverted.

☑ Weaknesses in BIND have been publicly discussed and exploited for many years. Attacks based on sniffing and spoofing of DNS messages are best addressed by authentication and encryption.

☑ Some gateway products, firewalls, and network and security appliances are also susceptible to DNS attacks.

☑ Many malicious programs modify HOSTS or RHOSTS in order to redirect IP addresses to inappropriate, illegitimate, or spoofing sites, against the user's expectation.

What to Scan?

☑ In order to protect against Web-associated attacks, you need to scan (at a bare minimum) HTTP, SMTP, and FTP.

☑ HTTP's "market share" in total traffic has been eroded by the emergence of other protocols, such as IRC, P2P, and online gaming protocols. In fact, some port 80 traffic includes other non-Web traffic such as VOiP.

☑ There are plenty of solutions for SMTP mail, both at the organization's gateway and at the ISP level. Web-mail or Web-mail-only solutions are less common, reflecting the difficulties in this sector.

Where to Scan?

☑ Multi-layered solutions, where scanning and filtering takes place at the Internet gateway, on the desktop, and sometimes at other places such as LAN servers, are more secure than scanning only at the desktop or only at the gateway. Hooking it into Internet services at the LSP level on the desktop may be more trouble than it's worth.

☑ Perimeter scanning can block known exploits before they reach the target program and system. Ordinary on-access scanners are ineffective against HTML threats, because code is rendered and scripts executed before anything is written to disk. Protection based on browser help-objects can be unreliable.

How to Scan?

☑ Real-time, in-line scanning of Web traffic is resource-intensive, but is necessary where delays due to the scanning process would reduce service levels below acceptable standards (e.g., for DNS lookups).

☑ A proxy scanner is suitable for use where real-time dispatch and receipt is not practical or expected (e.g., SMTP is a "store and forward" technology, not real-time). Security products that cover a number of data transmission protocols may use a combination of proxy and inline scanning, using the most appropriate scanning method for each protocol.

☑ Scanning using a gateway device is complicated by the need to deal with constituent packets and to reassemble the transmission and check its contexts. When a packet stream is inspected serially, the scanner doesn't see the full context.

Parsing and Emulating HTML

☑ HTML has a number of quirks that make it challenging both to parse it and to provide an emulation mechanism for it. While it's easier for a scanner to interpret strings obfuscated by using escaped characters than it is for most humans, there are other complications such as IE's handling of 00 bytes.

☑ Modern Web servers can also send compressed Web pages, if a gateway device doesn't filter out "Accept-encoding" requests.

☑ Without HTML emulation, a Web-facing scanner is reliant on purely reactive identification techniques; it can only identify known threats. Emulation, however, requires the scanner to be able to interpret and run multiple scripting languages correctly, and to de-obfuscate the code.

"JS/Feebs@MM" family

☑ JS/Feebs was the first field mass-mailing virus that was heavily polymorphic.

☑ This was a classic example of a virus family where a new variant was released as soon as antivirus companies caught up and detected the current variant.

☑ It also highlighted the need for antivirus companies to develop more effective script and HTML emulation in order to detect proactively.

Browser vulnerabilities

☑ Browser vulnerabilities may include buffer overflows in the browser itself, or in applications and DLLs.

☑ Problems with the security design can also introduce vulnerabilities such as cross-site scripting issues.

☑ Unsafe or buggy plug-ins can introduce vulnerabilities when an unsafe configuration is considered to be trusted.

☑ Quirks in the user interface are exploited to misrepresent an illicit site as a trusted site.

Testing of HTTP-scanning Solutions

☑ Testing the effectiveness of perimeter-based solutions poses a number of significant problems. In general, detection rates and proactive blocking success are not tested. The problems include the need to test scanning of network transmissions rather than files, and the fact that detection of specific threats may be masked when other generic protection mechanisms pre-empt the detection mechanism.

☑ Compiling a valid test set for HTTP detection testing is not the same as extracting HTML samples from a standard AV test set. For one thing, many HTML threats aren't normally found carried as standard HTTP traffic.

☑ A test set can be compiled by capturing pages from suspicious sites. However, simply scanning the collected objects doesn't constitute a valid test.

Tangled Legal Web

- ☑ Technology for protection against malware has a complex and dispiriting legal history. A number of patents have been taken out that make it difficult for companies using standard technologies and approaches to avoid infringing the holder's rights.
- ☑ The Hilgraeve patent is a very broad patent that protects scanning network traffic from viruses, and is currently held by ClearSwift.
- ☑ Trend Micro has a patent that covers the use of a proxy for detecting and removing viruses from FTP and SMTP traffic.

Frequently Asked Questions

Q: What's the difference between HTTP and HTML?

A: HTML is the most-used markup language for creating Web pages, though the term is also used more generically to include related Standard Generalized Markup Language (SGML) descendants such as Extensible Hypertext Markup Language (XHTML). It formats Web content into a form in which it can be interpreted by a browser. HTTP is the underlying protocol for transferring information on the Web. HTTPS uses the same syntax, but requires the browser to use a Transport Layer Security (TLS)/Secure Sockets Layer (SSL) encryption layer.

Q: Aren't a lot of e-mails HTML?

A: Sure, despite all the efforts of anti-virus and anti-spam gurus, and the ASCII Ribbon Campaign against HTML e-mail (www.asciiribbon.org), for whom the security aspect is only one of the reasons for not sending or accepting HTML mail. The dangers and specific malware threats found in e-mail and in Web browsing are certainly related and sometimes overlap, but by no means identical. This is one of the reasons that compiling a sample set for testing the effectiveness of Web scanners is less than straightforward, as explained in this chapter.

Q: Aren't mass mailers like Mytob, MyDoom and Bagle still having a big impact?

A: Sure. At time of writing they're still making the "top ten" lists of malware reported to vendors. There isn't an exact correlation between numbers reported (detections) and infections. In principle, you could have a comparatively small number of bot-compromised machines flooding the Internet with huge volumes of a given instance of malware, which isn't actually causing new infections, but is being reported widely because of all the protected machines reporting detections. It's actually very difficult to assess the real impact of older malware, especially in light of our lack of information on how many inadequately protected machines are out there.

Q: What kind of user mistakes are exploited in Web attacks?

A: Typosquatting is a common attack (registered variations on the registered names of legitimate business names). This loosely includes common misspellings (singress.com), typing errors such as missing or duplicated letters (synngres.com), similar but misleading names (syngressbooks.com, for instance, instead of syngress.com), or the right prefix with a different top level domain (syngress.ru). Commercial organizations may register many variations on their own name to lessen the risk of their brand being hijacked by phishing sites, malware distribution sites, and even unscrupulous competitors. (As far as we know, these variations on the legitimate syngress.com domain are purely fictitious examples.)

Q: What is a remote root exploit?

A: An exploit that allows the attacker to "root" a remote system: that is to get privileged access that gives them the opportunity to make significant changes such as installing malware ("root" on a UNIX or UNIX-like system is the name commonly given to the all-powerful overall-administrator; hence "root access" and "rootkit.")

Q: So what does pharming have to do with phishing?

A: Not a lot. Both terms derive as much from media fascination with hackerspeak as from any initiative on the part of the security community. Apart from the potential for combination attacks using both DNS poisoning and phishing techniques, the only real resemblance is that both involve some element of spoofing and deception, but then that applies to most forms of attack.

Q: Why is it easier to scan e-mail than Web traffic?

A: Because e-mail is a "store and forward" technology, scanning doesn't have to be real-time; you don't have to scan the message until it's all there on the server. Web traffic, however, does normally have to be real-time, and it's much harder to "look ahead" to get a full picture of the presumed malicious object. It may be possible for malicious code to have been executed before the scanner has determined that it's present. Conventional real-time scanners are of little use in situations where code is executed without ever being written to disk.

Q: Why do so many other protocols "piggyback" port 80?

A: Historically, HTTP was implemented as a means of tying together a number of discrete protocols such as Telnet, gopher and so on (though some of these are only partially supported by modern browsers and operating systems, if at all), and isn't necessarily confined to port 80 (e.g., port 8080 is commonly used for HTTP traffic). However, many applications such as GoToMyPC use Web services to initiate a connection without triggering firewall restrictions, and some other services fall back to port 80 if other ports turn out to be blocked.

Chapter 4

Big Bad Botnets

Solutions in this chapter:

- Bot Taxonomy
- How Botnets Are Used
- Bot Families
- Bot/Botnet Detection and Eradication

☑ Summary

☑ Solutions Fast Track

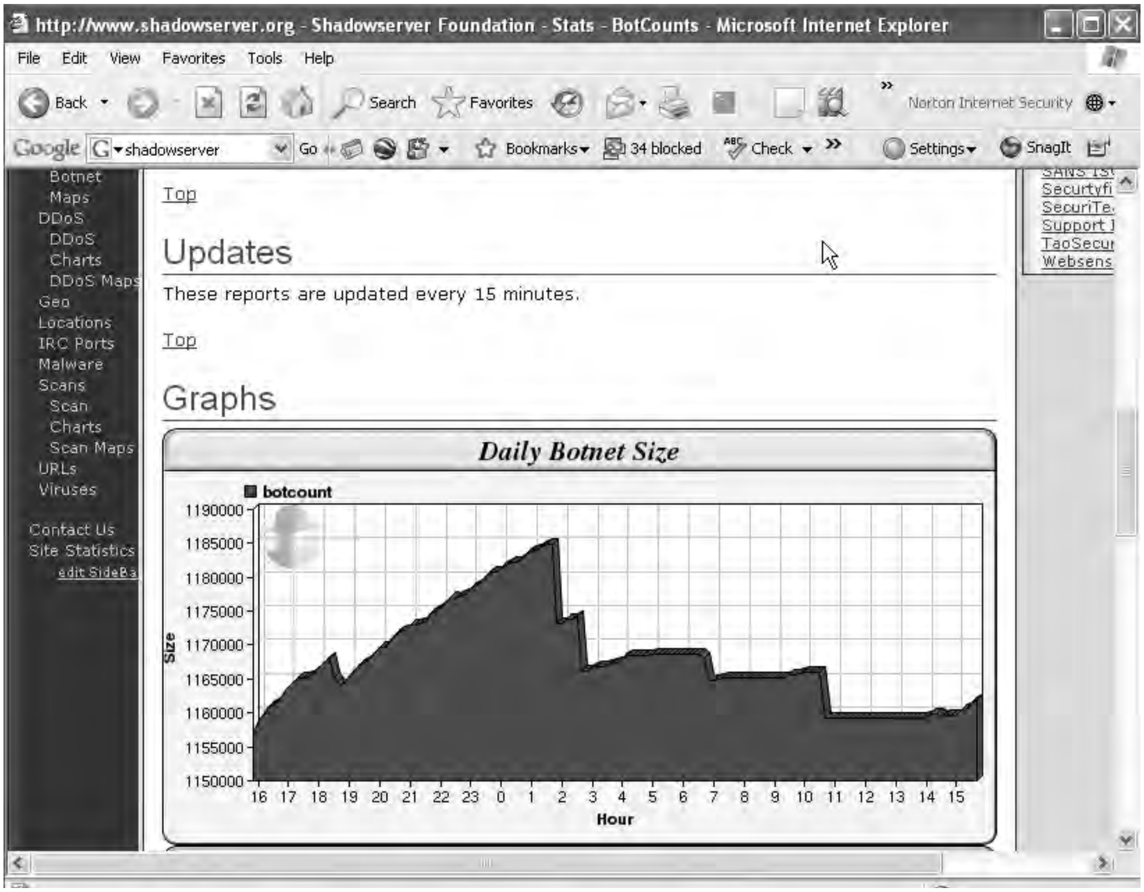
☑ Frequently Asked Questions

Introduction

In "Botnets: the Killer Web App" (Syngress), Craig Schiller and Jim Binkley state that botnets are "arguably the biggest threat that the Web community has faced." Certainly, they are the clearest current illustration of the way in which organized crime has not just discovered the Internet, but discovered the means to exploit it, or at least to exploit huge numbers of the systems connected to the Internet to make equally huge illicit profits. Figure 4.1 gives some idea of the scale of the problem.

In this chapter, Tony Bradley and David Harley revisit the subject and offer a comprehensive overview of the robot (bot) threat and its implications for the enterprise. We can't tell you everything about bots and botnets in a single chapter of this book, so, if you want real detail, you need to check out "Botnets: the Killer Web App," which is currently the best book we know specific to the subject.

Figure 4.1 Shadowserver Statistics Give Some Idea of the Size of the Problem


http://www.shadowserver.org - Shadowserver Foundation - Stats - BotCounts - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Norton Internet Security
Google shadowserver Go Bookmarks 34 blocked Check Settings SnagIt
Botnet Maps DDoS DDoS Charts DDoS Maps Geo Locations IRC Ports Malware Scans Scan Charts Scan Maps URLs Viruses
Contact Us Site Statistics edit SideBa
SANS IS Securtyfi SecuriTe Support TaoSecur Websens
Top
Top
Daily Botnet Size
botcount
Size
1190000 1185000 1180000 1175000 1170000 1165000 1160000 1155000 1150000
16 17 18 19 20 21 22 23 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Hour

Updates

These reports are updated every 15 minutes.

Graphs

Are You Owned?

Assessing the Bot Threat

Bots are widely believed to be the biggest current threat to Internet and Web security. It's not unusual for a bot herder to deploy a "drone army" of 10,000 to 100,000 compromised machines. At a meeting of the World Economic Forum in Davos in January 2007, some alarming claims were made.

Vint Cerf, often described as a "father of the Internet" for his contributions to the development of TCP/IP claimed that there are 600 million systems currently connected to the Internet, and that 100 to 150 million of them were compromised machines controlled by bot herders. John Markoff, a well known writer on technology, claimed that "a single botnet at one point used up about 15 percent of Yahoo's search capacity" by retrieving text for use in evading spam filters (he was probably referring to "hashbusters"), and that around 50 percent of all pirated Windows programs included Trojan functionality. (See http://news.bbc.co.uk/go/pr/fr/-/1/hi/business/6298641.stm.)

Are these figures accurate? Well, despite the celebrity status of the individuals quoted, they aren't people at the sharp edge of malicious software (malware) and botnet management, and it's as well to remember the old saw that "97.354 percent of all security statistics are made up," or at least incorporate much guesswork. A straw poll among some of the volunteers and experts that devote much of their time to bot management suggested that these figures were rather high, but Figure 4.1 and other Shadowserver Institute statistics (Figure 4.2) do indicate big numbers.

It's simply not possible to estimate with real accuracy, though. Technically, the way Internet addressing works (among other factors) simply doesn't allow it. Many of the problems relate to avoiding re-counting the individuals due to recidivism, re-use of Internet Protocol (IP) addresses, and Dynamic Host Configuration Protocol (DHCP) churn, and so on.

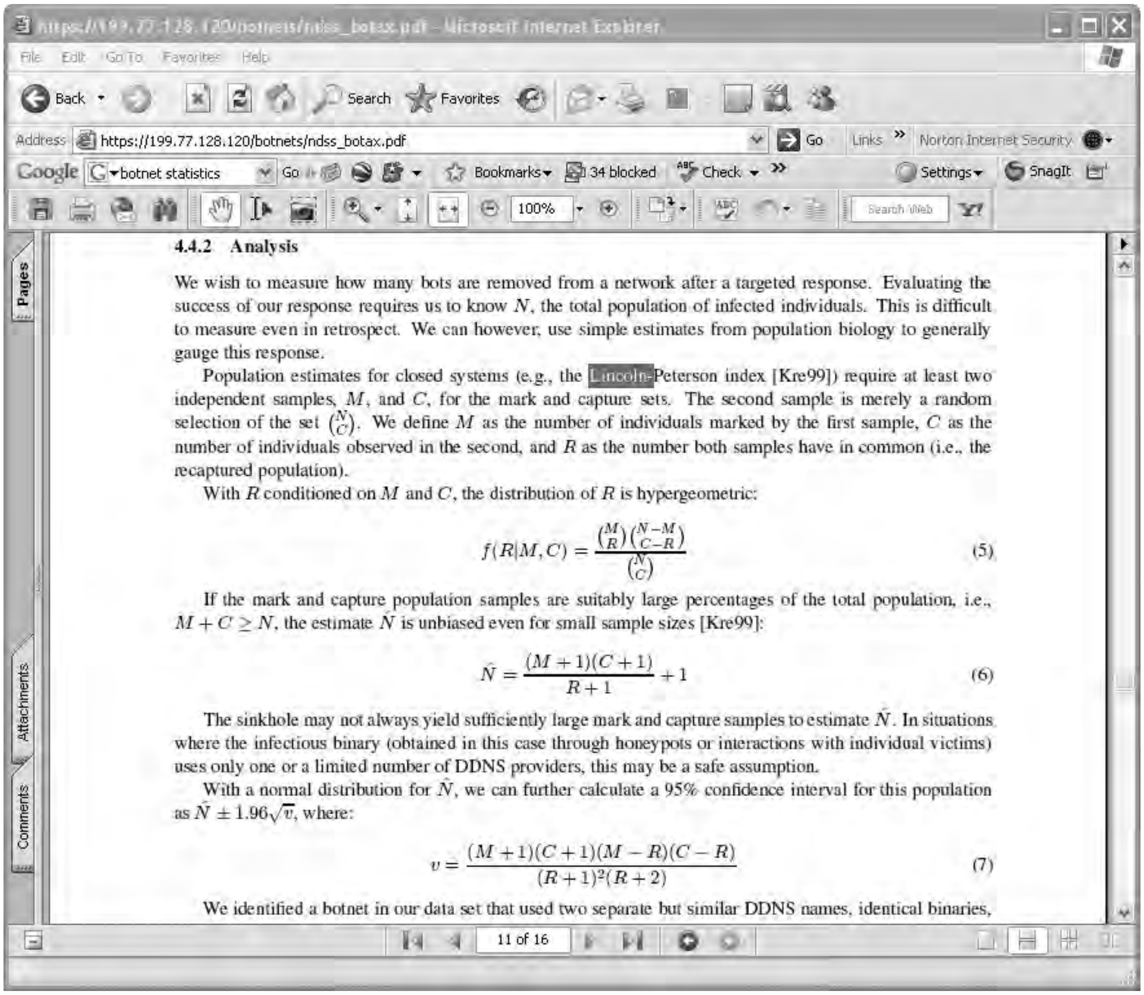
A paper at https://199.77.128.120/botnets/ndss-botax.pdf by David Dagon et al, though largely focused on botnet topology, considers botnet taxonomy (as opposed to bot taxonomy) in some depth. While it doesn't give much insight into the raw overall figures for compromised machines, it does give some sort of feel for the size of individual botnets. The Lincoln-Petersen Index quoted therein gives a methodology for estimating total populations in a closed system.

Some older statistics quoted by Martin Overton in his 2005 Virus Bulletin conference paper, while they seem quite puny in comparison, do at least indicate how both the problem and our perception of it have increased more recently. At the time it was written (around early 2005, using statistics largely from 2004 to 2005), there were considered to be:

- More than one million zombie PCs
- More than 30,000 botnets
- A 93 percent increase in infected PCs due to the take-up of broadband
- Botnets ranging in size from several hundred to more than 50,000, with an average size of 2,000 or more.

More recently, in an article in Virus Bulletin from 2006, Gadi Evron and Dr. Alan Solomon estimate that "there are 3.5 million bots on unique IP addresses used every day for spam purposes alone." (See Figures 4.2 and 4.3.)

Figure 4.2 Population Estimates for Overall Populations from "A Taxonomy of Botnets"



https://199.77.128.120/botnets/ndss_botax.pdf - Microsoft Internet Explorer
File Edit Go To Favorites Help
Back Search Favorites
Address https://199.77.128.120/botnets/ndss_botax.pdf Go Links Norton Internet Security
Google botnet statistics Go Bookmarks 34 blocked Check Settings SnagIt
100% Search Web
Pages Attachments Comments
11 of 16

4.4.2 Analysis

We wish to measure how many bots are removed from a network after a targeted response. Evaluating the success of our response requires us to know N, the total population of infected individuals. This is difficult to measure even in retrospect. We can however, use simple estimates from population biology to generally gauge this response.

Population estimates for closed systems (e.g., the Lincoln-Peterson index [Kre99]) require at least two independent samples, M, and C, for the mark and capture sets. The second sample is merely a random selection of the set $\binom{N}{C}$. We define M as the number of individuals marked by the first sample, C as the number of individuals observed in the second, and R as the number both samples have in common (i.e., the recaptured population).

With R conditioned on M and C, the distribution of R is hypergeometric:

$$f(R|M,C) = \frac{\binom{M}{R}\binom{N-M}{C-R}}{\binom{N}{C}} \qquad (5)$$

If the mark and capture population samples are suitably large percentages of the total population, i.e., $M + C \geq N$, the estimate $\hat{N}$ is unbiased even for small sample sizes [Kre99]:

$$\hat{N} = \frac{(M+1)(C+1)}{R+1} + 1 \qquad (6)$$

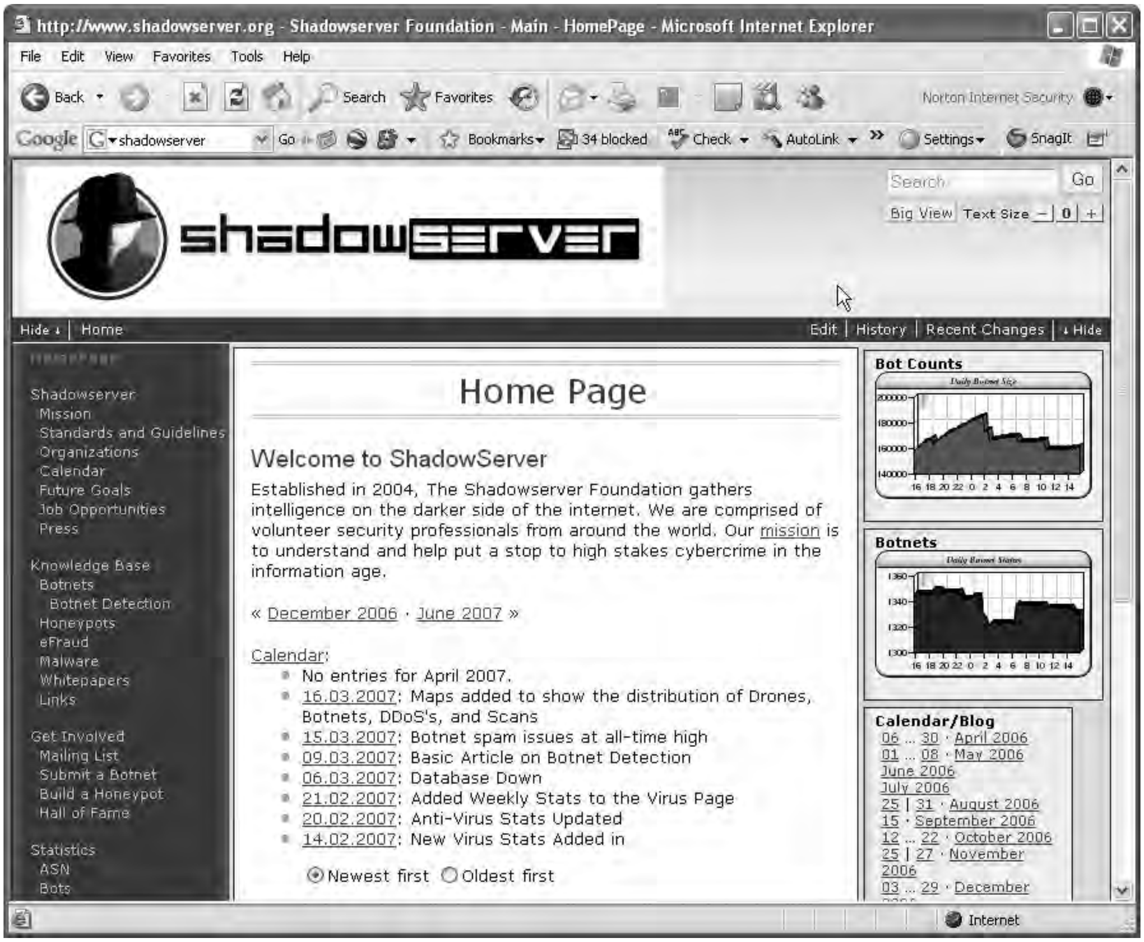
The sinkhole may not always yield sufficiently large mark and capture samples to estimate $\hat{N}$. In situations where the infectious binary (obtained in this case through honeypots or interactions with individual victims) uses only one or a limited number of DDNS providers, this may be a safe assumption.

With a normal distribution for $\hat{N}$, we can further calculate a 95% confidence interval for this population as $\hat{N} \pm 1.96\sqrt{v}$, where:

$$v = \frac{(M+1)(C+1)(M-R)(C-R)}{(R+1)^2(R+2)} \qquad (7)$$

We identified a botnet in our data set that used two separate but similar DDNS names, identical binaries,

Figure 4.3 Shadowserver Specializes in Botnet and Malware Intelligence Gathering



Home Page

Welcome to ShadowServer

Established in 2004, The Shadowserver Foundation gathers intelligence on the darker side of the internet. We are comprised of volunteer security professionals from around the world. Our mission is to understand and help put a stop to high stakes cybercrime in the information age.

« December 2006 · June 2007 »

Calendar:
- No entries for April 2007.
- 16.03.2007: Maps added to show the distribution of Drones, Botnets, DDoS's, and Scans
- 15.03.2007: Botnet spam issues at all-time high
- 09.03.2007: Basic Article on Botnet Detection
- 06.03.2007: Database Down
- 21.02.2007: Added Weekly Stats to the Virus Page
- 20.02.2007: Anti-Virus Stats Updated
- 14.02.2007: New Virus Stats Added in

[✓ Newest first] Oldest first

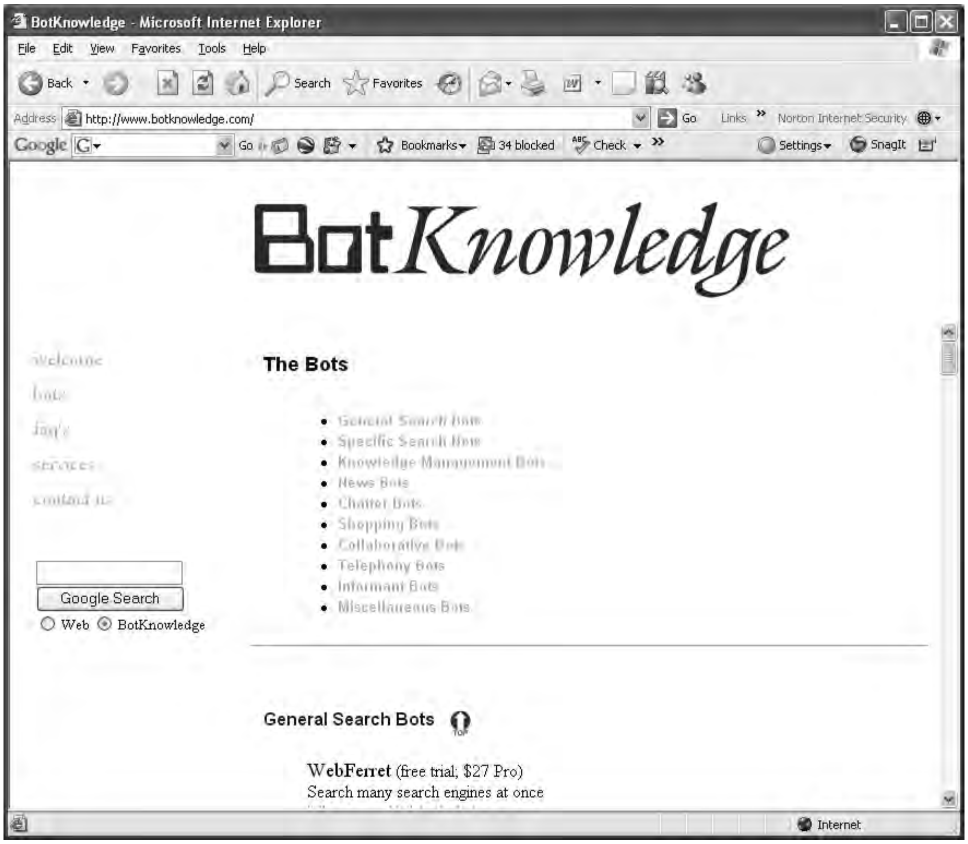
Bot Taxonomy

The term “bot” (from “robot”) is applied to many types of software that execute automated tasks. A very common use of the term refers to Web spiders or Web crawlers, automated scripts that gather and analyze directory and file information from Web servers. Search engines such as Google and Altavista are highly dependent on mechanisms such as these, which accomplish tasks too big and too intensive for human processing. Figure 4.4 lists some ‘benign’ bots. Of course, there are many other legitimate and less legitimate uses for automated programs. These can include, among other things:

- Gaming bots, which impersonate various types of game, play in multiple user games, and often assist beginners to get the hang of the game

- Auction site bots, which track items for sale looking for bargains, or acting to make a winning bid at the last moment.
- Instant Messaging (IM) and Internet Relay Chat (IRC) bots, which may be used to automate a variety of administrative and support tasks.
- Spambots that traverse Web sites, newsgroups, and other systems using screen-scraping Optical Character Recognition (OCR) techniques for circumventing Captcha screens, with the intention of harvesting e-mail addresses to be added to spam-fodder lists. Other spambots post spam links to blogs, Web page guestbooks, Web forums, wikis, and so on, often with the intention of increasing search engine ranking.
- Bots that search Web sites for "scrapeable" content or copyright infringing content, and other surveillance agents.

Figure 4.4 The Bot Knowledge Site Lists Many Benign Bots


BotKnowledge - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.botknowledge.com/ Go Links Norton Internet Security
Google Go Bookmarks 34 blocked Check Settings SnagIt
BotKnowledge
welcome
bots
faq's
services
contact us
Google Search
Web [✓ BotKnowledge]
Internet

The Bots

- General Search Bots
- Specific Search Bots
- Knowledge Management Bots
- News Bots
- Chatter Bots
- Shopping Bots
- Collaborative Bots
- Telephony Bots
- Informant Bots
- Miscellaneous Bots

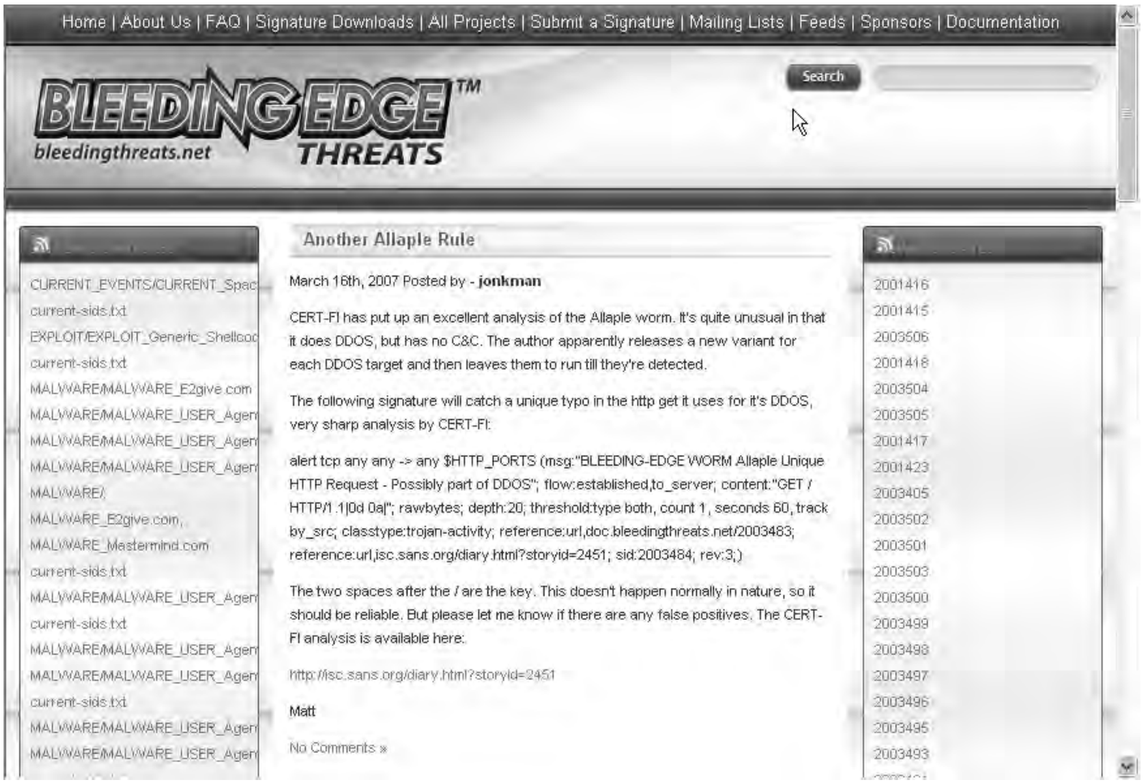
General Search Bots

WebFerret (free trial; $27 Pro)
Search many search engines at once

So, you might think that we could be a little more precise about what a malicious bot actually is. Is it a virus, a worm, a Trojan, or what? The answer is probably "yes..." That is, bot functionality, even in the most malicious type of bot, is not restricted to a single class of malware. A single bot variant may have characteristics of any one (or of more than one) of these threats. We will, therefore, consider the functionality associated with particular bot families in due course, rather than aim for a more specific definition.

A botnet (or bot army, or drone army) is a network of systems connected by the fact that they host an active bot. In principle, the hosted bots can be any number of types of legitimate or illegitimate bots, or a network of distributed computing nodes. As far as we're concerned in this chapter, however, we're talking about a specific and very prevalent type of botnet. That is, a population of "zombie" machines infected (or infested) by a malicious bot (or even more than one) and under the control of a remote "owner." Usually, they use a command and control (C&C) infrastructure, though it's not unknown for a bot to run with no C&C (see Figure 4.5).

Figure 4.5 Matt Jonkman's Rule for Allaple at www.bleedingsnort.com/


Home | About Us | FAQ | Signature Downloads | All Projects | Submit a Signature | Mailing Lists | Feeds | Sponsors | Documentation
Search
BLEEDING EDGE™ THREATS
bleedingthreats.net

CURRENT_EVENTS/CURRENT_Spac
current-sids.txt
EXPLOIT/EXPLOIT_Generic_Shellcod
current-sids.txt
MALWARE/MALWARE_E2give.com
MALWARE/MALWARE_USER_Agen
MALWARE/MALWARE_USER_Agen
MALWARE/MALWARE_USER_Agen
MALWARE/
MALWARE_E2give.com,
MALWARE_Mastermind.com
current-sids.txt
MALWARE/MALWARE_USER_Agen
current-sids.txt
MALWARE/MALWARE_USER_Agen
MALWARE/MALWARE_USER_Agen
current-sids.txt
MALWARE/MALWARE_USER_Agen
MALWARE/MALWARE_USER_Agen

Another Allaple Rule

March 16th, 2007 Posted by - **jonkman**

CERT-FI has put up an excellent analysis of the Allaple worm. It's quite unusual in that it does DDOS, but has no C&C. The author apparently releases a new variant for each DDOS target and then leaves them to run till they're detected.

The following signature will catch a unique typo in the http get it uses for it's DDOS, very sharp analysis by CERT-FI:

```
alert tcp any any -> any $HTTP_PORTS (msg:"BLEEDING-EDGE WORM Allaple Unique
HTTP Request - Possibly part of DDOS"; flow:established,to_server; content:"GET /
HTTP/1.1|0d 0a|"; rawbytes; depth:20; threshold:type both, count 1, seconds 60, track
by_src; classtype:trojan-activity; reference:url,doc.bleedingthreats.net/2003483;
reference:url,isc.sans.org/diary.html?storyid=2451; sid:2003484; rev:3;)
```

The two spaces after the / are the key. This doesn't happen normally in nature, so it should be reliable. But please let me know if there are any false positives. The CERT-FI analysis is available here:

http://isc.sans.org/diary.html?storyid=2451

Matt

No Comments »

2001416
2001415
2003506
2001418
2003504
2003505
2001417
2001423
2003405
2003502
2003501
2003503
2003500
2003499
2003498
2003497
2003496
2003495
2003493

IRC is not the only channel for communication between the bot herder and the compromised machines, but it is convenient for the herder to use an IRC server (or even a specific channel on a public network), not only because the protocol has a history of legitimate and less legitimate bot activity, but because it lends itself to synchronous dialog and data exchange. However, public IRC networks are increasingly aware of the problem and go to some lengths to avoid being incorporated into botnet activities, forcing botmasters to make more use of their own servers.

RFC1459 describes IRC as "...a teleconferencing system, which (through the use of the client-server model) is well-suited to running on many machines in a distributed fashion. A typical setup involves a single process (the server) forming a central point for clients (or other servers) to connect to, performing the required message delivery/multiplexing and other functions."

WARNING

"Here's a thought. Since so many botnets are reliant upon IRC, why don't we just stop IRC running on our networks (or use a safe, controlled IRC system)?"

"Well done, Watson! How do we do that?"

"Well, Holmes, can't we just block port 6667 on our network?"

"Bravo, Watson! But don't IRC servers listen by default to lots of other ports around 6660 to 7000? And don't botnet servers actually avoid port 6667 so that administrators don't pick them up on a routine netstat scan?"

"No problem, Holmes. We just block all those default ports as well."

"Back to the drawing board, Watson. I've just been reading a monograph by the esteemed Martin Overton in the 2005 Virus Bulletin Conference Proceedings, that says that C&C servers run modified IRC servers that can run on pretty much any port."

A zombie (or zombie PC) is a system controlled or controllable by an active bot. The term drone is also sometimes used. To make the distinction clearer, the bot is the "agent" software, and the zombie or drone is the compromised host.

A bot herder (or botherder) is the remote individual or group that "owns" a botnet or bot herd. He or she may also be referred to as a botmaster (bot master), botmeister, or zombie master.

NOTE

For once we use the term "owned" (or "0wned") or even pwned in its hackerspeak sense rather than its more traditional meaning. A system is described as 0wned when compromised by a hacker, rootkit, bot, and so on. The term bot herder is a little more problematical. While the term is a commonly used synonym for botmaster, perhaps it should be restricted to someone who engages in the specific activity of bot herding. This refers to the practice of migrating zombies from one C&C location to another to avoid disruption to the botnet when a C&C box becomes unavailable. This can happen when the box is traced and shut down or cleaned.

Zombie systems are compromised using a variety of techniques and exploits, including buffer and stack overflows and drive-by downloads. We'll consider compromise functionality more closely when we look at specific bot families.

Once a system is compromised, it can be used to scan its environment and propagate further. Indeed, propagation is an important bot function. As David Dagon has memorably said, "The network is the infection." ("The Network is the Infection: Botnet Detection and Response," www.caida.org/workshops/dns-oarc/200507/slides/oarc0507-Dagon.pdf)

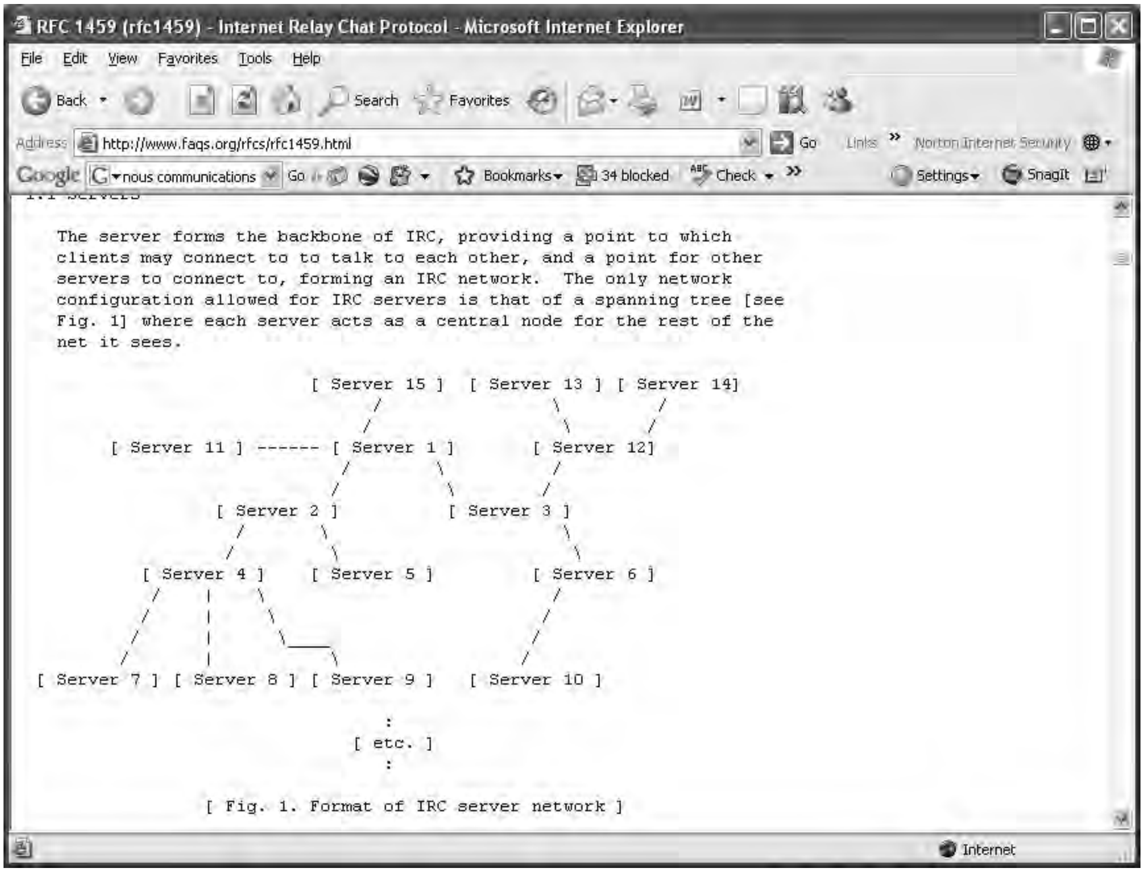
A C&C server is also sometimes referred to as a "rallying box." We speak a botmaster "rallying" victim systems in order to use them in an attack. Over the past few years, taking down the C&C server has been a major objective of those trying to manage the botnet problem, but this approach has become significantly less effective. Like any legitimate enterprise, botnet maintainers are aware of the advantages of diversification and redundancy, and a particular botnet may be served by a distributed "farm" of compromised, high-performance botnet servers, as well as ad hoc services from other zombie machines. A botnet is likely to include a wide range of connections and hardware, from a wide range of sectors. (Clearly, home users are particularly vulnerable, but some other sectors such as education may be difficult to police and therefore to defend.) While IRC seems to remain the "rallying medium" of choice, other tools such as Hypertext Transfer Protocol (HTTP) and Domain Name System (DNS) tunneling continue to gain ground. RFC 1459 is the Baseline Standard for IRC (see Figure 4.6).

In their book, Schiller and Binkley recognize a number of botnet characteristics:

- Modularity
- Adaptation
- The ability to target

Botnets are by definition modular. They consist of a network of compromised machines. However, the compromise (the bot) is in itself also modular. An initial attack may be carried out

Figure 4.6 RFC 1459 is the Baseline Standard for IRC


RFC 1459 (rfc1459) - Internet Relay Chat Protocol - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.faqs.org/rfcs/rfc1459.html Go Links Norton Internet Security
Google nous communications Go Bookmarks 34 blocked Check Settings SnagIt
Internet

```
   The server forms the backbone of IRC, providing a point to which
   clients may connect to to talk to each other, and a point for other
   servers to connect to, forming an IRC network.  The only network
   configuration allowed for IRC servers is that of a spanning tree [see
   Fig. 1] where each server acts as a central node for the rest of the
   net it sees.

                           [ Server 15 ]  [ Server 13 ] [ Server 14]
                                 /                \         /
                                /                  \       /
        [ Server 11 ] ------ [ Server 1 ]       [ Server 12]
                              /        \          /
                             /          \        /
                  [ Server 2 ]          [ Server 3 ]
                    /       \                      \
                   /         \                      \
           [ Server 4 ]    [ Server 5 ]         [ Server 6 ]
            /    |    \                           /
           /     |     \                         /
          /      |      \____                   /
         /       |           \                 /
 [ Server 7 ] [ Server 8 ] [ Server 9 ]   [ Server 10 ]

                                  :
                               [ etc. ]
                                  :

                 [ Fig. 1. Format of IRC server network ]
```

by more than a malicious program. A vulnerability (including human vulnerabilities such as the inability to resist social engineering that makes malware distribution by e-mail so successful) is found and exploited, defensive measures are neutralized, further modules are downloaded, other systems are scanned for vulnerabilities, and so on. Once the zombie system is compromised, the bot functionality is likely to be dispersed between a number of different programs, and components are modified or replaced according to the current function of the botnet, and so as to make detection of all the components of the compromise more difficult. This has a major implication in terms of removal and cleanup. Many organizations consider it necessary to re-image a system rather than risk missing infective components, which may re-zombify it after, cleanup.

The ability to change components and targets makes a botnet adaptive. Ranges of active zombie systems can be switched in and out, the nature of the bot itself can be changed by downloading and upgrading components, and the type of attack and target can be changed according to the whim of the botmaster (or, more realistically these days, according to market forces.)

Notes from the Underground

Designing a Botnet

David Dagon further identifies three botnet design goals:

- Robustness
- Mobility
- Stealth

What are the implications of these goals for C&C techniques?

- A single C&C server ("rallying point") is not robust: take it down, and your botnet is broken. No one likes a single point of failure, unless they're looking for an attack vector.
- A single hard-coded rallying point is neither robust (as before) nor mobile (you can't just move when the current one is taken down.) Hence the popularity of the combination of IRC and dynamic DNS.
- The operation has to be hidden from the "real" owners of the compromised systems, from anti-virus and other security service providers, criminal rivals who may steal your botnet, law enforcement, and the less formal groups who make it their business to watch for and counter bot activity.

Using free dynamic DNS (DDNS) services with a short Time To Live (TTL) or disposable domain names and hosts has several advantages for a botmaster: domain names and DNS records can quickly be discarded and replaced independently, prolonging the life of the botnet. Attack functionality can be distributed among a plethora of IP addresses or hosts, giving the same advantages of redundancy and network resilience that legitimate businesses so often strive for. Both Schiller & Binkley and Gavron & Solomon use the metaphor of the "head of the hydra": as fast as C&C channels and servers are taken down, others kick in. Gavron and Solomon also refer to nested botnet structures somewhat comparable to a terrorist cell network, where a botnet consists of X systems, each controlling Y systems, which in turn may control further systems, and so on. All of which confirms David Dagon's exhortation that "we must track botnets, not just bots." The bad guys have adapted to our increasing success at taking down C&C servers by moving to a less centralized and more diversified approach, reducing their reliance on single C&C servers and using "fast flux" DNS services. Free DNS hosting services may be used to point a subdomain towards an exploited IRC server; however, when the service becomes aware of such misuse, they will nullroute the subdomain by directing it to an inaccessible IP. Where IRC servers are able to mask interconnected servers and bot clients, the botnet is much less likely to be seriously disrupted when a single channel is uncovered.

Notes from the Underground

A Basic Botnet Life Cycle

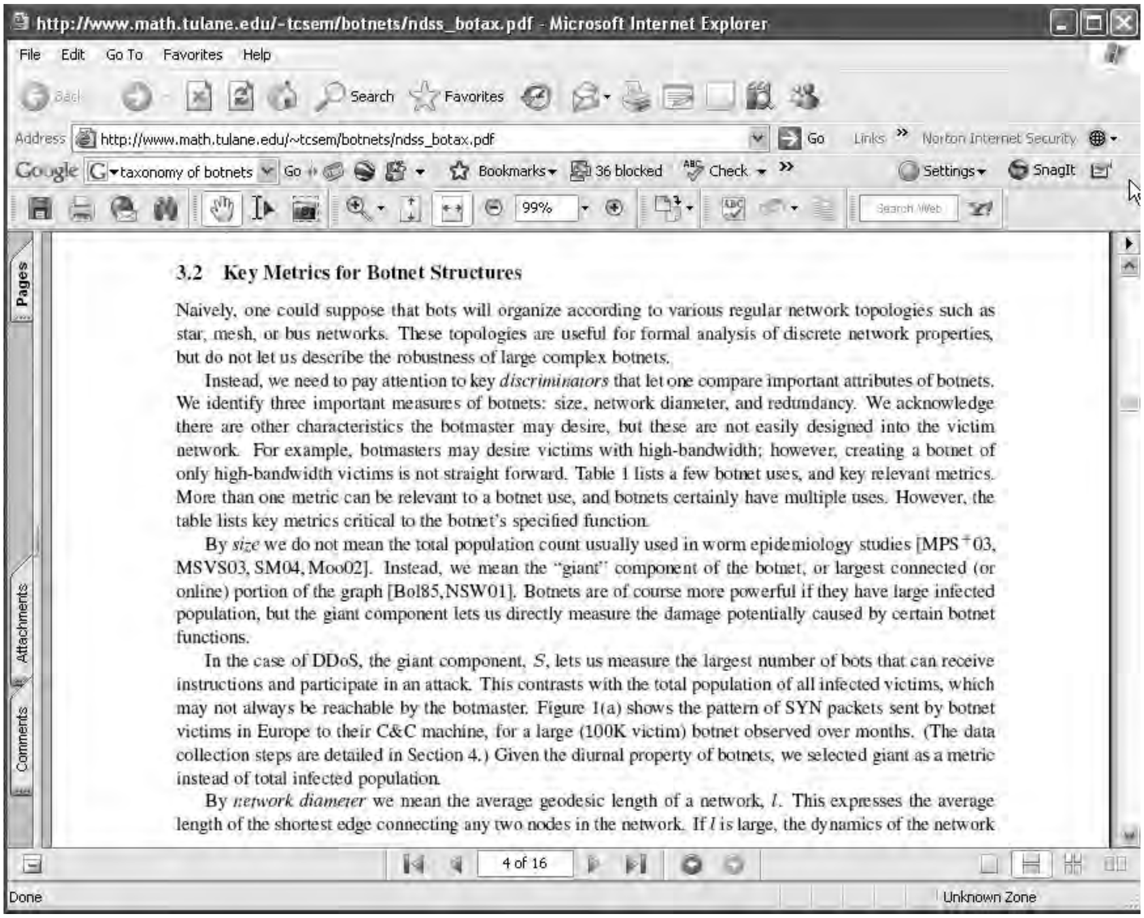
1. The botmaster establishes the components of the attack (acquire domain names, set infection vectors, initial payload, C&C channel, register DDNS).
2. The bot client is compiled and obfuscated, usually by multiple passes through run-time packer, and seeded (disseminated) by virus, worm, or whatever.
3. Once a system has been compromised by installation of the bot, it calls home to the C&C server by joining an IRC channel and waiting for orders.
4. The botmaster instructs the C&C server. Nowadays this is most likely to be in pursuance of a "commission" from a paying customer for a spam run, DDoS attack, click attack, and so on. The instruction specifies the target, time, and type of attack, which set of zombies should participate, and so on, and is relayed to those zombies.
5. The zombies receive and act on the instruction and report back accordingly, waiting for the next set of instructions.

Dagon et al point out that there are a number of alternative types of botnet organization in use, such as:

- **Decentralized Naming Resolution Botnets** Zombies use existing botnets for DNS resolution, rather than centralized DNS resources.
- **Tor Botnets** Botnets use the Tor proxy network to anonymize traffic, increasing the difficulty of detecting and dealing with such botnets.
- **Tunneling Botnets** Increasingly, bots tunnel through other protocols such as Network News Transfer Protocol (NNTP), Web logs, and so on.

The aforementioned paper on "The Taxonomy of Botnets" considers a number of botnet network models. Unfortunately, we don't have room here for a consideration of botnet topology in terms of response, but we recommend the paper to you. (See Figure 4.7.)

Figure 4.7 Dagon et al, Taxonomy Paper: See www.math.tulane.edu/~tcsem/botnets/ndss_botax.pdf


http://www.math.tulane.edu/~tcsem/botnets/ndss_botax.pdf - Microsoft Internet Explorer
File Edit Go To Favorites Help
Back Search Favorites
Address http://www.math.tulane.edu/~tcsem/botnets/ndss_botax.pdf Go Links Norton Internet Security
Google taxonomy of botnets Go Bookmarks 36 blocked Check Settings SnagIt
99%
Search Web
Pages
Attachments
Comments
4 of 16
Done
Unknown Zone

3.2 Key Metrics for Botnet Structures

Naively, one could suppose that bots will organize according to various regular network topologies such as star, mesh, or bus networks. These topologies are useful for formal analysis of discrete network properties, but do not let us describe the robustness of large complex botnets.

Instead, we need to pay attention to key *discriminators* that let one compare important attributes of botnets. We identify three important measures of botnets: size, network diameter, and redundancy. We acknowledge there are other characteristics the botmaster may desire, but these are not easily designed into the victim network. For example, botmasters may desire victims with high-bandwidth; however, creating a botnet of only high-bandwidth victims is not straight forward. Table 1 lists a few botnet uses, and key relevant metrics. More than one metric can be relevant to a botnet use, and botnets certainly have multiple uses. However, the table lists key metrics critical to the botnet's specified function.

By *size* we do not mean the total population count usually used in worm epidemiology studies [MPS⁺03, MSVS03, SM04, Moo02]. Instead, we mean the "giant" component of the botnet, or largest connected (or online) portion of the graph [Bol85,NSW01]. Botnets are of course more powerful if they have large infected population, but the giant component lets us directly measure the damage potentially caused by certain botnet functions.

In the case of DDoS, the giant component, S, lets us measure the largest number of bots that can receive instructions and participate in an attack. This contrasts with the total population of all infected victims, which may not always be reachable by the botmaster. Figure 1(a) shows the pattern of SYN packets sent by botnet victims in Europe to their C&C machine, for a large (100K victim) botnet observed over months. (The data collection steps are detailed in Section 4.) Given the diurnal property of botnets, we selected giant as a metric instead of total infected population.

By *network diameter* we mean the average geodesic length of a network, l. This expresses the average length of the shortest edge connecting any two nodes in the network. If l is large, the dynamics of the network

How Botnets are Used

Botnets are exploited for a wide range of questionable or downright illegal purposes. The following list is not claimed to be all-inclusive. It's entirely possible to use a botnet for pretty much any purpose for which you can use a single PC, while benefiting from the amplified computing power of distributed processing. However, the following types of attack are particularly associated with current botnet activity.

- Establishment and exploitation of Simple Mail Transfer Protocol (SMTP) mail relays and open proxies for spam dissemination. While we won't talk at length about spam in this chapter, spam distributed by this means can include a wide range of spam types, including:

    - "Normal" spam such as that relating to medicines, jewelry and timepieces, mortgage and financial loan offers, educational qualifications, sexual aids, cheap software. Often fraudulent but not necessarily so.
    - Out and out fraud such as the many 419 variations, phishing mails, mule recruitment scams, pump and dump scams, and so on.
    - Distribution of malware, including viruses and e-mail worms, Trojans, spyware, adware, keyloggers, backdoors, and (perhaps primarily) dissemination of bots to other systems and bot updates to systems already compromised.
- Denial of Service (DoS) attacks (usually for purposes of criminal extortion). Also, cloning attacks on IRC networks.
- Click fraud.
- Identity theft, including the theft, storage, and distribution of login IDs and passwords, credit card numbers, and other sensitive data, phishing Web pages, and so on.
- Key harvesting and cracking and other processor-intensive activities using distributed processing across zombie machines.
- Traffic sniffing for clear text information such as usernames and passwords and even data relating to competing botnets ("Know your Enemy: Tracking Botnets," by the Honeynet Project and Research Alliance: see www.honeynet.org/papers/bots/).
- Copyright violation (Dagon et al: "A Taxonomy of Botnets").
- Manipulating online polls and games.
- Adware attacks.

DoS and DDoS ATTACKS

DoS attacks are intended to reduce or destroy the functionality of a service, system. or site, most often by hitting it with so much traffic that it has no resources available to process normal traffic or supply normal services. A DoS attack may also be intended to obstruct communications between the victim site and its users/clients.

A Distributed Denial of Service (DDoS) attack is a DoS attack amplified by being distributed across/launched from a whole raft of machines (e.g., the machines comprising a botnet).

As well as disrupting a network by some form of packet flooding, or disrupting a server by making more service requests than it can handle, a DoS attack might also be aimed at preventing specific individuals or systems from accessing a service, but this is not something we've frequently observed as a botnet function. While we frequently associate DDoS attacks with the disruption of servers (especially Web servers, mail servers, and DNS servers), it's possible for any network device to be exploited as a DDoS vector, including routers. General classes of attack are commonly taken to include:

- Resource starvation attacks, aimed at disrupting services by wasting or reducing resources such as bandwidth (bandwidth saturation attacks), storage capacity, and processor cycles
- Misconfiguration attacks (e.g., attacks that involve disrupting DNS and routing information)

Sometimes, direct or secondary attacks on physical network devices are considered to be a separate category. However, most attacks in this category are covered in the other two categories, and direct physical attacks against hardware are not associated with botnet-originated attacks.

A pulsing zombie is used to execute intermittent DoS attacks. Rather than emitting a continuous stream of attack traffic, a pulsing zombie emits attack traffic in unpredictable bursts, randomized to lessen the risk of detection. The effects of such an attack are, in principle, likely to degrade rather than deny service (hence the term "degradation of service attack" sometimes used). This makes attacking machines harder to trace, and may be used to demonstrate the ability to launch an attack for purposes of extortion.

SYNs and Sensibility

SYN flooding is an attack based on the "three-way handshake" for establishing a TCP connection, which takes the following form. The initiating system sends a Synchronize (SYN) packet to another system, requesting a connection. The receiving system sends a Synchronization Acknowledge (SYN/ACK) packet in response, indicating that it allows connections, and reserves connection space while it waits to receive an Acknowledge (ACK) packet with the connection details from the initiating system. In a SYN flood attack, the victim site is bombarded with forged IP source addresses that don't actually exist, or are not reachable, or that will ignore the SYN/ACK because it's a "wrong number." The gateway/firewall at the victim site sends SYN/ACK packets to these addresses and waits in vain to receive an ACK packet in response. Thus the half-open connection is left hanging until it eventually times out. In the meantime, server resources are depleted or totally consumed, and the number of available connections is reduced or exhausted.

Daniel J. Bernstein defines SYN cookies as the "particular choices of initial TCP sequence numbers by TCP servers." Their use enables a server to avoid dropping connections when the SYN queue fills up. Instead, it sends the SYN/ACK response but discards the SYN queue entry. If it subsequently receives an ACK response, suggesting a legitimate handshake, it can reconstruct the SYN queue entry from the TCP sequence number.

NOTE

A LAND attack also sends a spoofed SYN packet where the same IP address, that of the victim system, is used for both the source and the destination. When this attack is successful, the victim machine may keep replying to itself continuously until it gets a migraine or is taken away by men in white coats.

UDP Flooding

User Datagram Protocol (UDP) flooding attacks are characterized by sending large numbers of UDP packets to random ports on the target system. UDP is a sessionless or connection-less protocol, meaning that it doesn't need to set up a connection before transferring data. As each packet arrives, the system will check for an application listening on that port. When it doesn't find one, it will send an Internet Control Message Protocol (ICMP) "Destination Unreachable" packet to the source address. This source address is often forged, so the attacker's system is not flooded with ICMP packets. This also makes it more difficult to trace the attacker. The idea is to keep the system so busy that it can't process valid connection requests any more.

Notes from the Underground

Smurf's Up

A smurf attack floods the victim system with broadcast ping traffic, which spoofs the source address so that it appears to be that of the target system. On vulnerable IP networks, traffic to those broadcast addresses will get a response from multiple hosts. Fortunately, however, few networks are still configured to be vulnerable to this kind of attack. Instead, routers are configured not to accept directed broadcast packets.

A broadcast address is one where each binary bit is set to one, which means that all hosts within a network are addressed rather than one specific address. It's usually used where the specific address of a single host isn't known, or when it's necessary to contact many hosts on a single network at once. A directed broadcast address routes datagrams in the usual way, but when it arrives at an interface directly connected to the target network, the datagram is forwarded using the data link-layer broadcast address. (See RFC 919.) In response to attacks like this, RFC 2644 recommends disabling directed broadcast forwarding by default.

The Smurf Amplifier Registry (SAR) (www.powertech.no/smurf/) offers a Web-hosted mechanism for probing IP networks for misconfiguration that leaves them vulnerable to a smurf amplification attack. It's also possible to retrieve the full SAR, which can be used as a blocklist see Figure 4.8.

A fraggle attack is a variation on this theme, where UDP echo packets with forged source addresses are sent to IP broadcast addresses.

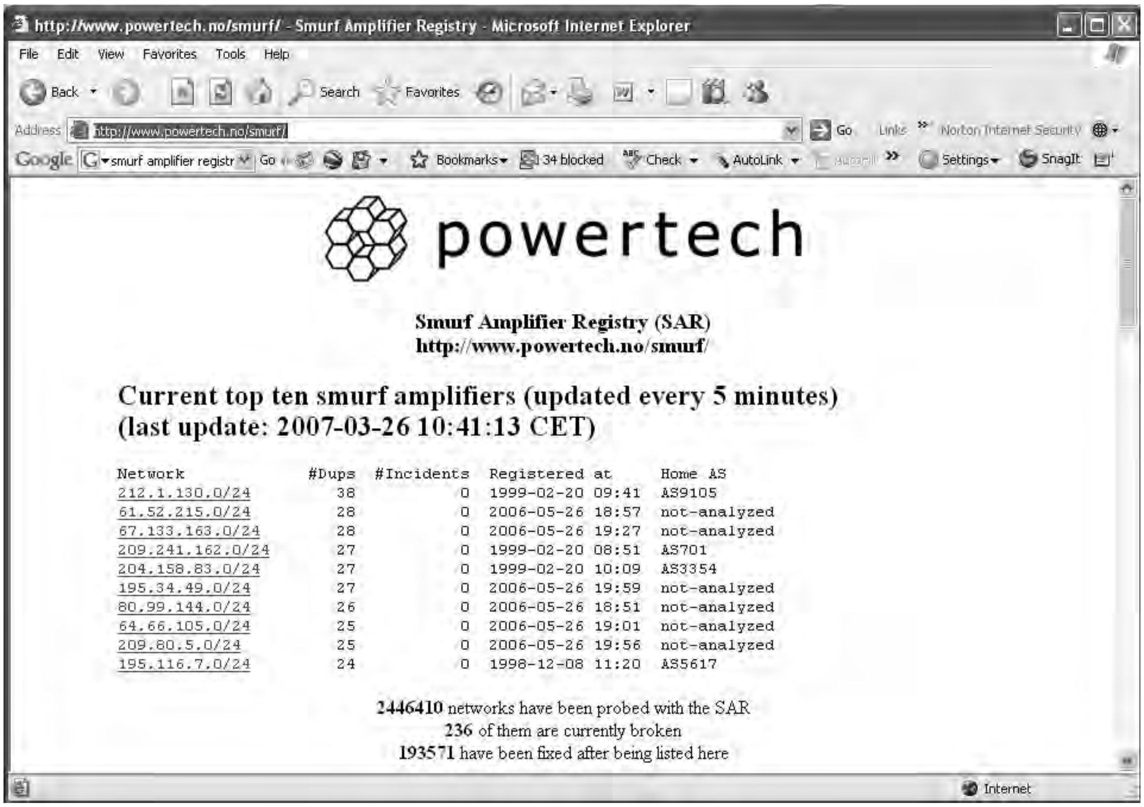
ICMP Attacks

ICMP flooding is an attempt to overwhelm a system with ICMP packets. These are usually either error messages of some sort, or ping Echo Request or Echo Response packets, but in a flood attack they're usually Echo Requests, often sent with "ping –f" so that fragmented packets aren't sent. Again, the intention is to keep the system so busy that it has no more resources to expend on valid traffic.

A Ping Flood is a simple attack, but can still be effective where the victim system has less bandwidth than the attacker. If the victim system is set to respond with Echo Reply packets, the effect of the attack is amplified by the fact that outgoing bandwidth is also depleted.

To mitigate the impact of a Ping Flood, Echo Request packets can be refused all the time, or refused if a volume threshold is exceeded. Either way, bandwidth wastage is reduced and the attacker gets less feedback on the effectiveness of the attack, but legitimate use of ICMP is impacted. Filtering only large Echo Request packets may be a useful compromise measure.

Figure 4.8 The Smurf Amplifier Registry


http://www.powertech.no/smurf/ - Smurf Amplifier Registry - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.powertech.no/smurf/ Go Links Norton Internet Security
Google smurf amplifier registr Go Bookmarks 34 blocked Check AutoLink Settings SnagIt
powertech
Internet

Smurf Amplifier Registry (SAR)
http://www.powertech.no/smurf/

Current top ten smurf amplifiers (updated every 5 minutes)
(last update: 2007-03-26 10:41:13 CET)

```
Network              #Dups  #Incidents  Registered at     Home AS
212.1.130.0/24          38           0  1999-02-20 09:41  AS9105
61.52.215.0/24          28           0  2006-05-26 18:57  not-analyzed
67.133.163.0/24         28           0  2006-05-26 19:27  not-analyzed
209.241.162.0/24        27           0  1999-02-20 08:51  AS701
204.158.83.0/24         27           0  1999-02-20 10:09  AS3354
195.34.49.0/24          27           0  2006-05-26 19:59  not-analyzed
80.99.144.0/24          26           0  2006-05-26 18:51  not-analyzed
64.66.105.0/24          25           0  2006-05-26 19:01  not-analyzed
209.80.5.0/24           25           0  2006-05-26 19:56  not-analyzed
195.116.7.0/24          24           0  1998-12-08 11:20  AS5617
```

2446410 networks have been probed with the SAR
236 of them are currently broken
193571 have been fixed after being listed here

Where the Echo Request includes a forged source address, it also constitutes a Distributed Reflected DoS (DRDoS) attack. When the packet flood hits a network misconfigured to allow exploitation of broadcast addresses, it generates an attack on the system whose address seems to be the source, since the first target will flood that address. Other services that can be exploited by reflector attacks include DNS amplification attacks. A "banana attack" redirects outgoing messages from the client back to the same system, preventing outside access and flooding it with its own sent packets, but this kind of attack is not commonly associated with botnets.

Tools & Traps

Ping Parameters

Ping syntax is platform-specific, but these are some of the parameters for the version supplied with Windows XP (summarized from www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/ping.mspx?mfr=true)

- **-t** Continue sending Echo Request messages to the destination until interrupted.
- **-a** Perform reverse name resolution on the destination IP and display corresponding host name.
- **-n Count** Send [Count] Echo Request messages sent.
- **-l Size** Specify the byte length Data field in Echo Request messages. The default is 32, the maximum is 65,527).
- **-f** Send Echo Request messages with the Don't Fragment flag set. It's intended for troubleshooting Path Maximum Transmission Unit problems, but has obvious implications in terms of bandwidth consumption.
- **-i TTL** Specify Time To Live (TTL) field. The default is host-specific, the maximum is 255.
- **-v TOS** Specify the value of the Type of Service (ToS) field between 0 and 255. The default is zero.
- **-r Count** Use Record Route option for Echo Request message and Echo Reply messages returned. Each entry corresponds to a hop. Count must be between 1 and 9.
- **-s Count** Use Internet Timestamp option to record the time of arrival for each hop. Count must be between 1 and 4.

Continued

- **-j HostList** Use the Loose Source Route option with the intermediate destinations specified in HostList.
- **-k HostList** Use the Strict Source Route option with the intermediate destinations specified in HostList. That is, the next intermediate destination must be a neighbor on a router interface.
- **-w Timeout** Wait [Timeout] milliseconds for an Echo Reply.

DNS Reflector Attacks

The Register (www.register.com) was the victim of a major DNS amplification attack in 2001, using forged requests for the MX records of aol.com which lasted about a week and used a huge list of DNS servers.

There have also been attacks on the DNS root servers, which supply DNS services to all Internet users. The most recent at the time of this writing took place early in 2007, and affected two of the 13 root servers.

Recursive DNS servers are particularly ripe targets for attack, because they process DNS requests for domains on which they are not authoritative querying the root name servers so that the attack affects the root DNS server, is passed on to the applicable top level domain server, and thence to the authoritative server for the target domain, whereas a non-recursive server only provides the information available locally.

A Measurement Factory survey has reported that "There are an estimated 7.5 million external DNS servers on the public Internet. Over 75 percent of domain name servers (of roughly 1.3 million sampled) allow recursive name service to arbitrary queries. This opens a name server to both cache poisoning and attacks. Over 40 percent allow zone transfers from arbitrary queries. This exposes a name server to attacks and gives attackers information about internal networks." (http://dns.measurement-factory.com/surveys/sum1.html)

US-CERT recommends that "Where possible, organizations should secure their DNS servers to ensure that they do not allow recursion or, at a minimum, restrict access to only trusted domains and disable the ability to send additional delegation information."

Vaughn and Gavron state that "Ideally, a recursive name server should only accept queries from a local, or authorized clients. Unfortunately, many recursive name servers accept DNS queries from any source. Furthermore, many DNS implementations enable recursion by default, even when the name server is intended to only serve authoritative data. We say that a name server is an "open resolver" if it provides recursion to non-local users." Their paper on "DNS Amplification Attacks" describes a DDoS attack abusing open recursive DNS name servers, using spoofed UDP packets, and including three detailed case studies with likely botnet involvement. (http://www.isotf.org/news/DNS-Amplification-Attacks.pdf)

There are, of course, attacks against e-mail and other messaging services such as subscription bombing. We don't at present see this sort of targeting in a botnet context, however.

Managing DoS and DDoS Attacks

Apart from paying off the extortionist, what options do you have? (Gavron and Solomon liken this "pay the piper" approach to Danegeld, a payment made by English communities to marauding Vikings in the hope that they would go and pick on someone else. Of course, they always came back for another payment.)

Firewalls can, depending on your firewall/network topology, mitigate some attacks simply by denying the relevant protocol or port. (Filtering on IP addresses may be less successful in a DDoS.) Some attacks are too complex to filter by firewall without losing legitimate traffic: TCP/80-borne attacks, for instance, where the firewall is not able to distinguish good traffic from bad. Depending on your network/firewall topology, your pipe in may be saturated before your firewall ever starts to see the traffic. On the other hand, modern stateful firewalls can differentiate between legitimate and some DoS traffic by confirming that TCP connections are valid before forwarding packets to service networks.

Switches can offer a number of features that can be useful in DoS mitigation, depending on the exact type of attack:

- Access Control Lists (ACLs)
- Bogus IP filtering
- Delayed binding
- Rate limiting
- Traffic shaping

Routers often have rate-limiting capacity nowadays, and routinely use ACLs, and may have other anti-flooding settings.

The Botnet as Spam Tool

An open relay is a mail relay that is configured so that any SMTP server can use it to send mail, without requiring authentication. Because of their widespread abuse by spammers, it's become much less usual for mail servers to be configured in this way (and those that *are* run the risk of a starring role on DNS blacklists). However, it's easy for bot-infected systems to be set up as relays in this fashion.

An open proxy accepts connections from one IP address and then resends the connection back out to another IP address. Such systems have a wide variety of uses to a botmaster, some of them spam-related, and notably for anonymization purposes when using an open relay.

The paper by the Lurhq Intelligence Group at http://www.lurhq.com/proxies.html provides a short history of proxy abuse, from the use of Wingate proxies for Winnuke and ping flood attacks, and illustrates a number of misuses, such as:

- Proxying to IRC or I Seek You (ICQ) to hide a source IP address
- Proxying port 25 to send spam
- Inflating a ranking on a "top 100" site
- Brute force HTTP authentication
- Click fraud (more of that shortly)

WARNING

In the "Education in Education" chapter, David and Judith Harley talk about end users in schools and businesses misusing external proxies so as to avoid a restrictive internal proxy.

Click Fraud

Pay-per-click (PPC) is an advertising technique using sponsored links or ads on Web sites. Performance-based advertising payment schemes have become extraordinarily popular on the Web. If a visitor is interested in the ad, he or she clicks on the ad, and the search network's content partner (primarily the site that displays the ad) is credited accordingly. Click fraud corrupts pay-per-click data by generating illegitimate clicks. It's not only the content partner who gains by this practice because of the commission they claim. It's also competitors in the same marketing space, who may delight in the fact that their rivals are stretching their advertising budgets to cover clicks that offer no sales prospect.

- Product PPCs (or price comparison engines) are fed by an advertiser's product database. When a search for that product, links to advertisers for that product type are shown.
- Service engines are fed by service databases.

Click fraud can take place when a group of real (preferably low-paid) workers are paid to click manually on ads. According to "Wired," the Times of India reported in 2004, the fact that Indian housewives, urban professionals, and college grads are hired to sit around clicking on ads, earning 18 to 25 cents per click and up to $200 a month. ("Click Fraud Threatens Web" by Adam L. Penenberg: see www.wired.com/news/culture/1,65324-0.html). However, in the botnet context, it's very common for the power of distributed processing to be employed to run automated scripts and other programs to generate clicks (and therefore income.) Simulation of human clicks by scripts presents some technical difficulties. For instance, large volumes of clicks from a small group of IP addresses can be flagged and discounted automatically quite easily.

There are, for instance, techniques for converting existing user traffic into clicks. The use of botnets takes advantage of the end users unawareness that their systems are compromised to make use of their activities, using redirects or DNS cache poisoning.

Advertising networks like Google Adwords may benefit in some sense from this type of fraud, but are disadvantaged by the need to pay refunds and even recompense following litigation.

Click Fraud Detection

Proving click fraud is no picnic. If all you have is an IP address (which may "hide" lots of NAT-ed or otherwise concealed addresses), how do you know whether a click is valid? We can't possibly go into detail on this issue here, but recommend that you check out the Tuzhilin report (http://googleblog.blogspot.com/pdf/TuzhilinReport.pdf). This defines two particular problems:

- The difficulty of conceptualizing a universally accepted definition of an "unacceptable click."
- Difficulties with full disclosure of an operational definition, which might tempt more members of the general public to take advantage of the disclosed information to participate in click frauds of their own. If there is no disclosure, however, there is no way for advertisers to verify clicks for which they've been charged. A number of suggestions have been made for detection:
  - Analysis of the advertiser's log files
  - Third-party corroboration using Web-hosted techniques such as single-pixel image placement, scripting, and cookie presentation and verification.

Bot Families

In this section, Tony Bradley and David Harley look at the characteristics of some of the best-known bot families. First of all, Table 4.1 shows some of the most commonly exploited ports and some of the associated bot infection vectors.

Now, let's look at specific families.

Table 4.1 Threats by Port, Identifier, or Bulletin

Port Number	Threat/Vulnerability/Exploit	Relevant Bulletin
	Microsoft Windows Plug-and-Play Buffer Overflow vulnerability	MS05-039
	Microsoft Windows Server Service Remote Buffer Overflow vulnerability	MS056-040
1025	RPC, Windows Messenger	
113	ident	
135 `	DCom, DCom2, lsass.exe, crypt32.dll, IIS Server over SSL MS03-026 (Microsoft Windows DCOM RPC Interface Buffer Overrun),MS04-011 (Microsoft Windows Local Security Authority Service Remote Buffer Overflow)	
139	NetBIOS, lsass.exe, crypt32.dll Secure Sockets Layer [SSL] Library DoS	MS04-011 (Microsoft Windows
143	IMAPD login	
903	NetDevil Trojan	
1433, 1434	MS-SQL Server, MSDE 2000	MS02-056 (Microsoft Structured Query Language [SQL] Server User Authentication Remote Buffer Overflow), MS02-061
17300	Kuang backdoor	
27347	SubSeven backdoor	
2745	Bagle backdoor	
3127	MyDoom backdoor	
3410	OptixPro backdoor	
445	NTPass, lsass.exe, crypt32.dll	MS04-011, MS03-049 (Microsoft Windows Workstation Service Buffer Overrun)
5000	UPNP NOTIFY Buffer Overflow vulnerability	MS01-059

Continued

Table 4.1 Continued

Port Number	Threat/Vulnerability/Exploit	Relevant Bulletin
6129	DameWare	
80	WebDav, CISCO IOS HTTP authorization	MS03-007 (Microsoft Webdav Buffer Overrun)
903	NetDevil backdoor	
UDP/88	Kerberos/ASN.1	
	DameWare Mini Remote Control Server Pre-Authentication Buffer Overflow	CAN-2003-0960
	VERITAS Backup Exec Agent Browser Remote Buffer Overflow	UNIRAS 20041217-00920

The Early Bot Catches the Worm

Malicious bots did not spring fully formed from the ethernet like Venus from the waves. There were a couple of evolutionary steps between early, legitimate IRC bots and SDBot, often considered the first modern malicious bot.

Pretty Park

John Canavan tells us in "The Evolution of Malicious IRC Bots," a white paper for Symantec (www.symantec.com/avcenter/reference/the.evolution.of.malicious.irc.bots.pdf), that the Pretty Park worm emerged in June 1999 with, in a rudimentary form, many of the characteristics of a modern bot.

- It retrieved and made use of system information.
- It retrieved and made use of ICQ login names and e-mail addresses.
- It retrieved dial-up networking settings including usernames and passwords.
- It was able to update itself over IRC.

SubSeven

Schiller & Binkley consider that version 2.1 of the Subseven Trojan, released at around the same time as Pretty Park, was also a significant step in this anti-pilgrimage, in that a bot connected to an IRC server could remotely control a SubSeven server. This set the stage for all malicious botnets to come. SubSeven was a Remote Access Tool (RAT) written, like Pretty Park, in Delphi. (One day we should look at the role played in Borland compilers in the history of malware, on both sides of the good/bad divide.)

Despite passing itself off as a remote administration tool (some RATs in this era were treated with extreme caution by AV companies, faced with the protests of RAT authors about the detection of their "legitimate" tools), SubSeven included such "black" capabilities as password stealing and keylogging. Potentially, SubSeven gave bot operator's full control of systems on which it was installed.

GT Bot

The upgrading in 1999 of the shareware mIRC client had a direct impact on the bot scene. This major enhancement included a resilient scripting language with the ability to respond directly to IRC server events, as well as supporting raw TCP/UDP sockets. These features offered enormous possibilities for new applications, but also for exploitation. Global Threat (GT) bots, based on the mIRC client bolstered by an assortment of malicious scripts, started to become a problem in 2000. Once installed, these would lie concealed on the compromised system and open an IRC channel, waiting for instructions.

Some members of this family attempted to spread from the infected machine across local networks, act as file servers, scan for open ports, and execute flooding attacks.

TFN, Trinoo, and Stacheldraht

Around 1999/2000 was also the time that DDoS attacks went public with a spate of attacks on major Web sites such as Amazon and eBay, which were seriously disrupted for long periods. Tribe Flood Network (TFN) was developed by "Mixter" and tested on compromised UNIX systems. TFN was, like more recent tools, made up of client and daemon tools; however, they were intended specifically for the implementation of DDoS attacks, using ICMP flooding, SYN flooding, UDP flooding, and Smurf attacks. TFN also furnished an "on demand" root shell, bound to a TCP port (http://staff.washington.edu/dittrich/misc/tfn.analysis). (We must acknowledge the debt owed by the security community at the time to the research of Dave Dittrich. Some of the best information available on this group of attacks derives from his analyses.)

TFN included the following command set, as displayed by the program itself if run without parameters:

```
       [tribe flood network] (c) 1999 by Mixter
usage: ./tfn <iplist> <type> [ip] [port]
<iplist>      contains a list of numerical hosts that are ready to flood
<type>        -1 for spoofmask type (specify 0-3), -2 for packet size,
              is 0 for stop/status, 1 for udp, 2 for syn, 3 for icmp,
              4 to bind a rootshell (specify port)
              5 to smurf, first ip is target, further ips are broadcasts
[ip]          target ip[s], separated by @ if more than one
[port]        must be given for a syn flood, 0 = RANDOM
```

Trinoo (or Trin00) attacks are described in some detail at http://www.cert.org/incident_notes/IN-99-04.html and http://staff.washington.edu/dittrich/misc/trinoo.analysis. Stacheldraht combined some of the features of TFN and Trinoo. You might find a consideration of some of the commands supported by Stacheldraht interesting and instructive when compared to the information that follows later regarding more recent C&C malware.

Original Stacheldraht Command Set (http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt; http://xforce.iss.net/alerts/advise48.php)

.distro user server	Install and run a new copy of itself using "rcp" to the account user on the system server.
.help	List supported commands
.killall	Kill all active agents
.madd ip1[:ip2[:ipN]]	Add IP addresses to target list
.mdie	Send "die" request to agents
.mdos	Begin DoS attack
.micmp ip1[:ip2[:ipN]]	Commence ICMP flood attack against target IP(s)
.mlist	List current targets by IP address
.mping	Ping all agents to check whether alive
.msadd	Add a new master server
.msort	Send pings to show counts/percentage of dead/alive agents
.mstop	Stop attacking specific or all IP addresses, according to parameter

.msrem	Remove a master server from the list of available handlers
.msyn ip1[:ip2[:ipN]]	Begin SYN flood attack against specified targets
.mtimer seconds	Set attack duration in seconds
.mudp ip1[:ip2[:ipN]]	Begin UDP flood attack
.setisize	Set size of ICMP packets for flood attack
.setusize	Set size of UDP packets
.showalive	Show all live agents
.showdead	Show all dead agents
.sprange lowport-highport	Set range of ports for SYN flood attack

Stacheldraht Additional Commands (http://www.ciac.org/ciac/bulletins/k-072.shtml; http://xforce.iss.net/alerts/advise48.php)).

.mack	Send TCP ACK flood
.mnul	Send NULL flood
.mstream	Send stream attack flood
.mhavoc	Send a "HAVOC" flood (mixed ICMP, UDP, SYN, TCP random flags, IP headers)
.mrandom	Begin flood attack with random TCP headers
.mip	Send flood of regular IP headers
.mfdns	Set source port for port 53 floods
.forceit	Enforce .mstop to stop all flooding
.left	Display time left before agent stops flooding

Trinoo used UDP for communication between handlers and agents, while the original Tribe Flood Network used ICMP. Stacheldraht used both. While none of these tools used IRC for C&C, Stacheldraht ("Barbed wire") did have commands for attacking IRC.

IRC Flooding Commands

.enter	Enter IRC flooding interface
.part	Part channel
.join	Join channel
.msg	Launch message flood

SDBot

SDBot has been a thorn in the side of the AV community for some years. Its resilience is partially due to "generosity" of the author in releasing the source code (not all open source is a good thing!). Ease of maintenance and modification has made it a popular choice for blackhats.

If an exploit is successful, the worm creates and runs a script which downloads and executes SDBot on the new victim host.

Modern variants use many infection vectors such as spam attacks in Instant Messaging (SPIM), CDs, infected attachments to e-mails, and drive-by downloads on phishing sites. The original motivation seems to have been to launch DoS attacks. In November 2006, Panda labs reported that SDBot.ftp.worm, a component of SDBot, was the most frequently detected virus. This is a testament to the staying power and adaptability of this approach. The June 2006 Microsoft report about the Malicious Software Removal Tool listed the SDBot as having detected 677,619 infected PCs, the second highest total at that time. (Number 1 was Rbot, with 1,914,046.)

Infection and Propagation

SDBot relies on weak security on target systems or the ability to leverage the current user credentials to connect with other network resources. SDbot assumes the same access rights and privileges as the user currently logged into the system.

This bot attempts to connect to and spread via default administrative shares found on a typical Windows system, such as PRINT$, C$, D$, E$, ADMIN$, or IPC$. Some variants come bundled with a listing of common username and password combinations (e.g., weak passwords such as "abc123" or "password," which can sometimes be used to connect with network resources as well.

Some variants scan for Microsoft SQL Server installations with weak administrator passwords or security configurations.

SDBot typically includes some sort of backdoor, which allows an attacker to gain complete access to compromised systems. The remote access Trojan (RAT) component connects to an IRC server and lies silently, waiting for instructions from a botmaster.

Using the RAT, a botmaster can collect information about the compromised system, such as the operating system version, computer name, IP address, or the currently logged in username. A botherder can also run IRC commands, directing the compromised computer to join an IRC channel, download and execute files, or connect to a specific server or Web site to initiate a DDoS attack.

Upon execution, SDbot will place a copy of itself in the System folder. Typically, this folder is *C:\Windows\System32*, but SDbot uses the *%System%* variable to find out where it is and then places a copy of itself in that folder. The file name used may vary.

SDbot also makes modifications to the Windows Registry aimed primarily at making sure that the SDbot software is automatically started each time Windows is booted up.

Typically, one of the registry values displayed in Table 4.2, or something similar, is added to one of the following registry keys:

- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices
- HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

Table 4.2 Registry Values Used by SDbot

“Configuration Loader” = “%System%\iexplore.exe“
“Configuration Loader“ = “MSTasks.exe“
“Configuration Loader“ = “aim95.exe“
“Configuration Loader“ = “cmd32.exe“
“Configuration Loader“= “IEXPL0RE.EXE“
“Configuration Manager“ = “Cnfgldr.exe“
“Fixnice“ = “vcvw.exe“
“Internet Config“ = “svchosts.exe“
“Internet Protocol Configuration Loader“ = “ipcl32.exe
“MSSQL“ = “Mssql.exe“
“MachineTest“ = “CMagesta.exe“
“Microsoft Synchronization Manager“ = “svhost.exe“
“Microsoft Synchronization Manager“ = “winupdate32.exe“
“Microsoft Video Capture Controls“ = “MSsrvs32.exe“
“Quick Time file manager“ = “quicktimeprom.exe“
“Registry Checker“ = “%System%\Regrun.exe“
“Sock32“ = “sock32.exe“
“System Monitor“ = “Sysmon16.exe“
“System33“ = “%System%\FB_PNU.EXE“
“Windows Configuration“ = “spooler.exe“
“Windows Explorer“ = “ Explorer.exe“
“Windows Services“ = “service.exe“
“cthelp“ = “cthelp.exe“
“stratas“ = “xmconfig.exe“
“syswin32“ = “syswin32.exe“

These registry values are used to modify the Windows Registry so that SDbot is started when Windows starts. The exact values vary according to which filenames are used on a particular installation.

NOTE

Some variants of SDbot may also create new files in the *%System%* directory for additional functionality. Two files that have been identified from known SDbot variants are *SVKP.sys* and *msdirectx.sys*.

The *SVKP.sys* file is a component of SVK Protector, a copy protection utility that prevents the software from being reverse-engineered. Some variants use this technique in an attempt to prevent security researchers or antivirus firms from being able to analyze the malware and determine how it works. *Msdirectx.sys* is designed to provide rootkit functionality for the software, and allow an attacker to gain complete access and control of the target system without being detected.

SDBot may attempt to communicate with a variety of IRC channels using its own IRC client software. Some examples of IRC channels used by known SDbot variants are:

- Zxcvbnmas.i989.net
- Bmu.h4x0rs.org
- Bmu.q8hell.org
- Bmu.FL0W1NG.NET

SDBot and other common bots continue to evolve. In fact, rather than new bots being developed from scratch, malicious developers generally modify an existing bot program into a new variant. Some bots, such as SDbot, have hundreds or thousands of variants, and make the standard antivirus naming convention of using the alphabet (variant A, variant B, and so on) cumbersome at best. In fact, the convergence not only of threat types but of bot families (bot authors borrow liberally from each other's code, though this also applies to other types of malware), make it difficult and even counterproductive to classify this group of threats by family.

Rbot

The RBot family is one of the most pervasive and complex bot families known. Since 2003, the core functionality of Rbot has continued to drive the primary functionality of hundreds of variants. By its very nature, however, Rbot morphs and evolves over time. File names and techniques used vary from one variant to the next, and may even be randomized, making accurate identification difficult.

Rbot was the first of bot families to use compression or encryption algorithms. Most Rbot variants rely on one or more run-time executable packing utilities such as Morphine, UPX, ASPack, PESpin, EZIP, PEShield, PECompact, FSG, EXEStealth, PEX, MoleBox or Petite.

Once infected with Rbot, a compromised system can be controlled by a botherder and used for a variety of functions including downloading or executing files from the Internet, retrieving CD keys for some computer games, creating a SOCKS proxy, participating in DDoS attacks, sending e-mail, keystroke logging, or capturing video from a Web cam if the compromised system has one connected.

Infection and Propagation

The Rbot family of worms uses different methods to seek out vulnerable targets and find systems to infect. Like the SDbot family, Rbot attempts to exploit weak passwords and poor security on administrative shares to spread across the network. Systems with simple or blank passwords on network shares are easy prey.

Rbot also exploits a range of known software vulnerabilities in the Windows operating system and common software applications. Some variants are also capable of exploiting backdoors or open ports created by other malware infections.

The primary means of propagation for the Rbot family is through Windows network shares. Rbot scans on ports 139 and 445 looking for open connections. If a target is found, Rbot then attempts to connect to the IPC$ administrative share on that system.

If Rbot succeeds in connecting with the target system, it tries to obtain a list of the usernames on the target machine that it can use to gain access. If Rbot cannot get the list of usernames from the target system, some variants will simply try a default list of usernames (like those listed in Table 4.3), which are preconfigured into the malware.

Table 4.3 Usernames Tried by Rbot to Connect with Network Resources

administrator	student
administrador	teacher
administrateur	wwwadmin
administrat	guest
admins	default
admin	database
staff	dba
root	oracle
computer	db2
owner	

For each username that Rbot finds on the target system and for the usernames it is pre-configured with, Rbot attempts to authenticate using a list of weak passwords commonly used. The list of passwords varies from one version of Rbot to the next, but commonly includes passwords like those found in Table 4.4. While we don't intend to give details like this for every possible bot variant, this table is useful if only as a guide to the sort of weak password that has already worked for the bad guys.

Table 4.4 Weak Passwords Commonly Found in RBot Variants*

7	chris	intranet	pwd
1	cisco	jen	qaz
12	compaq	joe	qwe
123	control	john	qwerty
1234	data	kate	root
12345	database	katie	sa
123456	databasepass	lan	sam
1234567	databasepassword	lee	server
12345678	db1	linux	sex
123456789	db1234	login	siemens
1234567890	db2	loginpass	slut
2000	dbpass	luke	sql
2001	dbpassword	mail	sqlpass
2002	default	main	staff
2003	dell	mary	student
2004	demo	mike	sue
access	domain	neil	susan
accounting	domainpass	nokia	system
accounts	domainpassword	none	teacher
adm	eric	null	technical
admin	exchange	oainstall	test
administrador	fred	oem	unix
administrat	fuck	oeminstall	user
administrateur	george	oemuser	web
administrator	god	office	win2000
admins	guest	oracle	win2k

Table 4.4 Continued

asd	hell	orainstall	win98
backup	hello	outlook	windows
bill	home	pass	winnt
bitch	homeuser	pass1234	winpass
blank	hp	passwd	winxp
bob	ian	password	www
bob	ibm	password1	xp
brian	internet	peter	zxc
changeme	internet	peter	

These passwords are used when trying to authenticate user accounts.

If it authenticates successfully with the target machine, Rbot then attempts to copy itself to the locations listed below and schedules a remote job to execute the Rbot software and infect the target machine.

- *\Admin$\system32*
- *\c$\winnt\system32*
- *\c$\windows\system32*
- *\c*
- *\d*

Known Vulnerability Exploits

Another method used by Rbot to propagate itself, is to use exploits of known vulnerabilities. Rbot variants may attempt to exploit one or more of the vulnerabilities listed in Table 4.5 below. If a vulnerable target is found, Rbot executes a small program instructing the target machine to connect to a remote server and to download the complete Rbot code. The connections back to the Rbot source may use alternate port assignments, but are typically made via HTTP (port 81) or Trivial File Transfer Protocol (TFTP) (port 69).

Table 4.5 Vulnerabilities Commonly Exploited by RBot Variants

Microsoft Windows LSASS buffer overflow vulnerability (TCP port 445)
Microsoft Windows ntdll.dll buffer overflow vulnerability (Webdav vulnerability) (TCP port 80)
Microsoft Windows RPC malformed message buffer overflow vulnerability (TCP ports 135, 445, 1025)
Microsoft Windows RPCSS malformed DCOM message buffer overflow vulnerabilities (TCP port 135)
Exploiting weak passwords on MS SQL servers, including Microsoft SQL Server
Desktop Engine blank `sa' password vulnerability (TCP port 1433)
Microsoft Universal Plug and Play (UPnP) NOTIFY directive buffer overflow and DoS vulnerabilities (TCP port 5000)
DameWare Mini Remote Control buffer overflow (TCP port 6129)
Microsoft Windows Workstation service malformed message buffer overflow vulnerability (TCP port 445)
Microsoft Windows WINS replication packet memory overwrite vulnerability (TCP port 42)
RealSystem Server SETUP buffer overflow vulnerability
Microsoft SQL Server 2000 Resolution service buffer overflow vulnerability
Microsoft Windows Plug and Play service buffer overflow vulnerability

Exploiting Malware Backdoors

Some variants of Rbot take the easy route and let other malware do the hard work. These variants are programmed to seek out the default backdoors opened by other malware such as the Bagle or Mydoom worms. Malware backdoors known to be targeted by some Rbot variants, include Bagle, Mydoom, OptixPro, NetDevil, Kuang, and SubSeven.

On initial execution, Rbot copies itself into the *%System%* directory (typically *C:\Windows\System32*). A common filename used by Rbot is *wuamgrd.exe*; however, different variants may use different file names. Some variants may actually randomize the file name so that it is different for each infected system. The file is copied to the *%System%* directory with the read-only, hidden, and system file attributes set, and the date/time stamp of the file is altered to match the date/time stamp on the *explorer.exe* file. As a result, even if a user stumbles upon the file ,it gives the appearance of being an old file that was installed with the operating system.

Rbot is highly configurable and has evolved significantly over time. It will add entries to the Windows Registry to ensure it runs automatically each time Windows is started.

The Registry value is configurable, so it changes from one variant to the next.. The Registry keys typically modified by Rbot are:

```
HKLM\Software\Microsoft\Windows\CurrentVersion\Run
HKCU\Software\Microsoft\Windows\CurrentVersion\Run
HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices
```

Some variants of Rbot are programmed to check the Registry periodically and reset the Registry values if they have been changed or deleted. Rbot also creates a mutex to make sure that only one copy of Rbot runs on a system at a time. Different variants of Rbot use different names for the mutex, but one example that has been identified is rxlsass01b.

Terminated Processes

Many Rbot variants also attempt to terminate processes associated with various security and competitive malicious programs.

Table 4.6 Some of the Processes Terminated by Rbot Variants

regedit.exe	MSBLAST.exe
msconfig.exe	teekids.exe
netstat.exe	Penis32.exe
msblast.exe	bbeagle.exe
zapro.exe	SysMonXP.exe
navw32.exe	winupd.exe
navapw32.exe	winsys.exe
zonealarm.exe	ssate.exe
wincfg32.exe	rate.exe
taskmon.exe	d3dupdate.exe
PandaAVEngine.exe	irun4.exe
sysinfo.exe	i11r54n4.exe
mscvb32.exe	

Once a system is infected, Rbot will attempt to connect to an IRC. The IRC server, channel, port number, and password differ between variants. TCP port 113 is used by Rbot for ident services required by some IRC servers.

TIP

Rbot (like many of the other bot programs, as well as other malware) often attempts to connect to network shares and other resources, using the credentials and access rights of the currently logged in user. You should use a login with restricted or limited access for day-to-day tasks, and only log in with full administrative privileges when it is necessary. This will limit the ability of malware to exploit the privileges of the logged in user to spread itself.

Agobot (Gaobot) and Phatbot

Agobot uses an innovative modular design and additional functionality; that is, infection is phased over three stages rather than delivered in one go by a single module.

- Stage one is the delivery of the first module, containing the IRC bot client and the RAT.
- Stage two is the shutdown of anti-virus processes.
- Stage three is to block access to a range of Web sites, most of them belonging to security vendors.

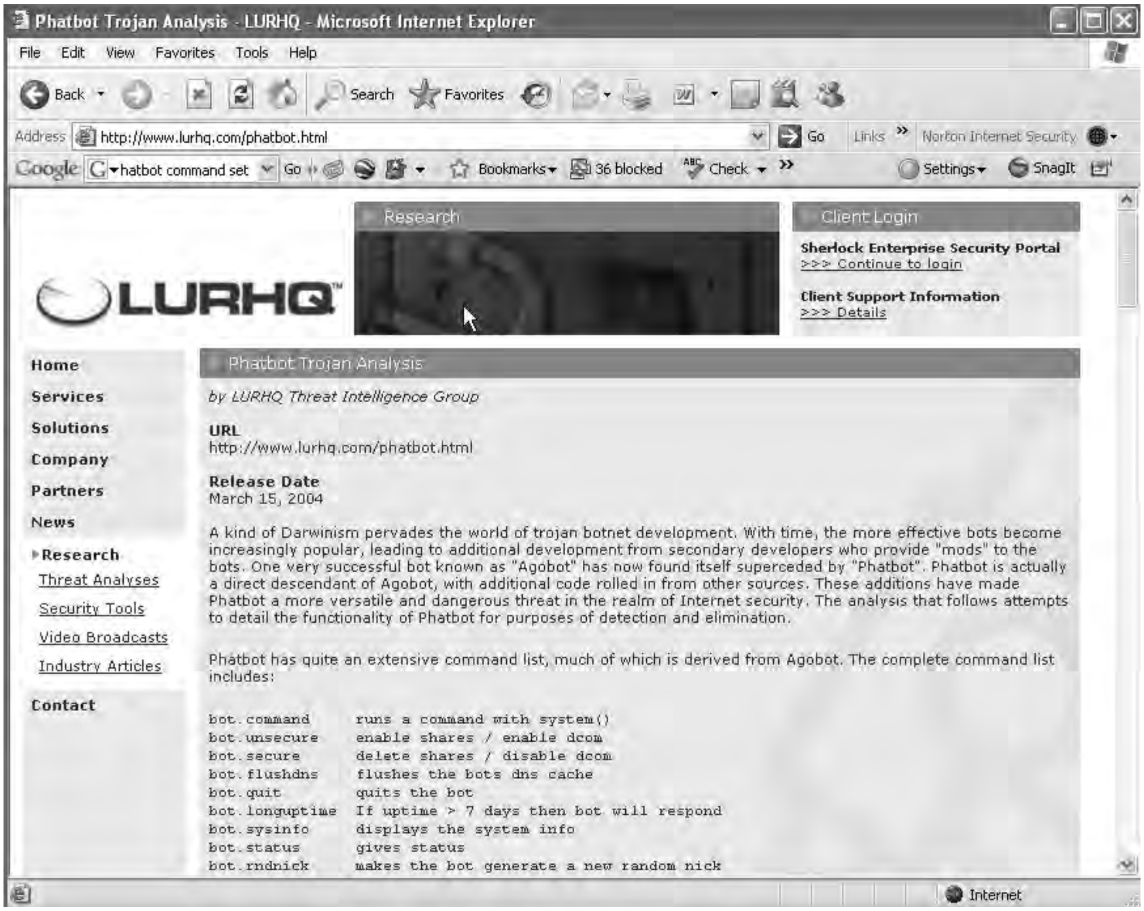
Delivering these modules in phases means that the attacker can modify the attack while in progress by updating the next module as appropriate, without having to rewrite or recompile the whole code.

Infection and Propagation

The Agobot family of malware propagates via network shares, as is common among the major bot families. However, Agobot has the added ability to propagate using peer-to-peer (P2P) networking systems such as Kazaa, Grokster, and BearShare. It makes itself available on the P2P network, using a randomized file name designed to lure unsuspecting users into downloading it and executing it on their computer.

The offshoot variants dubbed Phatbot use WASTE, a P2P protocol designed by AOL. WASTE was designed to use encryption for more secure file transfers via P2P; however, the sharing of public keys was too complicated and AOL eventually scrapped the project. Related bots include Forbot, Polybot, and Xtrembot. (See Figure 4.9.)

Figure 4.9 Lurqh Phatbot Analysis, Including Command Set


Phatbot Trojan Analysis - LURHQ - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.lurhq.com/phatbot.html Go Links Norton Internet Security
Google hatbot command set Go Bookmarks 36 blocked Check Settings SnagIt
Research
Client Login
Sherlock Enterprise Security Portal
>>> Continue to login
Client Support Information
>>> Details
LURHQ
Home
Services
Solutions
Company
Partners
News
Research
Threat Analyses
Security Tools
Video Broadcasts
Industry Articles
Contact
Internet

Phatbot Trojan Analysis

by LURHQ Threat Intelligence Group

URL
http://www.lurhq.com/phatbot.html

Release Date
March 15, 2004

A kind of Darwinism pervades the world of trojan botnet development. With time, the more effective bots become increasingly popular, leading to additional development from secondary developers who provide "mods" to the bots. One very successful bot known as "Agobot" has now found itself superceded by "Phatbot". Phatbot is actually a direct descendant of Agobot, with additional code rolled in from other sources. These additions have made Phatbot a more versatile and dangerous threat in the realm of Internet security. The analysis that follows attempts to detail the functionality of Phatbot for purposes of detection and elimination.

Phatbot has quite an extensive command list, much of which is derived from Agobot. The complete command list includes:

```
bot.command     runs a command with system()
bot.unsecure    enable shares / enable dcom
bot.secure      delete shares / disable dcom
bot.flushdns    flushes the bots dns cache
bot.quit        quits the bot
bot.longuptime  If uptime > 7 days then bot will respond
bot.sysinfo     displays the system info
bot.status      gives status
bot.rndnick     makes the bot generate a new random nick
```

Notes from the Underground…

Polybot Wants a Cracker

Polybot is another major bot family. There is a great deal of confusion when it comes to malware naming, however. One vendor may decide to call a threat one thing and a different vendor may give it a completely different name, despite many attempts over the years to standardize on nomenclature. A complicating issue, when it comes

Continued

to bots, is that many bots are offshoots or evolutions of each other, blurring the lines and sometimes making it difficult to choose whether a new variant is part of the original, or part of the new offshoot strain of malware. This has, to some extent, been a problem with all malware since the early days of virus detection, but has been aggravated in recent years by the proliferation of variants and subvariants and the convergence of code belonging to different malware types and families.

Polybot is an example of such a threat. Polybot is essentially Agobot, but with a technique thrown in that strongly resembles polymorphism as it is encountered in older viral malware. Polybot adds an "envelope" to the Agobot code, which re-encrypts the whole file each time it runs, essentially requiring a new anti-virus or Intrusion Detection System (IDS) signature for each new infection, and thus evading detection by antivirus or IDS products.

Agobot is programmed to terminate a wide variety of antivirus and security programs on infected systems, and also attempts to modify the Hosts file on the infected computer. This is mainly intended to stop the victim system communicating with Web sites associated with antivirus and security applications. Agobot also singles out the Bagle worm, terminating processes associated with that malware if they exist on the infected system. Blocking other, competitive malicious programs is a common feature of modern malware.

Agobot variants attempt to spread via open network shares, as do other bot families. Once a system is infected, Agobot will seek out usernames and passwords on the network using NETBIOS Extended User Interface (NetBEUI). It will then search for open shares such as the default administrative shares (c$, admin$, print$, and so on) and attempt to log in using the usernames and passwords it has found, as well as a pre-configured list of common usernames and passwords.

Agobot also attempts to spread malware via P2P networks, making itself available on those networks using enticing filenames designed to draw attention and increase the odds that the file will be downloaded and executed. It uses a pre-defined list of options to create file names randomly that may be of interest to users. (See Figure 4.7.)

Agobot will drop a copy of itself into the *%System%* folder (typically *C:/Windows/System32*) on the target system. The file name used depends on the variant, but common file names used by Agobot include *syschk.exe*, *svchost.exe*, *sysmgr.exe*, and *sysldr32.exe*.

To ensure that the bot functionality is operational, Agobot creates registry entries to start the bot automatically each time Windows starts. Some variants add a value called "Config Loader" and others add a value called "Svhost Loader" to the *HKEY_Local_Machine/Software/Microsoft/Windows/CurrentVersion/Run* key in the Registry.

Agobot will sometimes add a registry entry aimed at the Windows 95, Windows 98, or Windows ME operating systems. By referencing the dropped malicious file using the key *HKEY_Local_Machine/Software/Microsoft/Windows/CurrentVersion/RunServices* registry key, the bot software will execute, but the service will not be displayed on the Close Program dialog box, making it effectively invisible to the user.

Terminated Processes

Agobot targets a comprehensive list of programs and services for termination. The intention is to seek out processes associated with anti-virus or other security software, as well as processes associated with competing malware, and shut them down. Agobot variants also modify the hosts file of the infected machine to redirect attempts to reach the Web sites of antivirus and security vendors as well.

The hosts file, typically found at *%System%/drivers/etc/hosts*, contains entries for Web sites such as Symantec's LiveUpdate site or McAfee's download site, depending on what security software is loaded on the system. The entries added by Agobot, misdirect any attempts to connect with these sites to the loopback address (127.0.0.1), preventing the connection and blocking the machine from communicating with those sites.

A unique aspect of Agobot is that it will seek out and steal the CD keys for a variety of popular games. Agobot variants also open a backdoor on the infected system, and establish communication with a designated IRC server. This allows a bot master to issue commands to or take control of the compromised system.

The backdoor provides functionality including executing files on the infected machine, downloading additional files from Web or FTP sites, redirecting TCP traffic to the system, using the compromised system as a part of a distribute DDoS attack, and more. Agobot variants can also spread via a variety of exploitable vulnerabilities. Agobot variants target well-known vulnerabilities in CPanel and DameWare, as well as commonly used Windows and SQL Server exploits.

Table 4.7 Components of File Names Used by Agobot to Spread Malware via P2P

SET A	SET B (%s =)
%s -ADSL Playfix	Alessandra Ambrosia
%s -Autotuning (for Newbies)	Amanda Peet
%s -Cable Modem Playfix	Anna Kournikova
%s -CD Key Generator	Ashley Judd
%s -Character Cheat	Belinda Chapple
%s -Crack all versions	Britney Spears
%s -Game Trainer	Cameron Diaz
%s -Idem Duplicator	Carmen Electra
%s -Internet Play Fix	Chandra North
%s -Item Hack	Charlize Theron
%s -Map Hack	Christina Aguilera
%s -Multiplayer Cheat	Donna D'Erico

Continued

Table 4.7 Continued

%s -Newest Patch	Emma Sjoberg
%s -NOCD Patch	Gillian Anderson
%s -Tweaking utility	Halle Berry
%s -Unlimited Healt Trainer	Helena Christensen
%s -Unlock Everything Trainer	Jessica Alba
%s 3D Setup	Jolene Blalock
%s newest version crack	Karina Lombard
	Kate Moss
	Katie Price
	Kelly Hu
	Kirsten Dunst
	Kylie Bax
	Kylie Minogue
	Lexa Doig
	Michelle Behennah
	Pamela Anderson
	Salma Hayek
	Samantha Mumba
	Sandra Bullock
	Shakira
	Stacey Keibler

Spybot

Spybot (sometimes referred to as Milkit) is an evolution of SDBot that emerged in 2003. Like SDBot, the Spybot code is open source and available for the public to modify and contribute to in order to help develop further functionality for the product.

The main difference between the two is that Spybot adds a number of spyware-like capabilities such as keystroke and activity logging, Web form data capture, e-mail address harvesting, Web surfing activities, and more. It spreads via insecure or poorly secured network shares and by exploiting known vulnerabilities common on Microsoft systems.

In fact, Spybot spreads through a variety of methods, including the standard attempt to propagate by finding open network shares with weak or non-existent security. Spybot also spreads via some P2P networks, and seeks out systems compromised by other worms or malware such as SubSeven or Kuang2. In this way, it can piggyback existing backdoors or use open ports to infect systems. It has some similarities/relationships with Rbot, URBot, and URXBot.

Spybot contains the standard bot functionality of providing a backdoor for a botmaster to establish a command and control channel to the infected machine. However, it also adds some less common features such as the ability to broadcast SPIM. It also attempts to modify the registry to prevent various functions like blocking the user from installing Windows XP SP2 or disabling the Windows XP Security Center. Spybot will place a copy of itself in the *%System%* folder (typically *C:\Windows\System32*). Common file names used by Spybot include:

- *Bling.exe*
- *Netwmon.exe*
- *Wuamgrd.exe*

Table 4.8 shows some examples of common Registry modifications found with Spybot variants.

Table 4.8 Spybot Registry Modifications

To create a shared folder on the Kazaa P2P network:
Value: "dir0" = "012345:[CONFIGURABLE PATH]"
Registry Key: HKEY_CURRENT_USER\SOFTWARE\KAZAA\LocalContent
To ensure SpyBot is started automatically when Windows starts:
Value: varies, but will be something like "Microsoft Update" = "wuamgrd.exe"
Registry Keys: Entry made to one or more of the following
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Shell Extensions
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce
HKEY_CURRENT_USER\Software\Microsoft\OLE
To enable or disable DCOM:
Value: "EnableDCOM" = "Y" (or "N")
Registry Key: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\OLE
To restrict network access:
Value: "restrictanonymous" = "1"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa

Continued

Table 4.8 Continued

To disable specific services:

Value: “Start” = “4”

Registry Keys:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\wscsvc

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TlntSvr

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RemoteRegistry

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Messenger

To prevent Windows XP SP2 from being installed:

Value: “DoNotAllowXPSP2” = “1”

Registry Key:
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate

To disable the Microsoft Security Center:

Value:

“UpdatesDisableNotify” = “1”

“AntiVirusDisableNotify” = “1”

“FirewallDisableNotify” = “1”

“AntiVirusOverride” = “1”

“FirewallOverride” = “1”

Registry Key: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Security Center

To disable the Windows Firewall:

Value: “EnableFirewall” = “0”

Registry Key:

HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\WindowsFirewall\DomainProfile

HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\WindowsFirewall\StandardProfile

Spybot will connect to a designated IRC server and join an IRC channel in order to receive commands from a botherder. Some variants will also start a local HTTP, FTP, or TFTP server.

Keystroke Logging and Data Capture

An added feature of Spybot is its ability to capture keystrokes and retrieve personal information, which can be used for further system compromise, or for various forms of identity theft. Variants of Spybot will scan the infected computer for cached passwords, and will log the keystrokes typed on the computer to try and get information such as usernames, passwords, credit card or bank account numbers, and more. The keystroke logging specifically targets windows with titles that include bank, login, e-bay, ebay, or paypal.

Spybot propagates through the same standard means of access as other bot families. Locating open or poorly secured network shares and using them to spread and compromise other systems is the primary method of propagation. Spybot comes pre-configured with a list of commonly used usernames and passwords for general purpose, as well as passwords designated specifically for SQL server account logins.

In addition to network shares, Spybot also seeks out and targets systems that are vulnerable to specific vulnerabilities. Spybot will scan for vulnerabilities on the computers it can communicate with and infect when possible.

Mytob

The Mytob family of worms is an example of convergence between different types of malware. The originators of Mytob took a mass-mailing worm and combined it with bot IRC C&C functionality based on the SDbot and Spybot families. The hybrid combination results in faster propagation and more compromised systems lying dormant, waiting for a botmaster to give them direction.

Mytob arrives on the target system via e-mail with some sort of file attachment. The purpose of the e-mail is to trick (or lure) the user into opening and executing the file attachment, thereby installing the worm on their system and continuing the cycle of infection and propagation. Like other mass mailers, it uses social engineering to trick the recipient into executing the malicious code, and forges the originating e-mail address in order to cover its own tracks and generally muddy the waters.

Mytob is a mass-mailing worm first and foremost. However, an infected system will attempt to connect to *irc.blackcarder.net* and join a specific IRC channel for further instructions. Mytob spreads almost exclusively via e-mail. Once a system is infected, it scans the system for files with file extension like the following from which to harvest e-mail addresses. These are largely (apart from the Windows Address Book) file types used by Web pages and so on.

- *.wab*
- *.adb*
- *.tbb*
- *.dbx*
- *.asp*
- *.php*
- *.sht*
- *.htm*
- *.txt*
- *.pl*

The domains listed in Table 4.9 are eliminated from the harvested e-mail addresses before Mytob starts generating the spam e-mail messages in order to propagate, with the intention of extending its ability to remain undetected.

Table 4.9 Mytob Eliminates Harvested E-mail Addresses from These Domains

.gov	gov.	mydomai
.mil	hotmail	nodomai
abuse	iana	panda
acketst	ibm.com	pgp
arin.	icrosof	rfc-ed
avp	ietf	ripe.
berkeley	inpris	ruslis
borlan	isc.o	secur
bsd	isi.e	sendmail
example	kernel	sopho
fido	linux	syma
foo.	math	tanford.e
fsf.	mit.e	unix
gnu	mozilla	usenet
google	msn.	utgers.ed

NOTE

Mytob sends itself out using its own SMTP engine, but attempts to guess the recipient mail server to make the malware e-mail more convincing. Mytob will try to incorporate any of the following into the target domain name, trying to guess the right mail server: mx, mail, SMTP, mx1, mxs, mail1, relay or ns.

Bot/Botnet Detection and Eradication

What options do we have for bot and botnet detection and removal?

Globally, there are many formal or informal groups watching for bot activity and liaising on takedown of known offenders. We've previously mentioned Shadowserver (www.shadowserver.org), but there are many others working at least partly in this space, including North American Network Operators Group (NANOG) (www.nanog.org), many CERTs and WARPs, and so on. While the Anti-Phishing Working Group (APWG) focuses primarily on one particular dimension of the botnet-driven dark economy, its membership includes law enforcement, financial institutions (that is, primary phishing victims), security vendors, and so on, so that the range of information that can be exchanged between those members can be very wide.

Specific vectors such as e-mail, are closely watched by vendors, system administrators, and so on. ISPs have long been blocking port 25 for systems whose addresses are allocated using DHCP. This can be inconvenient for home-grown/locally implemented and managed mail servers, but reduces the risk of spam outgoing from your network, originating from bots, mass mailers, and other malware. In the enterprise, it's essential to restrict SMTP traffic to authorized servers. Large, heterogeneous and loosely coupled sites using common gates and Network Address Translation (NAT) can be particularly vulnerable to blacklisting, where bot-generated mail (and other) traffic leakage from compromised machines shares address space with properly protected sites and systems.

Anti-virus and other security vendors work their socks off to keep signature-based solutions flowing, as do other groups such as the Snort user community. However, such solutions remain largely reactive. Gavron and Solomon point out that with around 12,000 new bot samples submitted every month, we are long past the point where anti-virus can be regarded as a single point solution. Virus detection is, or should be, an understated way of saying virus management (prevention is too much to hope for). It should sit at all levels of the network from the perimeter to the desktop, and include preventative and recovery controls, not just detection.

Antivirus products are capable of detecting a great deal more than simple viruses, and are not (popular opinion notwithstanding) reliant on simple detection of static strings (signatures). Scanners can detect known malware with a very high degree of accuracy. Even better, they cope with a surprisingly high percentage of unknown malware of all types, using advanced heuristic techniques. Bots offer particular difficulties not only because they employ sophisticated evasion techniques such as multiple packing, but because of the ways in which they are disseminated in short, topographically limited spam runs, over HTTP, over insecure network shares, via unpatched vulnerabilities, and so on. These factors often make them far less susceptible to straightforward technical measures such as code analysis.

Instrusion Detection Systems are vital to the enterprise's anti-bot defenses. A Network IDS (NIDS) should focus on local and outgoing traffic flows as well as incoming Internet traffic, while a Host IDS (HIDS) is likely to pick up symptoms at a local level of bot activity that can't be seen over the network.

At either level, an IDS can focus on either anomaly detection or signature detection, though some are more or less hybrid. Snort is a superb example of a signature-based NIDS with a sophisticated approach to rulesets, in addition to its capabilities as a packet sniffer and logger. You can, of course, write your own Snort signatures, and in Chapter 5 of "Botnets: the Killer Web App" David Harley uses some examples of signatures based on work from Joe Stewart and Martin Overton (as well as elsewhere in this book). The flexibility of the signature facility is illustrated by a number of example signatures, one of which could almost be described as adding a degree of anomaly detection to the ruleset. However, you can also tap into a rich vein of signatures published by a huge community of Snort enthusiasts in the security community. Ourmon, discussed at length by Schiller and Binkley, is an open source tool. Its network monitoring and analysis capabilities are easily adapted to anomaly detection, in terms of bot compromise and botnet-originated attacks.

IDS is best deployed as part of an Internet Prevention System (IPS) strategy, whether it's part of a full-blown commercial system or one element of a multi-layered defense, which may incorporate commercial and/or open source components. Tripwire is an integrity management tool, which can dovetail neatly with an IPS deployment, using a database of file signatures (message digests or checksums, not attack signatures) to detect suspicious changes to files. The meaning open source version of Tripwire doesn't cover the same range of platforms that its commercial big brother does, because it is limited in its reporting facilities, and is closer to the traditional view of change detection in terms of static objects (in particular, files and directories). However, if the devices you want to protect are all Portable Operating System Interface (POSIX)-compliant, it's still worth a little DIY.

Other, more hypothetical techniques discussed by Dagon in various contexts, include DDNS-based detection, DNS monitoring, traceback, and expanded use of threat metrics.

In their botnet book, Schiller and Binkley discuss a number of detection approaches, including network infrastructure tools and techniques, such as sniffing with Wireshark or tcpdump, intrusion detection with Snort, ngrep et al, Layer 2/3 isolation measures, and so on. Sniffing, as Binkley points out, requires considerable filtering to extract signal from noise, but if you have the resources and capacity to make use of it, can tell you a great deal about botnet activity in your own backyard. Finding one bot on a local system and seeing what systems it talks to can help you track other systems tangled in the same botnet.

Cricket (http://cricket.sourceforge.net) is an example of an open source Simple Network Management Protocol (SNMP) which uses RRDTOOL to make graphs of network performance and traffic (http://oss.oetiker.ch/rrdtool/rrdworld/). Cricket is installed on a collection/analysis box and monitors switches and routers with SNMP requests.

Netflow is an industry standard tool for network monitoring, that can handle aggregated statistics for traffic flows. Because these stats are presented as pre-aggregated network traffic data, they are quicker and easier to analyze for specific problems than raw packets, but can also flag when and where to look for detail such as specific IPs among raw logs and packet data. Binkley also recommends the flow-tool package (at www.splintered.net/sw/flow-tools) and Silktools (http://silktoolslsourceforge.net).

There is no single, simple approach to investigating a suspected botnet; however, the chapter in this book on detection and forensics are (unsurprisingly, in the current climate) rather bot-oriented. They should give you a good overview of the tools and techniques available. Make the best of all the resources available to you, from spam and abuse notifications to the logs from your network and system administration tools. Don't overlook the value of automating reports using tools like Swatch. These don't just help you monitor the health of your systems. In the event of a security breach, they give you an immediate start on investigating what's happened and assessing and repairing the damage.

TIP

Information resources for netflow and related tools:

www.cisco.com
http://net.doit.wisc.edu/~plonka/packages.html
www.stats.net.wisc.edu
www.sans.org/reading_room/whitepapers/commerical/778.php
http://www.securityfocus.com/infocus/1796, www.securityfocus.com/infocus/1802

ISP's cooperate on information sharing, preventative and monitoring solutions such as honeynets, and so on.

The term "Darknet" is applied to IP address space, which is routed, but which contains no active hosts, and therefore no legitimate traffic.

You might also hear the terms "network telescope" (http://www.caida.org) or "black hole" (because traffic that finds its way in there doesn't get a response, but simply disappears). Properly analyzed and interpreted, Darknet traffic is a source of valuable data on a variety of attacks (backscatter from spoofed addresses, DoS flooding), and widely used to track botnets and worm activity, since it is assumed that any traffic that finds its way in there, is due to either misconfiguration or malicious intent. Malicious software on the lookout for vulnerable systems can generate a great deal of source material for flow collection, sniffers, and IDSes, without generating the volume of false positives associated with some IDS measures.

The Cymru Darknet project (http://www.cymru.com/Darknet/) defines a Darknet as containing at least one "packet vacuum" server to "hoover up" inbound flows and packets without actively responding and thus revealing its presence. Darknets can be used as a local early warning system, but are primarily encountered as a global resource for sites and groups working against botnets on an Internet-wide basis.

Internet Motion Sensor (IMS) uses a large network of distributed sensors to detect and track a variety of attempted attacks. (http://ims.eecs.umich.edu/)

A darknet resembles a low-interaction honeypot. A honeypot is a decoy system set up to attract attackers, in order to learn more about their methods and capabilities. Or, according to Lance Spitzner, "an information system resource whose value lies in unauthorized or illicit use of that resource" (www.newsforge.com/article.pl?sid=04/09/24/1734245).

A honeynet is comprised of several high-interaction honeypots in a network, offering the attacker real systems, applications, and services to work on, and monitored transparently by a honeywall bridging device.

An excellent resource for honeynet information is the collection of "Know Your Enemy" papers at http://project.honeynet.org/papers/kye.html.

Summary

Bots are a serious threat to Internet and computer network security, unique in their ability to compromise tens or hundreds of thousands of systems, waiting to be used as a drone army for all kinds of malicious activities.

Bot technology is a complex and fast moving area. Bot herders have developed sophisticated mechanisms for staying concealed. A site administrator is likely to have, or at least is considered having, some or all of the tools we've discussed in this chapter, not to mention elsewhere in this book. Is there any site of any significant size nowadays that doesn't have antivirus software or a firewall? The trick, though, is to make the best use of them for proactive and reactive detection.

In this chapter, we looked at some of the major bot families. The bots discussed in this chapter are by no means all of the bot threats available. Malware has shifted from fast-burner viruses and worms intended to spread the fastest and gain infamy and bragging rights, to the blackhat economy of today, where the malware author is part of a gang aiming at financial gain, working on sophisticated financial models.

Monitoring network traffic is not just a matter of ensuring a healthy flow, but an early warning security system, supplementing your firewall and IDS measures. There is no single measure that guarantees detection of bot activity, but good monitoring of multi-layered defenses will contribute immensely to keeping the bots away. However, this isn't a problem that's going to go away just because individual organizations get better at defending their own sites. Botnets parasitize the infrastructure of the Internet itself, and these days, that's pretty close to the framework of society itself. The threat is already barely manageable, and it's only the cooperation and coordinated response of volunteer groups, security and infrastructure vendors, law enforcers and law makers, customers and providers, that stops it from being even worse. As in many areas of security, it's not just a matter of defending your own castle walls, but also of being in touch with the rest of the community under attack.

Solutions Fast Track

Bot Taxonomy

☑ Bots and botnets may well constitute the biggest threat to Internet and Web security at present, as suggested in "Botnets: the Killer Web App" and in pronouncements by such eminent figures as Vint Cerf and John Markoff. However, it's not currently possible to estimate the exact size of the problem, due to factors such as the re-use of IP addresses, recidivist infections, DHCP churn, and so on. Evron and Solomon have estimated that there are 3.5 million bots used on a daily basis for spamming.

- ☑ The term "bot" is applied to agents performing a wide range of legitimate and illegitimate purposes, such as gaming bots, auction bots, administrative IRC bots, spambots, and so on. A bot is not a single type of threat in the same sense that a virus might be considered to be.
- ☑ A botnet (drone army) is a network of systems connected by the fact that they are compromised by the presence of an active bot. Zombie (drone) systems are generally controlled by one or more central C&C systems.
- ☑ IRC is still the most commonly used channel for C&C, but other media such as HTTP and DNS tunneling are gaining ground.
- ☑ Bot herding is, strictly speaking, the practice of migrating zombie systems from one C&C server to another, but the term bot herder is often used nowadays synonymously with botmaster or bot controller.
- ☑ Botnets are modular, not only in the sense that they are comprised of a number of component drone systems, but also in the sense that bots in themselves infect and operate in a modular fashion.
- ☑ Botnets are highly adaptive. The range and structure of the net itself adapts to need as C&C servers are taken down, and the type of attack and target is changed according to the whim of the botmaster and current "commissions."
- ☑ Botnets incorporate three primary design goals: robustness, mobility, and stealth. An example of robustness is the avoidance of reliance on a single C&C server. The combination of IRC and dynamic DNS allows mobility.

How Botnets Are Used

- ☑ A botnet can be used for any number of illicit distributed processes, many of them closely associated with spam and e-mail fraud, including phishing, pump and dump scams, and so on.
- ☑ Other common uses for botnets include the launching of DDoS attacks, click fraud, identity theft, traffic sniffing, and self dissemination by launching and distributing malware.
- ☑ DoS attacks include resource starvation attacks and misconfiguration attacks. DDoS attacks amplify the impact upon victim systems, by channeling the attack through multiple systems. They're usually used in order to extort money from the victim site.
- ☑ A SYN attack exploits the "three-way handshake" for establishing a TCP connection, by leaving connections half-open so that the number of available connections is depleted or exhausted. In turn, UDP floods cause floods of ICMP "Destination Unreachable" packets. Smurf attacks flood victim systems with spoofed ping traffic.

- ☑ A Distributed Reflected DoS attack uses forged source addresses to direct overwhelming traffic to the forged address. DNS amplification attacks are particularly effective against recursive DNS servers.
- ☑ DDoS attacks can often be mitigated by firewall and switch/router configuration.
- ☑ An open relay is a mail relay configured so that any SMTP server can use it to send mail (including malware incorporating its own SMTP server functionality). An open proxy accepts connections from one IP address and resends to another, which has a wide variety of uses to the botmaster.
- ☑ Click fraud corrupts pay-per-click data by generating illegitimate clicks.

Bot Families

- ☑ One of the approaches to diagnosing the presence of a botnet is by observing traffic on particular ports associated with particular programs and vulnerabilities.
- ☑ Interim malicious bot-like malware includes Pretty Park, SubSeven, and GTBot.
- ☑ TFN, Trinoo, and Stacheldraht, though they don't exploit IRC (however, they can attack it) and are Solaris/UNIX-based, are an important evolutionary step. In particular, they brought DDoS attacks to the attention of the world and the Web at large, although the attacks around the turn of the century were not commonly associated with extortion, unlike today's attacks.
- ☑ SDBot owes some of its longevity to the fact that its author released the source code openly. Ease of maintenance and modification makes it a popular basis for evolving malware. Like most bots, it makes extensive use of weak network share protection.
- ☑ The Rbot family continues to thrive. It was one of the first to use compression and encryption algorithms. Runtime packing is used by most Rbot variants. It also uses a wide variety of known Windows vulnerabilities, and includes such wrinkles as a list of default usernames and weak passwords. Some variants use default backdoors opened by other malware, such as worms and Trojan installations.
- ☑ Rbot has significant "retroviral" functionality. It tries to disable a number of processes associated with security programs and "rival" malware.
- ☑ Agobot uses a phased, adaptive approach to infection. It uses other infection vectors such as P2P networking systems like Kazaa. Phatbot offshoots use WASTE, AOL's P2P protocol, with enhanced encryption for more secure file transfer.
- ☑ Polybot is essentially Agobot with enhanced polymorphic functionality.

- ☑ Spybot evolved from SDBot, and its source is freely available. Its main innovation is its use of spyware-like functionality such as keylogging, activity logging, Web form data capture, and so on.
- ☑ Mytob combines code and functionality from standard mass mailers on one hand and bot IRC C&C functionality on the other, so it represents a transition from the mass mailers of the first half of this decade into full-blown bots.

Bot/Botnet Detection and Eradication

- ☑ There are many formal and less formal groups of individuals and organizations monitoring botnet activity, and taking action to hamper their effectiveness. These groups combine the expertise and capabilities of many sectors, including security vendors, law enforcement, ISPs, and so on.
- ☑ Specific vectors such as port 25, should be closely monitored for unauthorized SMTP traffic. Microsoft File Share ports like 135 through 139 and 445 are a major indicator of bot activity.
- ☑ Anti-virus products, especially where aggressive heuristics are deployed, successfully detect a proportion of unknown malware, as well as dealing very capably with known malware and variants. However, there is no longer any such thing as a single point solution in this space, if there ever was.
- ☑ Multilayered anti-virus is far more effective than limited deployment, and more adaptive to changes in the threatscape.
- ☑ Network IDS detects anomalies and attack signatures in network traffic, while Host IDS may pick up anomalies and detect malware manifestations not currently active on the network.
- ☑ IDS is best deployed as part of an IPS strategy, and may fit well with an integrity management solution such as Tripwire.
- ☑ David Dagon suggests a number of alternative approaches to detection, such as DDNS-based detection, DNS monitoring, traceback, and threat metrics.
- ☑ A wide variety of approaches to detection are discussed in Schiller and Binkley's book for Syngress, including a wide variety of network tools.
- ☑ Tools that fit into a detection strategy include sniffers like Wireshark and tcpdump, SNMP-based analysis, aggregated statistical analysis tools like Netflow, and reporting tools such as Swatch.

- ☑ A darknet is IP address space which is routed but contains no active hosts, and therefore no legitimate traffic. The terms "network telescope" and "black hole" may also be used. The assumption is that if you do pick up traffic, it's due to either misconfiguration or malicious activity, and can therefore be used to gather attack information.
- ☑ IMS uses a large network of distributed sensors to detect and track potential attacks, and also uses proprietary transport layer service emulation techniques to attract payload data.
- ☑ A honeypot is a decoy system designed to attract attack traffic. A low-interaction honeypot sits passively and collects information, but is fairly easy to detect and avoid for the attacker. A high-interaction honeypot can supply more information, but may be open to partial or complete compromise.

Frequently Asked Questions

Q: If bots are such a big problem, why isn't there more fuss about them?

A: Because they aren't very well understood. While they've been mentioned in the press, consumer level computer books, and so on for a year or two now, most of the informed discussion has been in specialized (and expensive) security books and periodicals. While there's now some useful info on Wikipedia and elsewhere, it tends to lack detail. So responses to the question "What do you know about bots and botnets?" tend to range from "What's a botnet?" to "Arrrgh! The sky is falling!! We're all going to die!" As so often in security, the issue is both wider spread and less dramatic than many would have you believe.

Q: What's the difference between Instant Messaging and IRC?

A: There's actually something of an overlap. While IRC is seen more as a medium for group discussion (many-to-many or one-to-many), it can be used for one-to-one chat (much as I used to use "talk" back in the Jurassic, when I was a VMS and Unix administrator...) At the same time, IM services owe quite a lot to the older IRC service, without relying on the same underlying protocol. The security implications tend to be protocol- or implementation-specific, though there are obvious generic problems with file-sharing, for example. As far as botnets are concerned, though, IRC has very obvious advantages from a C&C point of view.

Q: Why can't I just kill off IRC on my systems to stop botnets operating on my networks?

A: First of all, botnets don't have to use IRC. Indeed, they don't have to use a C&C structure at all, and we may see more movement away from that model over time. The second issue is that it isn't that easy to kill off IRC. As we've indicated in a sidebar earlier in this chapter, you can't just block the relevant ports, because there are so many possible ports. Even if you can lock down user systems so that they don't install legitimate IRC clients, that doesn't say much about a bot's ability to install IRC client functionality, or even server capabilities, on a compromised system. Having said that, a locked down XP SP2 or Vista system is probably a lot safer than the average home system, not only because of the intrinsic improvements in operating system security, but also because malware finds it more difficult to take advantage of a user with restricted privileges.

Q: What is the advantage to an attacker of passing a new variant through a runtime packer?

A: Doing this, especially when the program is passed through several packers, results in some obfuscation of the code, and makes it harder for a known virus scanner or heuristic scanner to detect a variant, even if it's essentially a lightly modified variant of known malcode. However, if the scanner is able to recognize that a packer has been used, that fact can be used in heuristic mode as an indicator of possible or probable malicious content.

Q: Aren't botnets associated with phishing?

A: Indeed they are. But to explore that relationship in detail would make this chapter extremely long. You can learn more in the Schiller/Binkley book, and Lance James' "Phishing Exposed" (both published by Syngress.) We do address this topic (and the economic aspects of spam in general) a little further in the chapter on the "Blackhat economy."

Q: What is a pulsing zombie?

A: This is a term applied to a zombie that's used in an intermittent DoS (or Degradation of Service) attack.

Q: Are DoS attacks always associated with extortion?

A: By no means. Many DoS and DDoS attacks have been made for other reasons: to "prove" the inferiority of a vendor or service supplier, to try to put a competitor out of business, to make a political point, or out of sheer mischief or malice.

Q: What are some of the common methods used by bots to spread and infect new systems?

A: All of the major bot families target insecure or poorly secured network shares. Typically, the bot contains a list of common usernames and passwords to attempt, as well as some capability to seek out usernames and passwords found on the target system. Since Agobot, P2P networking has been added to the bot disseminator's armory. However, the usual suspects such as spammed malware or URLs to malicious objects, are still offering good service to the bad guys.

Q: How do bots typically ensure that they continue running?

A: Bots, like most modern malware, generally modify the Windows registry. They add values to registry keys to make sure that the bot software is automatically started each time Windows starts up. While viruses have done this for many years, current malware such as bots and spyware, go to extreme lengths, using sophisticated techniques to make automated removal of an infection or infestation as difficult as possible. They also attempt to ensure their own survival by searching out and disabling processes associated with security products such as firewalls and antivirus. There are many examples of malware that modify the hosts file on the compromised system, by diverting requests for anti-virus/security sites to the loopback address of 127.0.0.1, so that security software cannot reach those sites to retrieve updates and patches. The Spybot family adds Registry entries to block the installation of Windows XP SP2, as well as registry entries to disable the Windows Firewall and the Windows Security Center.

Q: What ports do I need to watch?

A: Bots and other malware can often use any port (which is why you can't just stop IRC bots by blocking IRC ports), but they are often characterized by the use of a specific port. You may find the table of threats by port and vulnerability at the beginning of the section above on bot families useful. There are a number of Web resources that list specific threats by port, but you shouldn't rely on their being 100 percent accurate, comprehensive, and up-to-date. Try googling on "bot ports" or "Trojan ports." The threat analysis reports from Joe Stewart on www.LURHQ.com, now merged with SecureWorks, are a great source of information on ports and bot behavior generally.

Q: Which is the best anti-virus program?

A: There isn't a single best-of-breed solution. You have to understand the technology well enough to understand what your needs are, and then compare solutions. Look for solutions that combine a number of approaches and are flexible enough to accommodate changes in the threatscape. Slowly but inevitably, security is moving away from pure reactive detection and towards generic preventative measures, but don't waste too much time on anyone who says "this is the only solution you'll ever need" or "...and it never needs to be updated."

Q: If I'm not sure my AV is detecting a specific threat, how can I check?

A: There are sites that offer multi-vendor virus scanning, such as http://scanner.virus.org/, www.virustotal.com and http://virusscan.jotti.org/. Using more than one is useful in that they may use different products and configurations, which can increase the likelihood of detecting something new. There are caveats, though. The fact that none of these sites detects a given object as malicious isn't absolute proof that it's not. And such sites are easily misused for purposes they were never intended for (e.g., as a shortcut to comparative product testing, or as a means of armoring a new bot variant by patching it until it isn't detected by any product).

Q: Do I need on-demand virus scanning if I have on-access scanning on all the time?

A: On-access or real-time scanning gives you ongoing protection. Every time you access a file, it's checked for infection. On-demand scanning is usually a scheduled scan of a whole system. That's worth considering *if* you can set it up for a deep scan using aggressive heuristics *and* you can do that without making the system unusable by slowing it to a crawl. At any rate, most antivirus vendors strongly recommend a regular scheduled scan, and usually set one up at least once a week, by default. Also, if you have systems that can't conveniently be scanned on-access (e.g., some order Windows versions are not supported by most antivirus vendors). Some other operating systems aren't supported by commercial AV at all. This may not be an issue in terms of native

threats, however, if the system can be used as a stepping stone by malware for other platforms (this is sometimes referred to
as heterogeneous virus transmission), you still need to be able to scan it. If the system is unsupported by a native scanner, you may be able to scan it remotely from a server on another platform. The other time you need on-demand scanning is if you're running a forensic examination or simply cleaning up after a known infection or infestation. Again, you'll need the most paranoid settings.

Q: Do I need anti-virus on my Macs and Linux machines when there are so few threats on those platforms?

A: Malware for OS X is still fairly rare, but it happens. There is more malware for older Mac OS versions, though it's seldom seen now. Linux has been around a lot longer than OS X, and has attracted a lot more malware (but very few real viruses). You might want to bear in mind that many of the first DoS/DDoS attacks to catch the public's attention

Chapter 5

Crème de la Cybercrime

Solutions in this chapter:

Introduction

To quote Sarah Gordon, "Who writes this stuff?" (Unfortunately, we can't actually find an online source for that paper, which is a shame, as it's still worth reading.)

Actually, Gordon's name crops up a lot in the first section of this chapter, when David Harley discusses "Old School Virus Writing." Her work in this area comes close to defining what we know about the virus writers of yesteryear, from her conversations with "Dark Avenger" (not referenced here, but worth checking out) to her contributions to forums such as alt.comp.virus (which is where David first made her acquaintance, virtually speaking), to her many articles and papers for conferences, vendors, and so on. In fact, those of us who have worked seriously on psychosocial issues in malware management will usually admit to owing Ms Gordon a huge debt for her groundbreaking work in this area, and the bulk of this section summarizes some of that work.

In the second section, Enrique González talks (with a little interference from David here and there) about the "Blackhat Economy" where the botmasters and phishers, descendants of the old school virus writers with a purely criminal raison d'être, pursue their sinister agenda.

In the third section, Ken Dunham (Director of the Rapid Response Team at iDefense) and Jim Melnick (Director of Threat Intelligence, also at iDefense) present a case study on hacker-for-hire "Wicked Rose" and his cronies in the NCPH Hacking Group.

Finally, in "Lurkers in Your Crystal Ball," Enrique takes a shot at predicting some future trouble spots.

Old School Virus Writing

There are many people who are convinced that it's the anti-virus (AV) industry that does all or most of the virus writing. We discussed that in Chapter 1, though, and don't intend to go back to that debate here.

It's interesting, nevertheless, that Sarah Gordon's 1994 EICAR conference paper "Why viruses are not – and never were – a problem" already cited a number of similar allegations from contemporary sources about groups (academic virus researchers, AV software developers) as well as individuals within the research community. However, she also quoted a description of "young, disaffected rebels, vicious and without remorse..." This is much closer to stereotypes that have been current among many groups, including that same research community. In fact, Gordon's "Who writes this stuff?" paper includes the comment that "Back in the early 90s, we were certain that they were depraved young men with chips the size of Manhattan on their shoulders. They wrote viruses to destroy the world, make societal protests, and take the place of the girlfriends they could never have, all while listening to heavy metal in their darkened rooms."

We can now see several shifts in perception over the past ten or fifteen years, far away from the picture of the young sociopath with technical skills far exceeding his moral and ethical awareness. Gordon's early research, based on Lawrence Kohlberg's developmental scale, suggested that the young people writing viruses at that time were actually "on target" in terms of ethical development, and could generally be expected to "age out," especially when influenced by ex-writers of malware.

Generic Virus Writers

Gordon's influential paper "The Generic Virus Writer" presents four case studies, dealing with individuals representing one of each of four categories of virus writer, but selected from a larger group of survey respondents:

- The adolescent virus writer
- The college student virus writer
- The adult and professionally employed virus writer
- The mature, reformed ex-writer

Pivotal to the paper is the contention that not all virus writers are bad people intent on doing damage. In fact, some are within normal ethical developmental model ranges. While her sample as a whole comes from a very mixed group, Gordon's case studies indicated that the first two were within the normal range of ethical development for their age group. However, no adult virus-writing respondent out of the whole group answered in such a way as to indicate that they regularly functioned at level three according to Kohlberg's model (see below). This was in contrast to a control group of virus non-writers who consistently behaved in accordance with level three, stage five. (The full paper is available at http://researchweb.watson.ibm.com/antivirus/SciPapers/Gordon/GenericVirusWriter.html.)

Notes from the Underground

Kohlberg's Model

The model used in Gordon's paper assumes three levels of moral development, (with each level characterized by two stages):

- **Level One, "Pre-Conventional Morality"** Goes through punishment orientation, where morality is defined in terms of punishment avoidance, to instrumental orientation, where good behavior is seen as being positively rewarded.
- **Level Two, "Conventional Morality"** Actions are judged on merit of intent, and the subject learns to accept authority out of a sense of duty.
- **Level Three, "Post-Conventional Morality"** Moral principles are internalized: the individual moves to considering the welfare of the many above that of the individual, and even towards normative ethics based on self-chosen principles.

Continued

To make sense of the results above, we need to understand the distinction between stages 5 and 6, into which Level 3 is divided. In stage five, the "social-contract orientation" phase, the needs of society supersede self-interest. In stage six, "Normative ethics, based on self-chosen principles are applied in all situations...`Right' is an obligation to the universal principles of equality, justice and respect for persons." However, people do not generally operate consistently at level six, which is often considered hypothetical, and there's no indication in the study that the control group were any different. In other words, the demarcation between the adult virus writers who responded and the adult members of the control group is sited between stages five ("accepting authority out of a sense of duty") and six (prioritizing the welfare of the majority.)

In a 1996 follow-up paper, Gordon went on to consider motivation, which at that time was widely assumed to include boredom, seeking notoriety, and bragging rights, curiosity and exploration, and peer pressure. And, of course, sheer malice. She also touched on the teaching and condoning of malware writing skills in educational establishments, an issue that continues to make waves. For example, in 2003, many people signed a public letter protesting the use of virus creation in an academic course at the University of Calgary (www.avien.org/publicletter.html), and a subsequent course at the same institution on spyware and spam created similar controversy.

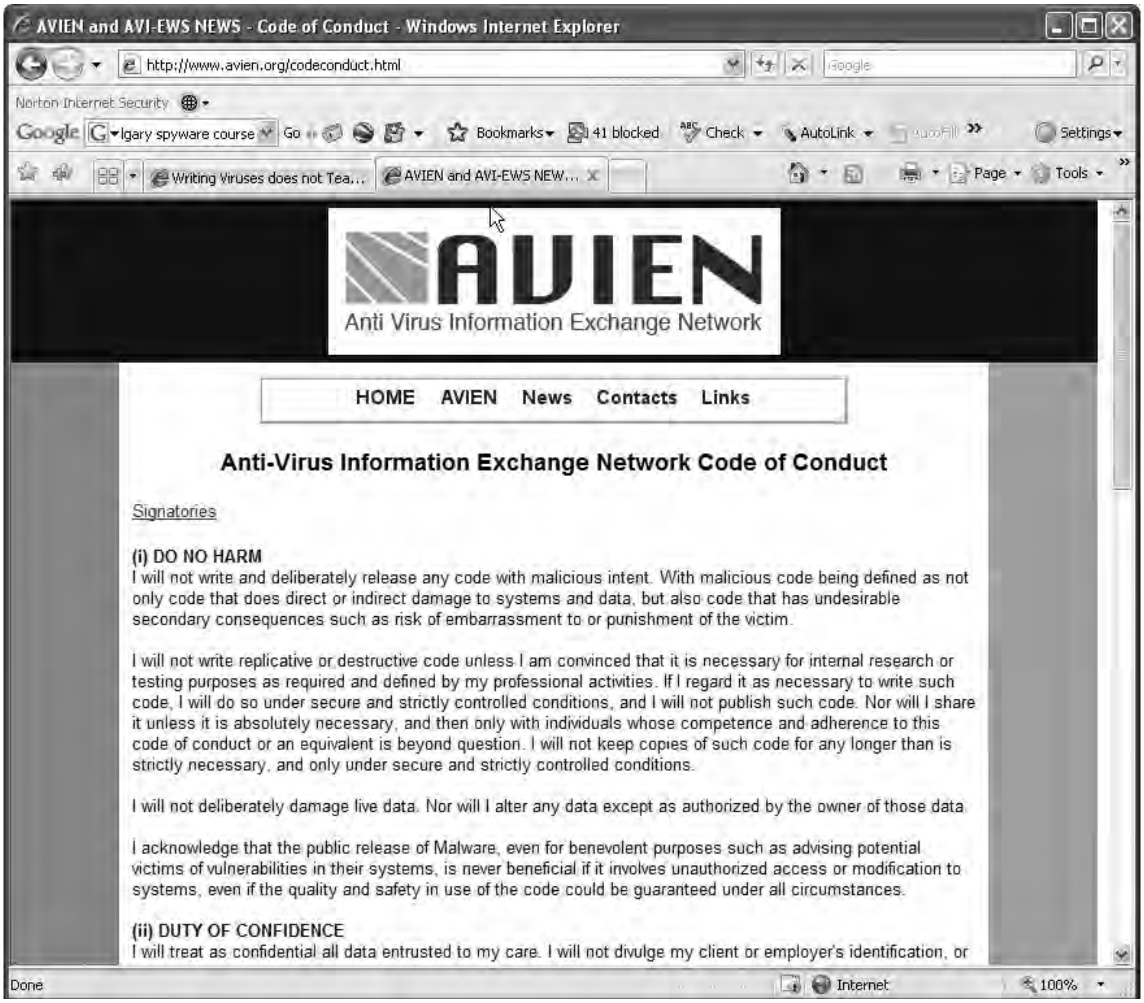
For our purposes, however, the most interesting parts of the paper may be those dealing with "The Next Age Virus Writer" (competent programmers who could make a living out of software development) and "The New Age Virus Writer." The latter term is used to refer to traditional virus writers whose motivation is as described above, but who are entirely focused on technically adept virus creation. But she also uses it to apply to amateurs experimenting with malware; "Many people in the workplace now feel that it is not only their right, but their duty to play around with viruses."

Notes from the Underground

Playing with Viruses

Is there a contradiction here? After all, this book has been written by people who frequently "play around with viruses" in the workplace (well, some of us!). We'd like to think that the difference lies in the word "amateur" (my word, by the way, not Gordon's.) Nearly all of us are in some sense Information Technology (IT) professionals, and the same can apply to members of this group. However, as AV/anti-malware professionals (whether from the vendor side or from a client organization), we are bound (or should be) by codes of conduct (for example the AVIEN code of conduct www.avien.org/codeconduct.html, as shown in Figure 5.1) and a commitment to meet professional standards as well as sound organizational policy. (You might also find Ken Bechtel's notes on procedures in the Defense-in-Depth chapter relevant to this issue, as well as the chapter on DIY Malware Analysis.)

Figure 5.1 The AVIEN Code of Conduct


AVIEN and AVI-EWS NEWS - Code of Conduct - Windows Internet Explorer
http://www.avien.org/codeconduct.html
Writing Viruses does not Tea...
AVIEN and AVI-EWS NEW...
AVIEN
Anti Virus Information Exchange Network
HOME AVIEN News Contacts Links
Done
Internet
100%

Anti-Virus Information Exchange Network Code of Conduct

Signatories

(i) DO NO HARM

I will not write and deliberately release any code with malicious intent. With malicious code being defined as not only code that does direct or indirect damage to systems and data, but also code that has undesirable secondary consequences such as risk of embarrassment to or punishment of the victim.

I will not write replicative or destructive code unless I am convinced that it is necessary for internal research or testing purposes as required and defined by my professional activities. If I regard it as necessary to write such code, I will do so under secure and strictly controlled conditions, and I will not publish such code. Nor will I share it unless it is absolutely necessary, and then only with individuals whose competence and adherence to this code of conduct or an equivalent is beyond question. I will not keep copies of such code for any longer than is strictly necessary, and only under secure and strictly controlled conditions.

I will not deliberately damage live data. Nor will I alter any data except as authorized by the owner of those data.

I acknowledge that the public release of Malware, even for benevolent purposes such as advising potential victims of vulnerabilities in their systems, is never beneficial if it involves unauthorized access or modification to systems, even if the quality and safety in use of the code could be guaranteed under all circumstances.

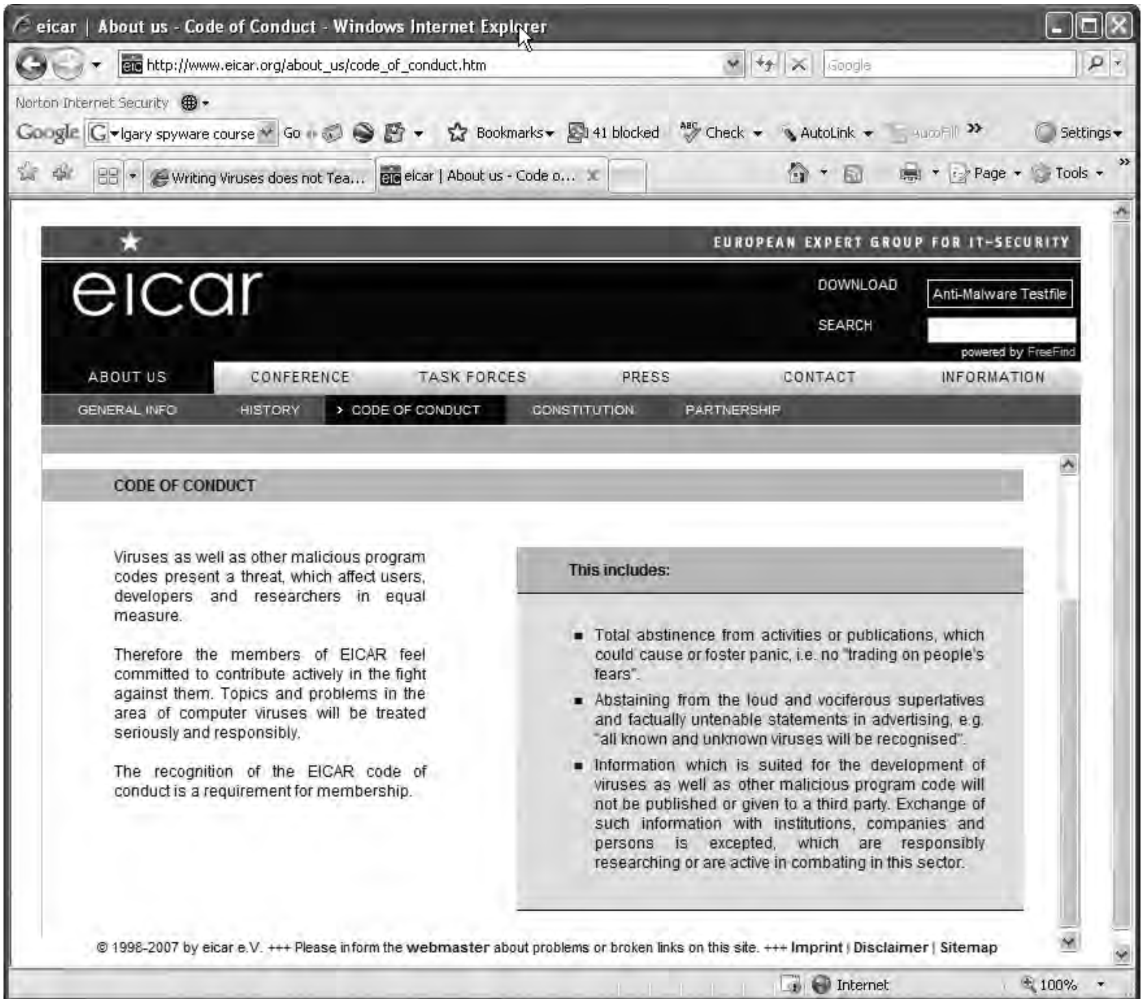
(ii) DUTY OF CONFIDENCE

I will treat as confidential all data entrusted to my care. I will not divulge my client or employer's identification, or

NOTE

By the way, it isn't obligatory for AVIEN members to sign up explicitly to the Code of Conduct, although they do have to conform to rules and guidelines that address some of the same issues, directly or indirectly. And, of course, most AVIEN and AVIEWS members are also members or employees of other organizations that do have an obligatory code of conduct (e.g., the WildList Organization (http://www.wildlist.org/conduct.html) and EICAR, formerly the European Institute for Computer Antivirus Research, as shown in Figue 5.2).

Figure 5.2 The EICAR Code of Conduct at www.eicar.org/about_us/code_of_conduct.html



CODE OF CONDUCT

Viruses as well as other malicious program codes present a threat, which affect users, developers and researchers in equal measure.

Therefore the members of EICAR feel committed to contribute actively in the fight against them. Topics and problems in the area of computer viruses will be treated seriously and responsibly.

The recognition of the EICAR code of conduct is a requirement for membership.

This includes:

- Total abstinence from activities or publications, which could cause or foster panic, i.e. no "trading on people's fears".
- Abstaining from the loud and vociferous superlatives and factually untenable statements in advertising, e.g. "all known and unknown viruses will be recognised".
- Information which is suited for the development of viruses as well as other malicious program code will not be published or given to a third party. Exchange of such information with institutions, companies and persons is excepted, which are responsibly researching or are active in combating in this sector.

© 1998-2007 by eicar e.V. +++ Please inform the **webmaster** about problems or broken links on this site. +++ **Imprint** | **Disclaimer** | **Sitemap**

This has been a very brief and inadequate summary of Sarah Gordon's very substantial body of work. Another paper that might be considered immediately relevant is "Virus Writers: the End of Innocence?" at http://researchweb.watson.ibm.com/antivirus/SciPapers/VB2000SG.html. There are links to many of her papers at her Web site, including, most relevantly to this section, http://www.badguys.org/papers/cybercrime.html. However, we now move on to the 21st century, and the ongoing professionalization of electronic crime.

The Black Economy

The term virus has been applied to replicative computer software for over 20 years now. It all started with small programs that used to attach their own code to normal, legitimate applications. Such behavior was compared to the "infection" method used by the viruses known to medicine, so the use of the name was "extended" to cover replicative software.

Notes from the Underground

Virus Origins

Surprisingly, Dr. Fred Cohen, the "inventor" of computer virology, doesn't seem to have been the originator of the term "virus" in this context. When Cohen first presented his ideas to a graduate class in information security, it was apparently his seminar advisor, Len Adleman, who suggested the parallel and therefore the name. (Adleman is better known as the "A" in RSA, the encryption algorithm: his colleagues Rivest and Shamir supplied the "R" and the "S.")

Cohen's first groundbreaking paper on the topic was published in 1984, and his dissertation in 1986 virtually defined the field of AV technology. We have yet to see an antiviral technique that isn't a variation on the technologies that Cohen defined. (See also his book "A Short Course on Computer Viruses," published by Wiley.)

However, replicative software precedes the publication of his work, important though it undoubtedly is.

As time went by, malicious software evolved and became worms, Trojans, and so on. That's what came to be called malicious software (malware), designating all kind of code capable of (or created with the aim of) affecting or infecting a system maliciously, without the user's consent or knowledge. Although the term malware was widely used by professionals (who are usually very insistent on using the term virus only for replicative malware), people carried on calling everything malicious a virus, the same way they had always done.

Up to that time, a lot of people used to ask the same question: what do virus authors get out of generating and distributing that code? The usual answer was as simple as the question: fame. They just wanted to be "recognized" in a certain "world" where the best coder is the "king of the world" in computing. Many people found it hard to believe that all that complicated work and research (at least, that was the popular view of the black art of virus writing) could be inspired purely by the quest for notoriety. Well, this is not exactly the truth. There were cases where a more tangible benefit was sought. That was the case, for example, with Trojans programmed with the intention of stealing login credentials from AOL users. There was a time when this kind of information was seen as particularly valuable and allowed some people to trade with it, obtaining economical benefits. In fact, this form of phishing continues in other contexts such as MySpace.

Around the end of 2004, the use of code intended to get an economic benefit started to become a normal practice. Then came the Big Bang, and the viral world changed dramatically. And now, the question was the same, but with a different answer: what do they get by generating and distributing that code? And the answer? Money.

This is the time when the term malware evolved into what the experts designated as "crimeware," referring to all kinds of code, primarily intended to obtain information leading directly or indirectly to a criminal profit. It is rather funny that with all these evolutions and changes, only security professionals commonly use this term. No matter what you do or how hard you try, some things never change...

Let us introduce some of the models that can be used for generating criminal profits, though there are far more possibilities than the ones presented here.

Spam

This is probably one of the simplest methods of making money from the Internet, and it certainly has one of the most peculiar names in this business. But the fact that it is simple does not mean it is ineffective.

In fact, how many people using e-mail have never received a spam message? I would almost dare to suggest none. The fact is that spam has been around in one form or another since the 1980s (depending on what we consider the first spam message. Most of the junk mail broadcast in the 1980s seems to have been chain letters, but that is another story) and it is still with us.

As we said before, spam is based on an action as simple as sending an e-mail (though "professional" spammers often use specialized software.) However, it has different characteristics from the e-mails we are used to dealing with. The two significant characteristics of a spam message that makes it different from a typical e-mail message are:

- The "message" it tries to transmit
- The volume of e-mails sent

The main objective of the spam is to "sell something." It does not matter if it is a bunch of pins, stock actions, pills or elephants. As long as someone is ready to buy it and someone else makes a profit from it, the job is worth it – to the spammer! (Hence the fact that one of the standard definitions of spam is that it "offers a disproportionate benefit to the sender of the mail.")

Are You 0wned?

Spam Defined

Paul Vixie, in his book "Sendmail: Theory and Practice 2nd Edition" (Paul A. Vixie and Frederick M. Avolio, Digital Press, 2001), defined an e-message as being spam if it met *all* of three conditions:

- The recipient's personal identity and context are irrelevant, because the message is equally applicable to many other potential recipients (that is, it isn't significantly personalized)
- The recipient has not verifiably granted deliberate, explicit, and still-revocable permission for it to be sent (it's unsolicited)
- The transmission and reception of the message appears to the recipient to give a disproportionate benefit to the sender

You might find the discussion of this definition at http://tqmcube.com/contrib.php useful, and the book is certainly of interest.

So let's get to the real point. A spam message contains an enticing offer to buy something that will help you in some way: health, money, comfort, or whatever you may need (or not). (Of course, there's no guarantee that the goods or services you expect to buy are what you think they are, or even that they exist at all.) On the other hand, who should an advertising message be sent to? Obviously, to someone who is interested in the article(s) sold. But how does an advertiser know who is interested in buying it? Part of the potential customer-base may be targeted, but a great deal of spam is sent purely on the basis of a list of addresses the spammer happens to have acquired. Anyone will do, actually, so messages are sent to "anyone," hoping they become interested in the article. The funny thing is that this "blind" part can be as effective as the targeted part. That's because when there are no significant sending costs, even a tiny percentage of positive responses may be worth having.

Legal "solutions," however, are generally based on the distinction between solicited and unsolicited mail (usually on the basis of opting in or opting out of mailing lists.)

Let me explain very simply why spam is not just a nuisance to the individual recipient. Spam is based on e-mail, and e-mails are sent using the Internet. The Internet consumes bandwidth, and the more bandwidth is used, the slower the communication. If too much bandwidth is consumed, communications may collapse. Have you ever suffered a situation of 10 minutes without Internet connection? So spam is a real problem, and if anything, it's getting worse rather than better. In fact, the problem is much wider than we've indicated here, in that spam is not just untargeted advertising material. The term is often applied to a wide variety of nuisances, including:

- Fraud:
  - Pump and Dump fraud
  - 419s
  - Phishing
  - Pyramid schemes
  - Job fraud
  - Mule solicitation
- Malware
  - Bot, Trojan, keylogger, and so on, dissemination
  - Mass mailers
  - Backscatter from spoofing malware
  - Misdirected malware alerts
- Hoaxes, chain letters
- Unsolicited commercial mail ("real" electronic junkmail)
- Other unsolicited bulk mail

We won't consider these further now, however, since spam is considered in more detail in Chapter 2.

Notes from the Underground

Spam with Spam

It was 1970 when Monty Python, the famous British group of comics, presented a sketch in which a married couple enters a cafe. When they look at the menu they realize that almost every single dish on it includes a meat-based foodstuff called "Spam." Along with the couple, there is a group of Vikings at the cafe shouting enthusiastically in praise of Spam. So, in the three and a half minutes the sketch lasts, the word Spam is used over 130 times. The use of all those repetitions of the word in such a short period of time has thus been compared to the high volumes of e-mails over short time periods that characterize spam. SPAM is a registered trademark of the manufacturer of the foodstuff, Hormel LLC (www.spam.com).

A Word about Dialers

Though these are a bit "old fashioned," they are still of interest for reasons we will explain later in this section. These are applications developed to "dial" a number, but in a covert way.

You have probably owned (in fact you probably still do) a telephone, which is capable of storing certain pre-saved telephone numbers. You just have to dial the number and save it by pressing a combination of keys on the telephone keypad. After that, all you have to do is press a number with another key and the telephone will automatically dial the number stored on that position of the "agenda." Easy and practical, isn't it?

Something similar can be done with your Internet-connected computer. If you want to connect to the Internet, you must "call" your Internet service provider (ISP) to tell him or her who you are and that you want to get connection. (That is, you authenticate yourself to the system.) Once the ISP has checked that you are the customer, he lets you go through and surf. To call your ISP, you replace the telephone with the computer, and instead of dialing the number (pre-saved or not) with a keypad, you use a modem. Normally, you only use one ISP so the telephone number does not change and you normally configure your computer to dial the ISP's number automatically. This is where dialers enter the scene. What they do is change that telephone number, so as to make your computer call a different ISP. What difference does that make? The new number uses a special and higher tariff, part of which is refunded to the provider, making a profit on every call.

Botnets for Fun and for Profit

With the arrival of broadband and DSL connections, dialers have been less of a problem. But many other problems have replaced them, many of them related to the rise of the botnets and therefore addressed in Chapter 4. Note, however, that the mail-related problems fit nicely into the general category of spam (419s, phishing, pump and dump fraud, and so on.).

- Click fraud (simulation of user clicks to commit fraud against Web sponsors and advertisers)
- Phishing
- Pump and Dump and other e-mail fraud
- Online games
- Login data
- Mule recruitment (tricking people into taking part in money laundering)
- General spam generation and relaying
- Malware dissemination, botnet building
- Adware and Spyware
- Ransomware (encrypting data and then demanding money for the decryption key)
- Demanding money to stop Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks
- Professional espionage (e.g., targeted Trojans to facilitate data leakage)
- Identity theft
- Pyramid fraud

These issues are explored in some detail in Chapter 2. The issue here, though, is that a whole black economy is based on botnets run by organized criminal groups. Among the many excellent resources at the Anti-Phishing Working Group's Web site at www.antiphishing.org, is a fascinating paper by Christopher Abad of Cloudmark on "The Economy of Phishing: a Survey of the Operations of the Phishing Market" (www.antiphishing.org/sponsors_technical_papers/cloudmark_economy_of_phishing.pdf), that describes the phishing marketplace as a "loosely connected group of forums where participants can trade goods, services, and money" and "the key goods are credentials." He proposes a model based on differentiated roles such as mass (these are people, not mass mailers in the sense of e-mail worms and viruses) template providers, server managers, and cashers. To this we might add finer-grained differentiation of roles to include bot programmers, mule-drivers, bot herders, and so on.

Now, however, Ken Dunham and Jim Melnick look at the operational history of the Network Crack Program Hacker (NCPH) hacking group.

"Wicked Rose" and the NCPH Hacking Group

Zero-day attacks, where an attack occurs before the vulnerability it exploits is publicly known, are a growing cause of concern for security professionals in the 21st century. An unprecedented number of zero-day attacks took place in 2006, most of them involving Microsoft Office documents. Ken Dunham, Director of the Rapid Response Team, and Jim Melnick, Senior Threat Intelligence Analyst, led the VeriSign iDefense intelligence team in tracking down Chinese hackers-for-hire, responsible for many of the attacks in 2006. "Wicked Rose" or "Rose Hacker" (MeiGei HeiKe) is the ring-leader of the NCPH hacking group, and this is the story of their maturation into a significant global threat.

Introduction to NCPH

NCPH has about ten members or associates. There were four core members as of 2006:

- (Wicked) Rose
- KuNgBiM
- Rodag
- Charles

There are also at least six other associates within the group, others who have a connection of some kind, as well as two additional positions (apparently unfilled) whose purpose is unclear. However, "Rose" or "Wicked Rose" seems to be the primary leader. Membership rules, recruiting goals, and standards are unknown. However, some members appear to be current or former students of Sichuan University of Science and Engineering (www.suse.edu.cn and www.study-in-china.org/school/Sichuan/suse/).

The group is believed to be responsible for the development and deployment of exploit code related to vulnerabilities in Microsoft Word (Malformed OLE Structure Code Execution) and Microsoft Excel (Malformed BIFF Structure Code Execution.)

Public Knowledge of a Zero-day Word Exploit

The story of NCPH zero-day attacks began, as far as the public are concerned, on May 18, 2006. On this day, the Internet Storm Center reported a new attack, possibly a zero-day attack. iDefense worked closely with the SANS Institute and other organizations to analyze the changing threat landscape as it related to exploitation of this vulnerability. Within the next 36 hours, iDefense gained access to multiple codes and extracted a new rootkit called GinWui. Independent research suggested the following:

- Exploitation targeted a new vulnerability that allowed attackers to exploit computers running fully patched versions of Microsoft Word 2002 and others successfully.
- Exploitation dated back to May 12, 2006, and involved at least six unique hostile exploit files. iDefense confirmed that attacks targeted two organizations, one in the United States and one in Japan.
- The Chinese-authored rootkits GinWui.A and GinWui.B were used in several attacks. iDefense identified the rootkits' source and authors as being Chinese: he ain't a Thespian "Wicked Rose" and others profiled later in this case study.
- Successful installation of the rootkit requires Administrator or Debugger rights. Initial exploitation, however, does not require Administrator rights.
- iDefense identified unique malicious code attacks pointing to nease.net, and authored several Snort signatures for detection of this traffic. iDefense continued to monitor other domains related to the attack.

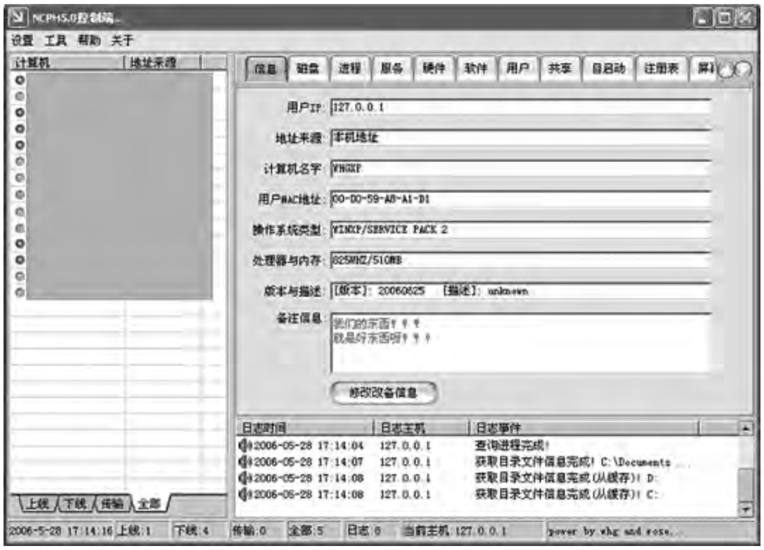
The original attack upon a large Department of Defense (DoD) entity within the USA began on May 12, 2006. Targets were apparently selected by the attacker on the basis of a Google search. Three variations of a Microsoft Word zero-day attack were involved in the attack. A few dozen attack files were first distributed to less than a dozen targets, to identify which version was in use within the organization.

Once attackers had identified the vulnerable version of Microsoft Word in use, close to 200 messages were sent out to multiple targets in the organization within 24 hours. This second wave of attack was distributed as "Planning document 5-16-2006.doc." This attack code was enhanced to a higher standard than the earlier variant, which had been sent out earlier to identify the vulnerable Word version . A third attack commenced on May 17, 2006. During this period, the Internet Storm Center and others got involved and the case became public. In the end, iDefense identified six unique samples, of which three are more prevalent than other variants.

The GinWui Backdoor Rootkit Payload

Zero-day attacks commencing in May 2006 attempted to install GinWui, a backdoor Trojan horse and Windows rootkit. A Dynamic Link Library (DLL) file called *winguis.dll* and several SYS files installed themselves when a computer was successfully attacked through an exploit. Two versions of the GinWui rootkit were installed during several attacks in May and June 2006.

Figure 5.3 NCHP 5.0 Screenshot (GinWui Rootkit)



```
用户IP:      127.0.0.1
地址来源:    本机地址
计算机名字:  WHGXP
用户MAC地址: 00-D0-59-AB-A1-D1
操作系统类型: WINXP/SERVICE PACK 2
处理器与内存: 825MHZ/510MB
版本与描述:  [版本]: 20060625   [描述]: unknown
备注信息:    我们的东西!!!
             就是好东西呀!!!

日志时间              日志主机     日志事件
2006-05-28 17:14:04   127.0.0.1    查询进程完成!
2006-05-28 17:14:07   127.0.0.1    获取目录文件信息完成! C:\Documents ...
2006-05-28 17:14:08   127.0.0.1    获取目录文件信息完成(从缓存)! D:
2006-05-28 17:14:08   127.0.0.1    获取目录文件信息完成(从缓存)! C:

2006-5-28 17:14:16 上线:1  下线:4  传输:0  全部:5  日志:0  当前主机:127.0.0.1  power by whg and rose...
```

Wicked Rose is the author of the GinWui malicious code. His code and support posts related to GinWui distributions exist on the Chinese NCPH and Evil Octal forums. Wicked Rose communicated and was associated with WHG and others on this forum. WHT hosted version 3.0beta.3 of the NCPH remote control rootkit code on May 2, 2006. This distribution of GinWui was largely unknown and undetected by AV companies at the time of release.

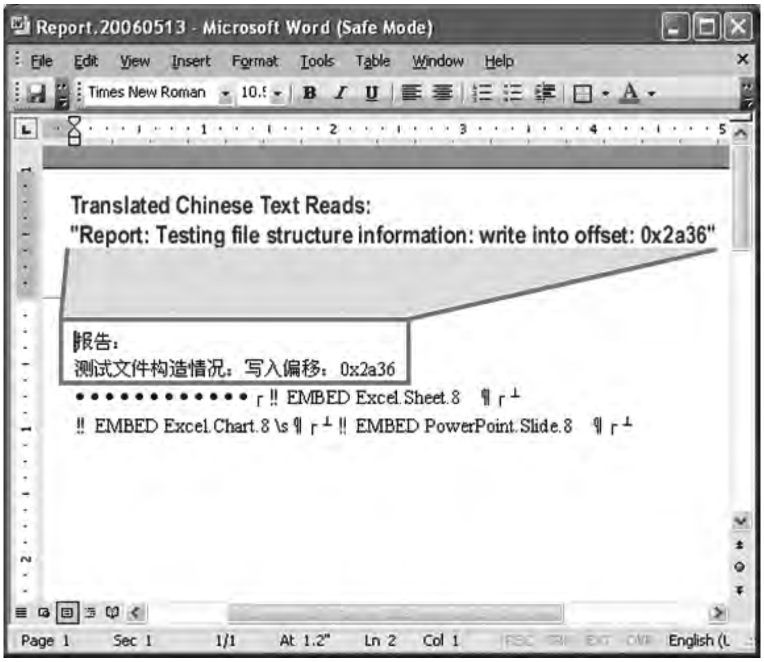
Versions of GinWui used in targeted attacks of May and June 2006, were private versions not released to the public. This indicates that Wicked Rose either constructed the zero day attacks or sold private code to other coders users who performed the actual attack.

Wicked Rose later documented additional updates to his rootkit code, version .50. By this time Wicked Rose was performing full-time development of this malicious code as a hacker for hire.

June 21, 2006-2007 - Continued US Targeted Attacks

Just over a month later, following initial, exploratory, GinWui-based targeted attacks, another Microsoft Word exploit occurred on June 21, 2006. A spoofed e-mail containing a hostile Microsoft Word document was sent to a target. Analysis of the attack suggested that a test file was used to identify what version of Word might be running within the targeted organization, rather than using a refined targeted attack upon a known version of Microsoft Word. Chinese text within the Word document reveals Chinese characters discussing a systematic evaluation of offsets for Microsoft Word exploitation:

Figure 5.4 RipGof Attacks Reveal a Chinese String Related to Systematic Testing of Offsets for Exploitation


Report.20060513 - Microsoft Word (Safe Mode)
File Edit View Insert Format Tools Table Window Help
Times New Roman 10.5
Page 1 Sec 1 1/1 At 1.2" Ln 2 Col 1 English (L

Translated Chinese Text Reads:
"Report: Testing file structure information: write into offset: 0x2a36"

报告:
测试文件构造情况: 写入偏移: 0x2a36

```
••••••••••••• ┌ ‼ EMBED Excel.Sheet.8 ¶ ┌ ┴
‼ EMBED Excel.Chart.8 \s ¶ ┌ ┴ ‼ EMBED PowerPoint.Slide.8 ¶ ┌ ┴
```

Backtracking Targeted Attacks: RipGof

In June 2006 another targeted attack emerged, but it was not GinWui that was used this time but a new program, RipGof.B. The attack attempts to exploit MS06-027 to install RipGof.B, a Trojan horse. This is the same exploit code used in the previous zero-day attacks linked to Wicked Rose and NCPH. The exploit code was still private at this time, indicating that the author of both the GinWui and RipGof attacks is the same individual or group, or affiliated through underground criminal operations.

RipGof.B is an improvement of the exploit formerly used in GinWui attacks. RipGof.B attacks included improvements to shellcode that attempts to fork to different locations, based upon the address value of the stack, in order to exploit multiple versions of Microsoft Word. Once installed, RipGof.B attempted to connect to enjoy.irdet.com and enjoy.bmwsee.com over TCP port 80 HTTP. It ran as a rootkit and backdoor Trojan horse, and "phoned home" with stolen data to a Chinese server.

RipGof malicious code did not exist as a distribution in the underground, which led investigators to look into the original RipGof.A malicious code. Over a year prior to the 2006 targeted attacks, RipGof.A had emerged into the wild. RipGof.A attempted to exploit the Jet Engine Database exploit (www.microsoft.com/technet/security/Bulletin/MS04-014.mspx) in March 2005. This proved that attempts had been made at exploitation and installation of code through RipGof for a year prior to more sophisticated codes and attacks.

In summary, RipGof and GinWui attacks both used the same unpublished exploit code against Microsoft Word, and both installed rootkit-based codes to steal and send information back to Chinese sources. This circumstantial evidence reveals that Wicked Rose and the NCPH group are likely to have begun their exploitation efforts at least a year and a half to two years before the sophisticated attacks that commenced in 2006. Once the group found a vulnerability within Microsoft Word, they were able to improve upon it and on their targeted attack techniques, in order to distribute multiple targeted attacks and malicious programs for criminal gain.

Timeline of Events

Wicked Rose and the NCPH hacking group were implicated in multiple Microsoft Office-based attacks over a two-year period. An attack in 2006 used RipGof.B. RipGof.A had first emerged a year earlier, using a relatively unsophisticated exploit. Over the next year, the Evil Security Team, also out of China, created the Dasher worm and used the PcShares Trojan in an attack. Wicked Rose gave a recommendation on the Trojan the day it was updated in the spring of 2006, demonstrating a close affiliation between Wicked Rose and the Evil Security Team actors. Multiple attacks that took place in May and June and later in 2006 were related to privately held exploit code for both Microsoft Word and Excel, known to have been developed by Wicked Rose.

Notes from the Underground

Wicked Rose Timeline

A timeline of proven events related to Wicked Rose attacks follows:

- April 22, 2005 – RipGof.A JetEngine DB Attack
- Dec. 19, 2005 – Dasher worm and PcShare Trojan attack by Evil Security Team
- April 27, 2006 Update to windowsupdates.net attack site
- April 30, 2006 – Wicked Rose drops out of school
- May 2, 2006 – 3.0beta3 NCPH remote control (GinWui) public release

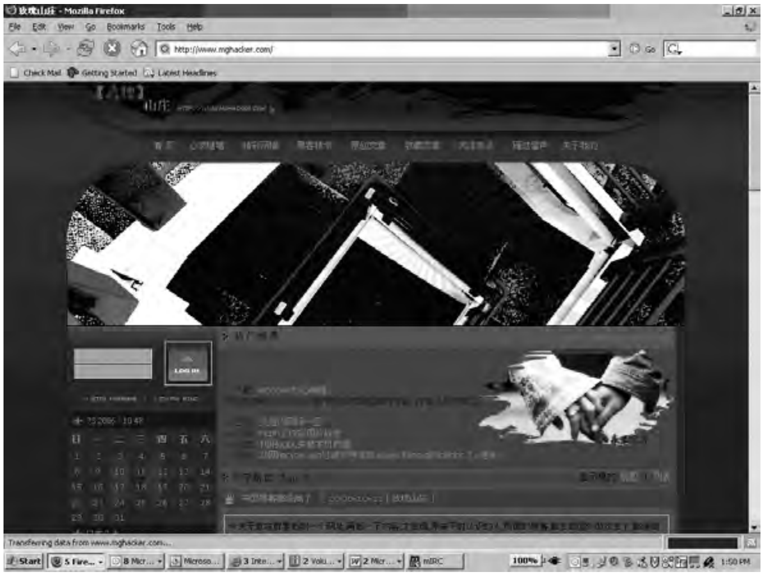
Continued

- May 12, 2006 –Initial probing and GinWui.A exploitation attempts against US target
- May 15, 2006 – PcShare Trojan update recommended by Wicked Rose on day of new release
- May 16, 2006 – Update to windowsupdates.net attack site
- May 16, 2006 – Multiple GinWui.A attacks against US target
- May 18, 2006 – SANS reports zero-day attack
- May 19, 2006 Update to windowsupdates.net attack site
- May 20, 2006 – GinWui.B Attack
- May 20, 2006 – WZT Kicked out of NCPH
- May 29, 2006 – GinWui.C attack
- June 1, 2006 Update to windowsupdates.net attack site
- June 9, 2006 – Mdropper.F attack
- June 14, 2006 – Daserf.A attack
- June 15, 2006 – Mdropper.G attack
- June 15, 2006 – Booli.A Trojan attack
- June 16, 2006 – Flux.E attack
- June 18, 2006 – RipGof.B attack
- June 23, 2006 – PPDropper.A
- June 23, 2006 – Booli.B Trojan attack
- June 25, 2006 – GinWui.D attack
- June 26, 2006 – GinWui.E attack
- Sept. 27, 2006 – PPDropper.F attack
- Sept. 30, 2006 – GinWui.G attack
- Oct. 9, 2006 – Wicked Rose reports pay increase; likely in September

Introduction to Wicked Rose and NCPH

Just who are Wicked Rose and the NCPH hacker group? They are a group of college or former college students in China who apparently room with one another and regularly support each other's mutual hacking interests. In-depth research implicates Wicked Rose as the ring-leader of the group, responsible for managing hacker-for-hire relationships, and paying group members for their work as hackers. During the time of targeted attacks in 2006, their income increased significantly, to an equivalent of full-time wages for part-time hacking.

Figure 5.5 Wicked Rose's Web Site



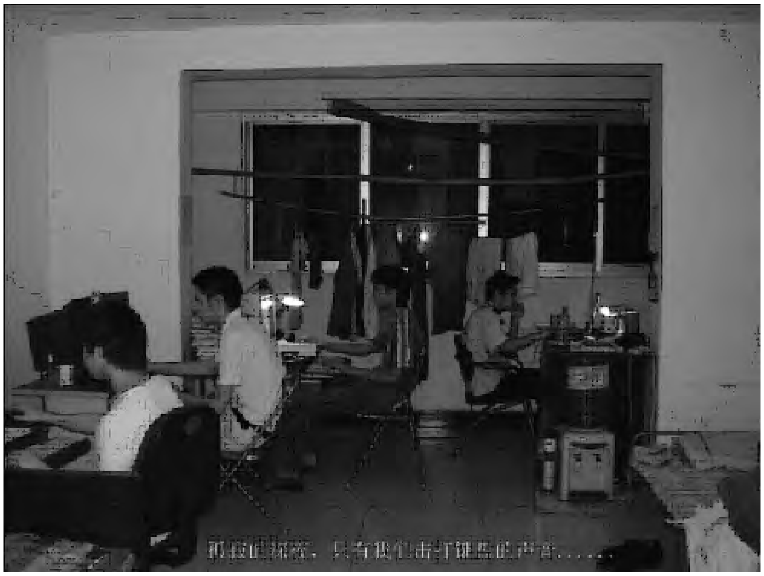
Rose is approximately 20 years old (as of 2006), a student at the Sichuan University of Science & Engineering. In the spring of 2006, Wicked Rose claimed to have dropped out of school in favor of full-time hacking opportunities. Specifically, his blog entry on April 30, 2006, claims that he did not register for his university exam. He performed significant updates to his rootkit code from March through June 2006. He later returned to school, apparently by September 2006.

Wicked Rose claimed responsibility on his blog for targeted e-mail based attacks containing Microsoft Word and CHM exploits, from the spring of 2006.

Figure 5.6 NCPH Studio Web site www.ncph.net

Registration information for ncph.net reveals a Chinese registrant. The main location of the NCPH group is in Zigong, Sichuan Province, in south-central China. The NCPH group (NCPH Studio) in Zigong, China, is shown here:

Figure 5.7 NCPH Hackers at Work in the "NCPH Studio": Left to right: "Wicked Rose," KuNgBiM, Charles and Rodag



Additional photos featuring Wicked Rose and NCPH hackers below are captured from their various Web sites and blog entries in 2006. Chinese translations for each photo are shown on the following page.

How Did NCPH Begin?

A fascinating posting dated January 5, 2006, on the Web site of the Sichuan University of Science and Engineering (SUSE) provides many insights into Wicked Rose and the founding of the NCPH group (http://jkx.suse.edu.cn/ShowInfoContent.aspx?Table=stu&ID=33). It is less clear, however, in explaining the group's motivations in attacking Microsoft Office vulnerabilities. According to the posting, Wicked Rose's real name is Tan Dailin. He reportedly became involved in patriotic hacking in 2001, and "participated many times in foreign network attack and defense," later working on a number of overflow attacks for various Microsoft-related vulnerabilities. In September 2005, he participated in a special military-related competition in network attack and defense in the Chengdu military region, where he scored very highly. He did so well that he then became involved in training and in developing dozens of sets of special hacker software for similar competitions. He also earned a special monetary bonus. He later got the idea of launching his own group, and recruited three other hackers to join him, calling the new group "NCPH." The group apparently has various companies that are clients. It is still unknown who is paying NCPH to develop Microsoft Office exploits and why they are doing so. Obviously, Wicked Rose has strong ties to the Chinese military (and, by extension, one assumes, the Chinese government), but NCPH's "employer" could also be an information broker of some kind or some other third party. The most troubling aspect is that this new approach to hacking in China has moved way beyond the patriotic hacking of the past into the realm of very targeted attacks.

Not everyone thinks that NCPH is the originator of the scores of MS Office vulnerabilities and exploits discussed here. In discussions between one of the authors of this section with white hat and information security researchers in Beijing in April 2007, some were of the strong opinion that, while NCPH hackers were purveyors of the programs they were probably not the original authors of the zero-days. The truth may be somewhere in between. Perhaps they developed some and merely facilitated others.
Figure 5.8 shows "Wicked Rose", Charles, Ronag, and KuNgBiM.

Figure 5.8 Wicked Rose and NCPH Hacking Photos

"Wicked Rose": From an ancient Chinese poem, expressing the devotion of his heart for hacking.

"After you choose the technology you love, you have to research every system and code everyday!"

Figure 5.8 Continued

Charles

"Silence belongs to our world..."

"Charles always laughs so brightly when searching for program problems!"

Ronag

"Behind every successful design, he always has a slight smile..."

KuNgBiM

"Only we can feel this kind of happy..."

- **WHG ("Fig")**

WHG is not a core member of NCPH, but a close affiliate of Wicked Rose. WHG appears to have been central to development of the NCPH rootkit, aka GinWui. WHG was credited by Wicked Rose as one of the authors of this malicious program. WHG is an experienced malware author whose real name may be Zhao Jibing (赵纪斌), and is believed to be employed in the Sichuan province of China.

WZT

WZT is a former member of the NCPH group who was kicked out during the time of zero-day attacks in May 2006. During this time, the zero-day attacks were publicly disclosed, increasing pressure upon the hacking group. It is feasible that WZT may have offended the group in some way related to zero-day attack techniques, or conflict or competition over hacker-for-hire deals or leadership control.

WZT was formerly a coding expert within the NCPH group, and has many years experience in hacking. He is responsible for creating multiple tools and regularly giving credit to one of the most famous hackers in China, Li0n, who was also the founder of the Honker Union of China (HUC) hacking group.

The Jiangsu Connection?

WHOIS registrant data for hacker sites and related domains used within attacks, revealed a connection with the Jiangsu Province of China. One domain, windowsupdates.net, was used in attacks and resolved to an IP address in the Sichuan Province. Meanwhile, the registrant "zhaofeng network" was reportedly based out of Jiangsu, not Sichuan. Some of the WHOIS information clearly contained fraudulent information, presumably to divert researchers away from the true identity and location of the attacker responsible for registering the hostile domain. The connection to the Jiangsu Province remains unclear.

The China Syndrome

Prior to Wicked Rose and NCPH hacker-for-hire attacks in 2006, Chinese hackers were mostly known for their patriotic hacking and related hacktivism. This disturbing development reveals two critical changes:

- The motives of some Chinese hackers are changing
- As of 2006, some Chinese hackers are regularly associated with sophisticated and targeted attacks

Wicked Rose demonstrates his involvement in his early blog entries and Web site posts in 2006, and before. An unknown company or entity reportedly paid Wicked Rose for hacking at the rate of 2,000 RMB (Renminbi, also known as "yuan") per month, about

$250 USD. At this time, Wicked Rose gave 200 RMB to NCPH hackers and kept the rest for himself. Once targeted attacks took place, the payment increased five-fold to 5,000 RMB monthly, with $1,000 a month going to NCPH hackers. This is a significant amount of money in China, effectively paying hackers a full-time wage for part-time hacking.

Throughout the summer of 2006, while Wicked Rose was not in school, over 35 zero-day attacks, proof-of-concept programs and attacks against un-patched Microsoft Office vulnerabilities were discovered in the wild. With Wicked Rose claiming responsibility for early attacks, being the lead author of GinWui, and leading the NCPH hacking group, there is little doubt as to the depth of his involvement in attacks to date.

By the end of 2006, attacks became increasingly sophisticated. In one instance, a popular PowerPoint file (previously distributed during the Christmas holiday season for two years running) was used within a social engineering attack upon one individual within an energy sector US-based company. The PowerPoint file was modified to include an exploit that silently installs malicious code. This same individual received another e-mail containing a Microsoft Word exploit. In this case, only one individual within the company was targeted and with just two messages socially engineered for maximum success. This was a much more targeted and stealthy approach for attacks compared to the earlier attacks performed by the group in the late spring of 2006.

NCPH continues to be a significant threat going forth for several reasons:

- Attacks continue to take place in the wild and are very difficult to identify on a targeted basis. Only the most sophisticated networks and system administrators are able to offer proper protection and capture targeted attack files before they achieve their aims.
- NCPH is a serious and dedicated hacking group; methodical and disciplined in their development of new exploits and attacks.
- NCPH is motivated by both the thrill and challenge of hacking and by profit.
- Attacks by the group are highly targeted and stealthy, being difficult to detect and remove.
- Whether they were the originators of zero-days and exploits in some cases (as disputed by some in China) or merely the purveyors of these programs the threat posed by the group remains very real.
- It is not known at this time whether other groups like NCPH may be forming in China. If it continues to be successful in meeting its goals, it may become (or may already be) a new template for similar Chinese hacking groups.

And now, Enrique makes some predictions about likely future targets.

Lurkers in Your Crystal Ball

When we plan some kind of project, we need at least a basic knowledge of the topics involved. Experience is probably the best teacher in most aspects of our lives, and security is not an exception. However, when we are facing new technologies, past experience is not so useful, though it does still have benefits if we can extrapolate from old experiences to new scenarios. So new projects based on new technologies imply letting our imaginations fly and peer into our particular ”crystal ball.”

That's what I will try to persuade you to do in this chapter: change your mindset, get into the "dark side" of security and try to guess what the attackers will try to do in the "near" future in order to get access to a private system, obtain private information, and so on. I know no security expert (or even non-expert), who possesses a crystal ball. On the other hand, I know quite a few experts and some amateurs (no matter what topic we are talking about) who are perfectly capable of imagining some of the things the future may bring. It is obvious that we do not have to be fortune-tellers or "gurus" to imagine some of the things that will happen in the near future, but knowledge of the field in which we are interested, is certainly helpful.

Things That Will Not Change (Much)

I'll take this section in two sections. In the first, I look at general issues that arise from human nature in general. In the second, I consider some specific technologies and look for current patterns that may tell us about future trends.

Social Engineering

Anyone who has been a medium e-mail user during the last three or four years has experienced this phenomenon. What is it and what makes it such a successful attack?

Social engineering is a very simple technique, consisting of catching people's attention by addressing them on a topic of interest and convincing them that what you are saying is true. Considering the results it has attained in the past (and the ones it is still achieving), it does not seem to be too difficult. That's why we have defined it as a "simple" technique. Let us go through some examples. Some of the subjects selected for this technique are: sports, pictures (especially pornography related ones), funny videos, false credit card charges, and computing.

If you are a sports fan and never miss a football match (soccer, for our American friends), and you receive an e-mail claiming that you can get (for free, of course) a couple of tickets to see the World Cup final, would you hesitate to have a closer look at the content of the e-mail? The same happens with computing. If you are a network administrator, a security manager, or simply a computer user conversant with good practice, and you receive an absolutely professional looking e-mail, pretending to come from a very reputable company or source telling you that something malicious is going around the world and that they provide prevention or a solution for it in an attachment, would you at least consider executing it? I think you would.

But, how does the sender know that the selected topic is one that will tempt the recipient to execute the attachment? He simply does not know. He just sends loads and loads of e-mail to loads and loads of people, hoping that someone will open the one they receive. Let us not forget that human beings are very predictable in some ways and, by nature, we are especially curious. Some topics never get out of fashion, and there is always someone interested in some others. So some success at snaring victims is almost guaranteed.

As our nature hasn't change in this aspect, the coming years will not see a bigger change in principle. However, it all depends on how much use is made of the technique in question. If the attackers keep on using it, some recipients will keep on falling for it. If the technique falls "out of use," it does not matter what human nature is, we will be deprived of the opportunity to fall for it. There's no doubt, though, that social engineering in its broadest sense is not going to fall out of fashion.

Notes from the Underground

Social Engineering for Beginners

People hacking, as this technique is often referred to, is a lot more than dodgy e-mails, though it is considered part of the toolkit used by mass mailer viruses and phishing. David Harley uses the definition "Psychological manipulation of an individual or set of individuals to produce a desired effect on their behavior." Note that this includes not only social engineering in the sense of confidence trickery, but also takes in the older, more sociological and anthropological sense of the term, so that we can apply it in such contexts as changing social behavior through legislation, or reducing end-user's susceptibility to security attacks through policy and education. In this sense, the term can also be used to refer to a whole range of attacks, from password stealing to money mule solicitation, from rootkit installation to phishing.

Social engineers exploit a number of human characteristics (we hesitate to say weaknesses: in some contexts, trust, for example, is a good thing.) David Harley describes "Seven Deadly Sins" that make social engineering easier (there are many more, of course, and the model has changed since it was first presented in 1997):

- Gullibility (trust)
- Curiosity
- Courtesy (desire to be helpful)

- Greed
- Diffidence (timidity, apathy)
- Thoughtlessness (irresponsibility)
- Vanity

These attributes make us more susceptible to being engineered by tools such as:

- Flattery
- Deceit and trickery
- Threats and bullying
- Misdirection (for example, an essential untruth made more convincing by circumstantial detail)

Back in Fashion

Fashion changes every season and, as someone said, "fashion designer*s change things enough to make sure everything from last season is useless*." But, if you keep your clothes long enough, there will come a season when you will not need to spend much, because the same clothes you wore years ago, will be in fashion again and you will be wearing "the latest collection." Assuming they still fit, of course.☺

Something similar seems to happen with malware. If you talk about DOS viruses infecting pure *.exe* and/or *.com* files, people look at you as a dinosaur. You seem to be talking about something that disappeared so long ago. Even if you talk about windows file infectors, it seems that you are "old fashioned." Infecting files as a normal behavior was "out of fashion" by the early years of the 21st century. Everything was pointing, basically, towards mass mailers, worms, and assorted Trojans. Yes, there were exceptions, and there will always be, but the general idea was that infecting other files was no longer "cool." However, by the middle of 2006, a new wave of file infectors reappeared. They received (as usual) several names, depending on the company detecting them (Viking, MyTob, Looked, and so on), but the important thing is that they were active, there were several variants of them and they did (or some of their variants did) infect, something "out of fashion" by that time.

Another category that got back onstage was the macro viruses. A bunch of "Kukudro" (aka Lafool) viruses using macros to exploit a vulnerability in Microsoft Word, appeared for a few weeks in 2006, giving some time between variants. Microsoft Office documents are also a common vector for targeted Trojans, embedded in a Word document or Excel spreadsheet, for instance, as we saw in the "Wicked Rose" case study.

So, as we can see, things get repeated in security as in fashion. Same look with a different "touch," but using a technique or vector we already know. This is a phenomenon we will keep seeing in future malware. The new "touch" will probably change according to current around exploits, but not looking for the new zero-day ones, but using the "old fashion" ones. The point of this is that even though patches and new versions have been published for known vulnerabilities, many people have still not learned to patch, so the effectiveness remains more or less the same, at least among home users and small businesses. Let us look at the example of "Netsky.P." It came out on 2003, exploiting a vulnerability patched in 2001. In 2006, it was still In the Wild (ItW) and one of the most common viruses around the world. I certainly do not want to spread malicious ideas, but if I was a "bad guy," I would revisit older techniques as well as attempting to be innovative.

Botnets

During 2003 and 2004, we witnessed a tremendous upsurge of "worm waves." They spread at a very high speed, in big volumes. The more systems they could reach, the more successful the wave was. Therefore, they achieved more fame. By the end of 2004/beginning of 2005, bots had become more and more spread around the world. Their main objective was to reach as many systems as possible. The difference from the worm waves of previous years, was that the authors were not looking for notoriety this time. In fact, they tried to be as quiet as they could, so they could create a real network spread as widely around the world as possible. In this way, their "dark" activities, usually involving fraud and other criminal activities, stayed effective for longer and were therefore more profitable.

In the coming years, unfortunately, we will not be able to get rid of the bots and the networks that link them. However, we may see a change in their behavior. A big machine is noisier than a small one. In the same way, a big botnet often generates much more traffic than normal, so might be discovered by the "noise" generated. But if they reduce their volume and include a smaller number of machines working together, their traffic will be lower and their chances of survival increases.

On the other hand, this would work on botnets where all the machines work "together." A well-designed botnet is able to let the machines receive instructions independently and make them work in sub-groups, big enough to be effective or small enough not to make too much noise. David Harley and Tony Bradley consider bot issues in more detail in Chapter 4.

The Shape of Things to Come

So much for things that will change in detail, rather than in broad outline. Now, let us look at one or two upcoming technologies that may present us with new types of technical problem.

Communication: A Common Problem

In the beginning, only DOS existed (and we mean at the very beginning), for PCs in the business world. OK, there were actually loads of different machines, operating systems, and even some possibilities of connection at that time, and some will remember with nostalgia the first Apple computers, the Commodore Pet and 40 shades of CP/M. Still, bear with this assertion for the moment, and we will see how it takes us to the point. When DOS (MS-DOS and PC-DOS) was "the one and only" common, popular, standard system and before viruses appeared, the risks were small. In fact, there were no risks. Ok, that is not exactly true either. We were all afraid that our floppy disks could somehow get folded, or that they could get into the influence of a powerful magnetic field, which could destroy the applications, games, or data contained within. Not to mention what a bit of unfortunate rain could cause if it happened to touch their recorded surface. But that was all we had to fear. No one cared about being attacked from the outside or about being "hacked," except for worrywart system administrators worried about internal/external data leakage through a very limited range of exit points, and the occasional joke program or Trojan. But these hardly touched the lives of the average end-user or even those lucky individuals who had one of the few (and very expensive) home systems then available. There was no connection between the machines apart from the floppies exchange the owners could establish between them, apart from hardwired or telephone connections using arcane communication and file transport protocols.

NOTE

It's been a long time since we had much to do with scriptable, text-interface focused communications and file transfer protocols, but we remember "kermit" with particular fondness. And yes, it's still out there. See: www.columbia.edu/kermit/kermit.html.

Life was difficult, because we could almost immediately pass information to anyone we knew simply by using our telephone, but we could not give anyone a picture, a document, and so on, unless they lived next door and we could hand it to them over the backyard fence. Yes, that was a problem. We prayed for "the future." In the same way that the telegraph inspired eager communicators to move towards the development of the telephone, we were waiting for the computers to be able to communicate to each other and make life easier. When the first viruses appeared, their only way to spread was by infecting either the boot sector of a floppy disk, or the files contained within the disk. Thanks to direct "communication" between users and the kind sharing process of their files, the viruses could spread around. Then people became worried about what they let into their machines, and they were scared that their neighbor might not only give them a game over the "fence," but also a virus that could destroy all the information they kept on the computer.

After some time, the miracle happened. The Internet arrived and we were "connectable" to anyone and everyone who could connect to us. That was the beginning of the end.

All of our problems increased as we became more interconnected. The malware was now able to spread by itself and it became less necessary to trick the user into doing the job. Nowadays, being "unplugged" or "disconnected" is not an option. So, get ready for the worst...

Automobiles

The automobile was a great invention from the point of view of making distances short, and that remains their main duty. However, first the inventor and then all the subsequent manufacturers wanted to make it more functional and more comfortable: headlights, heating, luxury seats, music and "a few" more features and capabilities were added throughout the years. And then they became intelligent, self-regulating the speed, the petrol consumption, and so on. An intelligent machine? That sounds very much like a computer. Then the possibility arises that my car can suddenly stop, give me a "blue dashboard," get infected with a virus, or even be hacked. Well, we're not at that point yet but that is another possible vision in our crystal ball.

The good news is that automobiles do not run on a Windows system. The bad news is that some of them already have Bluetooth. Therefore, there is at least one possible means of transmission for future malware trying to jump from one car to another.

NOTE

You could, however, check out Rob Rosenberger's post at Vmyths regarding the comments of several AV manufacturers in 2005, following reports that Lexus vehicles could be affected by a hypothetical virus. Bill Ussery, Lexus Product Communications Manager, was obliged to point out that while navigation systems in Lexus and Toyota vehicles use an embedded operating system, it isn't the widely exploit Symbian OS, and that the Bluetooth interface used doesn't support data export from the navigation unit.

OK, let us assume the malicious code exists (admittedly a large assumption) and that the means of transmission (whichever it is) is good enough as to make it travel around. What can malware do in my car? It is not going to infect my documents and it is certainly not going to steal my credit card number. Well, not at the moment. However, we can already think of a couple of different scenarios:

"Benign" or Accidental Malware

This would be software which affects our car but not with a "bad intention," so it can't be described as malware. It might be able to detect if a system is on or off, and switch it to the opposite state. Such systems could include radio, indicators, lights, wipers, air conditioning, windows (not Microsoft's!), anything... The risk is that it might also be able take the data from our "on board computer" (average speed, mileage, and so on) and send it over to the car next to us, reading at the same time that car's data and transferring them to our computer. That would give completely wrong readings on each car.

Truly Malicious Malware

This time the malware would be intentionally malicious. It could stop the car or turn off all kind of security warnings (such as running out of petrol, engine overheating, low on oil, and so on) that could lead to a partial or complete breakdown or, even worse, a crash.

As a further step, someone might create cross-platform malware affecting the car systems (no matter if its intentions are good, bad, or catastrophic), which would be cross-platform. In such a situation, it would be able to transmit, for example, to a mobile phone via Bluetooth. Bluetooth has a short range (but long enough to cover a few meters), but a car-borne infected system might be able to infect pedestrians' mobile phones.

These situations may seem too "Hollywood" to take seriously right now, but the important thing is that car computers are an unexplored and unexploited mine of opportunity. Their implementation within motor cars is already happening, and their interaction with mobile phones, hand held devices, computers and other systems is increasing in frequency. So we will see more discussion of risks, and maybe real problems.

VoIP

One of the technologies becoming more and more popular is Voice over IP (VoIP). This simply uses the data nets (IP is the "Internet Protocol," part of TCP/IP, the backbone of the Internet) to transmit voice traffic. This system saves a lot of money, as it can share Internet and telephony connections. It is a fairly new technology, and the market is still young. The fight to present the best offers and to win new customers is still going on in some countries and, in some others, it has not even started yet. However, this technology is not risk free. In fact, we have already seen a couple of security issues under consideration. In the coming years, we will see how the use of this technology grows even at an end-user level. Unfortunately, we will also see the growth of malware trying to take advantage of it. There are quite a few possibilities:

- Trojans being able to record conversations and sending them to the attacker
- Adware introducing advertisements in the middle of a conversation
- Identity theft attacks, allowing an attacker presenting himself/herself as someone else (compare "vishing" --see http://en.wikipedia.org/wiki/Vishing.)

But the scariest part of VoIP technology is that when you talk about Voice over IP, everybody thinks of it as being just about that: voice. Many do not realize that VoIP systems are capable of transmitting images along with the voice. This means that we could face a Trojan capable of controlling the camera and have a complete view of the premises it is located in. With that information, it might open a new office on the other side of the world, pretending to be an "official" partner and doing business on your company's behalf, but keeping all the profit for the scammer. This kind of system could also establish an adware distributor's paradise, inserting all kinds of advertisement into a conversation: banners, moving texts, full screen adverts, videos, and so on.

RSS

Really Simple Syndication (RSS) is an eXtensible Markup Language (XML)-based system that offers a rather simple information feed. Users interested in this information can subscribe to the feeder Web site through a specially designed application (there are many applications available), and be kept up-to-date with all the information being published on that site. In fact, all the sites offering news (radio stations, newspapers, TV channels, and so on) allow everyone to subscribe to their RSS feed, and the number of users of this system is growing day by day.

This means of communication is becoming so popular, that people are thinking of converting it in new e-mail formats in order to get in touch with someone else. It could be a great advantage for the users, as they could establish a direct line with their "friends." But, once again, the most popular communication systems become the best candidates for a malware attack. RSS is based on XML, which is a very simple and versatile language. Therefore, it would be very easy for an attacker to program some maliciously intended code, once a way of entering code has been discovered. Yes, entering the users system is not easy, but techniques like social engineering and buffer overflows can be applied in all sorts of contexts.

RSS is one of the attack vectors we may well see exploited in the future.

Podcast

Wikipedia defines a podcast as "the method of distributing multimedia files, such as audio or video programs, over the Internet using syndication feeds, for playback on mobile devices and personal computers." What does this mean? Let us suppose that you want to listen to a radio program or watch a video that is going to be broadcast through the Internet. However, you cannot do so, because it will be broadcast at the precise moment when you are supposed to have dinner with the gentleman/lady who works in the building next to your office, and with whom you have been trying to go out for over three months. Podcast is a solution (not necessarily the only one, but it's one that's known to work), provided that the radio station or the company broadcasting the content offers you the podcast option. It lets you listen or watch previously broadcast programs whenever you want to. Even more, it can also be used to share your own recordings, such as something funny you recorded, or even a promotional video demo by your new band. So imagine, for example, a multimedia file (as the definition says) capable of exploiting a vulnerability in order to do some kind of malware-like action. Well, it is not very difficult to imagine, is it? In fact, such a file exists already. It appeared in January 2005, and it was capable of downloading and installing malware on the victim's system by exploiting an existing vulnerability in Windows Media Player. If that multimedia file could be "broadcast" by a popular media resource such as a radio station, its success could be headlined in news and newspapers all around the world.

Well, we are not planning to give any ideas to the bad guys that they won't have thought of long ago, but podcasting is a very popular service. As it comes from (normally) trusted resources, nobody thinks of it as a possible threat. But let us suppose someone uses social engineering to convince users that they can download a very interesting podcast from an obviously trusted and safe resource. Do you think it could work? The future may prove exactly that.

Home Media Systems

When the Lumiere brothers invented the cinematograph, life changed in ways that still impact us. There was a new way to be entertained, and such enhancements as sound and color started to change the whole concept of entertainment from active to passive. Then television came into the home, and people were amazed and delighted. But someone thought that it would be great to mix them both and play films on TV. It was indeed a great idea, and everybody loved to watch films when they were comfortably seated on their own sofa.

But there was one problem. You could only watch the film that was being broadcast. You could not choose your own film, so sometimes it was not so funny or exciting. Aha! We will invent the video, and people will be able not only to watch what they want, but also record it to have it ready whenever they want. That was even more successful, and the use of such devices became extremely popular. But people wanted more, and home media system systems were developed, capable of playing and recording TV programs, films, music, and so on. Such systems are even able to connect to the Internet and provide complete access to all kinds of materials, whenever we choose. And that is the future: like it or not, most of us will eventually have such systems in our homes.

However, such a combination (going by such names as "home entertainment system" or "home media system") needs a special machine to control all of the operations. Of course, it needs a computer. And as such, it needs an operating system to be able to control and configure all the options. Therefore, we have a computer whose primary use will be entertainment. But there is a difference. Normally, a computer is on part of the time, and switched off for a significant part of each day. (We know there are exceptions to this scenario, but let us assume a standard case.) In the case of a home media system, even when someone does not want to watch the news, someone else wants to listen to music or someone else wants to watch cartoons. So, at the end of the day, the system stays on most of the time. It happens already in many homes that the TV stays on most of the time. Sometimes nobody watches it and you have to remind everyone to turn it off to avoid wasting money, but that is a different war that parents and offspring continue to fight.

The fact is that the system stays continuously on, and stays connected to the Internet. Furthermore, as more entertainment and information comes from sources outside the home, the home media system becomes a focus for all kinds of entertainment and educational activity, from movies to communication to research. That means that someone could try to access our central system to steal our information. Furthermore, they could try to "inject" advertisements or worse while we watch TV or listen to the radio. Another possibility is that they might be able to modify our system so it transmits a certain signal, as if the user was watching a certain program. That way, the data could be altered and give "faulty" results relating to the success of certain TV programs. These are just the beginning of a whole list of possibilities for system attacks. They may seem far-fetched, but in principle they are no different to attacks like click fraud, for which bot-compromised systems are already commonly used. What are the chances that different technology can be subverted to the same effect?

Cell Phones

Are you one of those "special" persons in the world who, being able to afford a mobile phone, still choose not to have one?

At the end of October 2004, in China alone there were 320 million mobile phones, compared to 310 million of fixed lines. From January to October of the same year, the aforementioned cell phones were used to send 176.000 million SMS messages. At the end of 2004, the same country was believed to have 53 million personal computers.

Let us suppose that all the personal computers run on any of the Windows operating systems available in the market. The percentage of computer viruses existing nowadays is much higher than the viruses affecting mobile phones (we will not consider different mobile platforms or operating systems for the moment.) So, even if all the Chinese computers were running a version of Windows, they would represent less than 17 percent of the mobile phone using population. There is no doubt that the use of the mobile phone will become more popular, and still less doubt that many people already consider it a basic element for their everyday life. So, if there is a potential population so big that can be attacked via "mobile" malware, how can an attacker resist such a prospect? You may think that if they wanted to attack so many phones, they would have done it already. So, you might assume, that means they are not interested. But the truth is very different. The real situation is that this technology is evolving so fast, that they have not been able to get the desired skills and they are probably doing some research. We can be sure that the attacks will come one way or the other.

However, this field offers a new possibility. Probably for the first time, the attackers are not generally focusing on Microsoft's operating system. Odd, isn't it? Not really. The situation right now is that the number of mobile phones using Windows-based operating systems is far less the number of phones using Symbian OS. The story goes on, but we actually go back to its beginning. The point here for the virus writer is still to get as many victims as possible, so the number of users is very important. Perhaps, in due course, we'll see a similar shift in motivation from mischief, bragging rights, glory hunting and competition with security vendors, to straightforward criminal profit, but at the moment cell phone malware authoring seems well behind that curve.

Damage & Defense

Virus Writer Phone Home

F-Secure have been tracking cell phone malware for many years, going back to a time when the most virulent "mobile virus" was a hoax, and have sometimes been accused of hyping up the issue (not by us!) In fact, in an article by Mark Woods called "Mobile Viruses – Don't Believe The Hype" (www.f-secure.com/f-secure/pressroom/protected/prot-3-2006/17-459-3671.shtml), they offer quite a balanced picture of the situation.

At the time it was written (apparently March 2006), they estimated that there were 319 known viruses for mobile phones. Clearly, this pales somewhat by comparison with the hundreds of thousands of PC/Windows viruses. To put things into a slightly more alarming perspective, however, those figures represent an almost ten-fold increase over a two-year period. And compare it to the number of known viruses specific to the Macintosh: considerably less than 100, most of which have practically disappeared since Apple launched OSX.

It does seem at present that most cell phone malicious programs are "Proof of Concept," although some can cause considerable inconvenience (e.g., some versions of SymbOS/Skulls cause all applications on the affected phone to fail). In fact, malware can damage mobile phones in a number of respects:

- Personal data stored on the phone can be lost
- It may have serious impact on phone functionality, requiring reinstallation/reflashing
- These forms of damage can have serious knock-on effects for the private or business user

However, there's little evidence at present of malware clearly motivated by a search for criminal profit, or the intent to install spyware or adware. It seems very likely, however, that the increasing convergence between cell phones, smart phones, PDAs, and other handheld technology will make mobile devices increasingly attractive to a wider range of criminals, in search of some form of profit.

Credit Cards

Nowadays, can you think of anyone having a bank account and not having a credit card? I can. My own father-in-law does not have any. He could have several of them, but he keeps refusing them. I know some readers will not believe me, and think this is just a funny story to make the tone of this chapter more amusing. I am sorry, but it is true: wherever he goes, and whatever he buys, he only uses cash.

Anyway, the fact is that the use of credit and debit cards is growing day by day. And, unfortunately, so is credit card fraud. More situations are known everyday where a credit card fraud has been perpetrated. But, wait a minute. We have talked about cards and credit cards. Are they the same thing? Most people tend to use the term interchangeably, but there are several kinds of cards. We are not going to explain here all the technologies and possibilities, interesting though this would be, as it could in itself cover a whole chapter or even a whole book. However, I will mention some different kinds of cards in order to illustrate some of the future possibilities for exploiting card technology.

The fraud we mainly hear of these days (and some readers may even have suffered it themselves) is fraud targeting credit and debit cards. However, once you enter the "cycle of cards," debit and credit cards are just the tip of the iceberg. You need, or want to have a card for the supermarket, another one for the club, one more for the flights you make, and another one for the petrol station. With all these cards, we face the first problem: your wallet is not big enough to carry such amount of cards. That can be solved. The real problem we face here is that, none of those cards relates directly to your financial resources in the same clear way as a credit or debit card does. That may give us a false sense of security. However, all of them imply that at some point money will be moved from the same bank account from which your credit and debit cards draw money. The false sensation of security may well come from the idea that the "bad guys" have not focused on abusing this kind of card yet. But in the future, this is one of the fraudulent scenarios we are likely to face. For example, if I found a card from a supermarket, or some other kind of store card, I would initially think that somebody has lost it, and I would not think much about it. However, if I decide to pick it up, go to the nearest supermarket (one where the card is valid, of course), have a good bout of retail therapy and use the card to pay for it, that's rather a different thing. I can shop for free! Now consider the increasing trend towards multi-function cards and electronic wallets. A card-like device can now hold many different types of data and functionality: identity information, access to funds, and so on.

Operating Systems

Despite our focus here on Windows (and DOS, in our more nostalgic moments), we acknowledge that there are several operating systems in common use (by us, too!). However, there are many people who still think that using this or that operating system will save them from being infected or being attacked. Wrong. They may be less at risk of being infected or being attacked, but that Windows Vista security, especially as regards malicious software. In fact, the AV industry has, on the whole, been fairly positive about Vista. Andrew Lee's article in Virus Bulletin in July 2006 ("Fixing the Virus Problem?") looked at User Account Control, Consent and Credentials, Code Integrity, Application Isolation, Service Hardening, and Windows Defender, and concluded that some of these measures would have a positive impact on the virus issue (at least in the short term, but warned that it would be a retrograde step if the Vista user was to feel "like so many misguided GNU/Linux and Mac OS users, invulnerable to attack from either viruses or the plethora of other undesirable software attacks, particularly ones that employ social engineering techniques..."

Symantec (www.symantec.com/avcenter/reference/Windows_Vista_Security_Model_Analysis.pdf) also looked at Vista privilege management (User Account Protection and Privilege Isolation in the more formal "Analysis of the Windows Vista Security Model") and were reasonably positive, while predicting that other weaknesses would be brought to light (notably privilege escalation errors) and that strenuous attempts would be made to find and exploit them.

Sophos, perhaps stimulated by bullish comments by Jim Allchin of Microsoft about Vista security (www.betanews.com/article/print/Allchin_Suggests_Vista_Wont_Need_Antivirus/1163104965), published a report (www.sophos.com/pressoffice/news/articles/2006/11/toptennov.html) indicating that their tests showed that some current malware threats were capable, under some circumstances, of bypassing Vista defenses. Interestingly, Allchin's response was, while not constituting a climbdown, clearly disassociating the company from suggestions that Vista users are immune to virus infection and in no need of AV software (http://windowsvistablog.com/blogs/windowsvista/archive/2006/12/19/windows-vista-and-protection-from-malware.aspx). Not a real surprise, given that Microsoft have themselves invested heavily in their own antimalware product line.

Well, that's all very well, but there are many people who have no faith in Microsoft security at all, and point to their own favored system as the Real Deal. In fact, you might be surprised at how many AV researchers are Mac and/or Linux users, at least some of the time. Is it because they regard them as "safer?" Well, in the sense of there being (much, much) less malicious software that affects these platforms, perhaps. But I can't think of any researcher who likes the word immune (even when it's used by their own marketing folks; in fact, least of all under those circumstances.

Summary

The change from AV to anti-malware technology in recent years has been dramatic. But the changes in malware authoring culture have been even more spectacular. At the beginning of the 21st century, the ether was still buzzing with the self-obsessed, self-important braggadocio of virus writers, chattering about their own technical skill and the incompetence of the AV companies. By the second half of the first decade, viruses and even worms have become a relatively minor part of the problem, compared to the bots and Trojans that now dominate the scene. Sometimes, such malware lacks the elegance of some of the proof-of-concept viruses that still make the analysis pages of Virus Bulletin, but they are a great deal more profitable, and profit is the primary driver for most authors nowadays. Even young malware authors are often tempted by financial reward, where previously they were preoccupied with self-glorification and image.

There's less research currently along the lines of Gordon's work in the 1990s, which may be in part because bragging is not only unprofitable, but risky for those engaged in criminal activities, so there are fewer verifiable "serious" malware authors to talk to, the self-advertising of "Wicked Rose" and friends notwithstanding. It's not easy to predict the exact shape of the future threatscape, but ongoing movement away from hobby virus writing to criminalization is a safe bet. Moreover, we guarantee that new technologies will be scrutinized for potential vulnerabilities and exploits, and that there are new tricks lurking in our future that we have not thought of yet. Not a comfortable thought, but not exactly a new situation, either.

Solutions Fast Track

Old School Virus Writing

☑ While there are other sources of information about old-guard virus writers from a more technical point of view (see, for instance, Vesselin Bontchev's "The Bulgarian and Soviet Virus Factories" at www.people.frisk-software.com/~bontchev/papers/factory.html), the importance of Sarah Gordon's work on the psychology of virus writers cannot be overestimated.

☑ Gordon's "Generic Virus Writer" papers, though they deal with a (necessarily) small sample, provide a significant counterbalance to the "spotty teenage psychopath" image of the virus writer, by quoting examples of virus authors whose ethical development seemed to be within normal limits.

☑ There is more "playing with viruses" than there used to be, probably more than when Gordon pointed out the issue. It can be addressed by encouraging a commitment to meet professional standards and codes of conduct such as AVIEN's or EICAR's.

The Black Economy

- ☑ There has been a significant swing away from hobby virus writing. Instead of notoriety and bragging rights, illegal profits now provide a much more powerful incentive. Similarly, elegant proof-of-concept viruses and worms tend to have less impact now than crimeware (spyware and bots, not to mention the criminal activities enabled and amplified by botnets.)
- ☑ Spam remains a problem in its own right (in terms of massive overload of junk mail). But in a more general sense, it's also a carrier for crimeware, fraudulent messages, malware, and so forth, and therefore a major component of the black economy.
- ☑ Dialers are intended to divert your Internet connection through an intermediate or alternative (and invariably expensive) provider.
- ☑ A whole black economy is based on botnets run by organized criminal groups, and incorporating a number of differentiated roles such as mass e-mailers, template providers, cashers, and so on.

"Wicked Rose" and the NCPH Hacking Group

- ☑ NCPH has about ten members, but (as of 2006) four core members, of whom "Wicked Rose" seems to be the leader.
- ☑ NCPH hackers make much use of the GinWui rootkit and the RipGof Trojan horse in attacks against US targets.
- ☑ NCPH are a group of college students in China who have made full-time wages out of part-time hacking.
- ☑ NCPH illustrate a shift among Chinese hackers from patriotic hacking to being "samurai" (hackers for hire).
- ☑ The group has made significant use of Microsoft Word vulnerabilities, PowerPoint vulnerabilities, and social engineering messages.

Lurkers in Your Crystal Ball

- ☑ Social engineering, in its criminal sense, is a way of duping people into believing something that isn't true, but works to the scammer's advantage. It's used extensively in e-mail fraud, malware dissemination, and so on. It's also used in many other non-e-mail contexts for stealing information and so on.

- ☑ A useful definition of social engineering in a more general sense is "Psychological manipulation of an individual or set of individuals to produce a desired effect on their behavior."
- ☑ In the field of malware, we note that new twists on old exploits and attacks repeatedly recur. For instance, the recurrent use of Word documents as an exploit vehicle.
- ☑ Interconnectivity brings many benefits, but it also increases the number of entry points for malicious activity.
- ☑ There have already been concerns about the possibility of malware affecting automobile systems, and VoIP technology already presents some specific issues (e.g.,vishing).
- ☑ RSS and podcasts have largely unexplored but real potential for misuse. As home media systems become more sophisticated, they may also become more vulnerable.
- ☑ There is already a great deal of malware for cell phones. However, this field seems to attract hobbyist virus writers rather than criminals at present.
- ☑ Credit cards are not the only type of card to offer potential for exploitation: consider store loyalty cards, for instance.
- ☑ There are many more attacks on Windows platforms than on other systems such as Linux or Mac OS. However, there is a persistent danger that users of other systems will overestimate the invulnerability of their favored system. This sort of "sense of false security" can make them particularly vulnerable to social engineering attacks.

Frequently Asked Questions

Q: Is it really that easy to make money out of e-crime?

A: Well, you need contacts, an infrastructure, and some technical knowledge, or access to someone who has such knowledge. And you probably need reasonable knowledge of the law to help you avoid being pursued by it. Excuse us if we don't give you specific information on how to pursue a career in organized crime.☺

Q: Aren't there any female virus writers?

A: Very few, at least that we know of. The first "Generic Virus Writer" paper addresses that issue.

Q: Gordon writes about alt.comp.virus. Is that a useful resource?

A: David Harley used to be very much involved with that group (he co-wrote and co-maintained the FAQ Gordon mentions), but stopped visiting some years ago. Like most of the "legit" researchers who used to represent the Good Guys there, he got bored with the poor signal-to-noise ratio. There's also an alt.comp.anti-virus, which attempts to represent a more white hat view (a.c.v. included a lot of virus writers and wannabes.) The moderated comp.virus group was a much better resource, but has been dormant for many years, and at present looks likely to be quietly put to sleep.

Q: So are there no hobbyist virus writers any more?

A: There are, but probably not so many, and they tend to be less vocal (or attract less attention).

Q: Don't AV researchers ever write viruses? Doesn't that affect their competence to defend us against malware?

A: Some don't. (And that includes some of the most competent researchers.) Those that do, do so under carefully controlled circumstances. No, they don't release them into the wild and then make us buy updates that will detect them. And most AV companies still don't employ virus writers (knowingly, anyway.) This is probably more common among other sectors of the security industry.

Q: What's the difference between a bot and a Trojan?

A: Bots, even in the limited sense of the term used here, are not exactly a single class of malware (except in so far as they further the interests of a botmaster, maybe). They mostly belong to the general class of Trojan, but some viruses and worms have also been classified as bots.

Q: Surely those sub-categories can't all be spam?

A: That depends on your point of view. It's not as all-inclusive as "anything I didn't want to receive" or "what I say it is."

Q: Are these all the existing crime models?

A: Not at all. New twists come up all the time, and it wouldn't be too difficult to fill a book on cybercrime alone.

Q: So what's the difference between a dialer and a porn dialer?

A: A porn dialer is used to dial into a pornographic "service." However, the main motivation is still usually to generate income through high connection charges.

Q: So what do NCPH actually do?

A: Develop and deploy exploit code, with the ultimate aim of stealing data.

Q: What's the real significance of this case study?

A: Several issues: the continuing success of this sort of targeted attack; the disciplined and cooperative nature of the group; the importance of the financial rewards in this study.

Q: What is "reverse social engineering"?

A: That's when the perpetrator masquerades as a figure of authority or information resource so that the victim makes the initial contact. This normally increases the credibility of the perpetrator.

Q: Don't viruses actually precede the first PC viruses?

A: The first virus for a microcomputer was probably "Elk Cloner," written by Richard Skrenta on the Apple II around 1981 or 1982 (depending on who you believe.) There was already some worm research afoot by then, notably the Shoch and Hupp segmented worm experiment at Xerox. That wasn't intentionally malicious, of course.

Q: Surely there aren't any Mac viruses?

A: There are actually quite a few (plus some other types of malware) that can affect operating systems earlier than OS X, but we rarely see them now. Macro viruses can still affect Macs as well as PCs, though their exact effect (and even their ability to replicate) will vary. There is OS X malware, but it hasn't had much impact. There is no absolute reason why that can't change, though. Also, OS X is basically a version of UNIX, and UNIX isn't invulnerable either. In fact, some of the fairly generic UNIX threats could work on a Mac with little or no modification, and we haven't yet fully explored the possibilities of Intel-driven Macs.

Q: What about Linux viruses, then?

A: There are one or two, but they're basically proof-of-concept. There have been significant Linux worms and rootkits, though.

Q: So the virus thing is hype?

A: Is it? There seems to be a feeling among non-Windows zealots that if you re-define a threat as "not a virus," it ceases to have any power. Unfortunately, viruses are just a (shrinking) proportion of the malware problem (in and out of Windows.).

Q: Are these just guesses at future scenarios or are they based on something more?

A: They are guesses based on experience and research. In fact, during the writing of the book some guesses have become true.

Q: Isn't the problem our reliance on technology?

A: That might be a social problem, but it's not the malware problem. The issues there are:

- Educating malice out of the human race. Well, that might take a while.
- Educating the victims to take better care of themselves, and to be less naïve.

Chapter 6

Defense-in-depth

Solutions in this chapter:

- Enterprise Defense-in-Depth
- Malware Detection
- Planning, Testing, Revising
- Personnel
- Look Beyond the Borders
- Documentation
- Malware Laboratory Procedures

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

Ken Bechtel, a veteran of the enterprise defense wars and a strong advocate of multi-layering and defense-in-depth, kicks off this chapter with an overview of defense-in-depth strategies. Paul Schmehl then looks more specifically at defense-in-depth in the enterprise. David Harley takes a look at intrusion and virus detection, and then we hand it back to Ken for an avalanche of material on implementing various aspects of malware defense.

Mitigating the impact of malicious code upon the enterprise requires more than just anti-virus (AV) software. It requires a well-thought out plan of action that addresses various contingencies. This chapter is designed to facilitate that thought process, and to outline procedures and issues that can help ensure a reasonable level of protection in a generic corporate environment.

Every workstation or server is a potential entry point for malicious software, and must be protected. The value of functional, up-to-date AV software cannot be underestimated. Coupled where possible with intrusion detection monitors (personal firewalls, integrity management tools such as *tripwire* and even basic network security features), a PC can be turned into a mini-fortress. The non-specific (generic) type of approach will be covered later.

Many security practitioners prefer a centrally managed infrastructure with a dedicated AV console. Such a system not only provides positive control of the AV software, but also provide critical reports and statistics, resulting in meaningful metrics. These can be used to further enhance the defensive architecture. Current AV products work best against known viruses. Vendors are improving their technology to detect new, unknown viruses using advanced heuristics, but these systems are still evolving, as are the technologies against which they are designed to provide protection.

Tools & Traps

Setting Choke Points

Networks have devices through which all traffic or specific types of traffic must pass, which are called "choke points." Proxy servers, e-mail servers, and mail gateways are ideal locations to check for malware in transit, since there are many versions of network and e-mail-aware malicious code.

For example, a large corporation may have over 20,000 desktop units, but still employ only one mail gateway. Care must be taken that the product is able to process

Continued

the traffic sent through that choke point. High traffic proxy servers can be particularly troublesome, as Hypertext Transfer Protocol (HTTP) scanning is still in its infancy (see Igor Muttik's chapter on "A Tangled Web"). The problem remains that if a product is causing too much delay (technically, this is often referred to as "latency") in e-mail scanning or HTTP traffic, this will only be discouraging and anger the user community, and they will look for ways around the "problem" area.

Remember, just because choke points are being covered, that doesn't mean it's sensible to relax controls on keeping individual desktops up-to-date. While the gateways are ideal to keep some malicious traffic out, all machines on the network are potential entry points. Choke point scanning should be seen as complementary to desktop protection, not as a replacement, more so now that even low-spec desktop machines and laptops have comparable functionality to server-class machines.

As part of a good "data hygiene" regimen, regularly scheduled virus scans should be made of all systems, as a supplement to real time (on-access) scanning. On UNIX systems, a scan of user data volumes may be scheduled more frequently than the systems areas, especially if tripwire (or other integrity checking software) is used to notify of any changes to the system files.

The use of multi-vendor products is slightly in dispute. Two general schools of thought exist. One side prefers the single vendor approach, arguing that there is a single point of support, update, and contact should the product fail. Major-league AV vendors share information, and statistically, they all generally score in the plus or minus 0.0x percent false negative range on known viruses, and if a product breaks or misses a virus, you only need to make one phone call.

The multi-vendor camp holds the view that even if the vendors generally score within less than 0.01 percent of each other, do you want that 0.01 percent malware to be the one infecting your network? This is a legitimate concern. Vendors may have their own threat levels, based on their customer base. As such, different vendors may put a different priority on the same virus sample, and detect that virus before another does. It is also known that virus authors will target one or more AV packages. By running more than one package at different levels of the network, detection capabilities are increased (albeit only by small percentages), and there is a backup should one of the products be found to be compromised. Redundancy is a necessity in the modern Internet-connected world, and you may never need the second layer of protection, but if you do, you'll be more than happy to have it.

Enterprise Defense-in-Depth

Defense-in-depth is a term frequently used in discussions of security practice. Often, comparisons are made to castles and moats, as though the enemy was without and safety lay within. However, in the world of computing, the enemy is everywhere and anywhere. Strategies designed to defend an enterprise and its data fall short when they do not account for the reality that attacks can come from anywhere at any time.

For example, an often-offered strategy for defending against the Slammer worm consisted of blocking port 1444/tcp, the port Slammer used to attack vulnerable Microsoft SQL Servers. As some unprepared enterprise administrators soon discovered, laptops that were infected with Slammer elsewhere and were subsequently connected to their networks, bypassed the firewall and rendered the port-blocking strategy irrelevant.

Tools & Traps

The Moated Walls of Jericho

Dr. Frederick Cohen, arguably the founding father of AV, describes the moated wall as "one of the great technological innovations of all time...a wonderful example of synergy." His point is that defense-in-depth combines a number of defensive techniques, resulting in a total defense, which is more effective than any of its components, if not "greater than the sum of its parts." Or, as Cohen puts it, "Defense-in-depth against computer attack also leads to synergistic effects that may cause each of the defense [sic] to operate with more strength than it could operate with alone," and quotes the example of a combination of access control and an integrity shell. (Chapter 3, "A Short Course on Computer Viruses" 2nd Edition (Wiley, 1994).

More recently, the Jericho Forum has for some years criticized the reliance on an external perimeter, advocating instead a policy of "de-perimeterization," a model they describe as connecting the enterprise and its business processes to its external stakeholders "seamlessly and securely." In the last year or two, however, the forum has inclined less towards the "death of the firewall" and more towards a more specific model of "micro-perimeterization," whereby protection is implemented at multiple levels using:

- Cryptography
- Data-level authentication
- Inherently secure protocols
- Inherently secure systems

While this model doesn't use the phrase "defense-in-depth," it's clearly more tolerant of a range of security solution types. You can find out more about the Jericho Forum at www.opengroup.org/projects/jericho/index.tpl.

In today's world of ubiquitous and disparate mobile devices that create porous (and non-existent) network borders, defending the enterprise against digital compromise requires an intimate knowledge not only of the network's topology, but also of its daily uses, the types of data exposed to risk, and the daily habits of its users, including where and when they travel with mobile assets in tow. It is not enough to think in terms of protecting an edge, for there really is no edge. Every asset, every host, and its use must be considered in a defense-in-depth strategy if it is to be successful in protecting the enterprise.

Getting to Know Your Network

Knowledge of network assets should be the primary goal of every security administrator. Before you can create a defense–in–depth strategy, you must first know what it is that you are defending. In any enterprise consisting of more than a handful of assets, it's not feasible to know the condition or the use of all devices by physically monitoring them. Tools are therefore required in order to present that knowledge in a meaningful, effective way.

Tools & Traps

Passive Monitoring versus Active Probing

Two important methods of asset discovery involve passive network monitoring and active network probing.

Passive monitoring:

- Intrusion Detection Systems (IDSes)
- Behavioral Analysis Systems

Active probing:

- Vulnerability Analysis Systems
- Patch Management Systems
- Host-based Defensive System
- Access Control Systems
- Policy Enforcement Systems

Choosing Your Network-Knowledge Tools

Choosing the right tools to gain knowledge of your network is important. Some tools require more manpower than others. For many tools, high levels of skill are required in order to interpret their outputs properly. Staffing levels should be included, as well as staff competencies, in any analysis of which product types to incorporate into your intelligence-gathering toolbox. In general, passive monitoring tools require high skill levels to interpret the results properly. Is X anomaly a bad thing? Is Y new service a threat to the network? Is device Z connecting to other devices in a dangerous or unapproved way?

On the other hand, active probing tools tend to generate large amounts of data that must be interpreted carefully. Plowing through page after page of vulnerability reports containing much material that can't be trusted completely (is Apache really not patched? Or is that a false positive?) or is irrelevant to your network (do you really need to know that port 445 is open and listening?), requires many hours of work sifting through false positives; time which might be better spent on other activities.

Tools & Traps

Passive Monitoring versus Active Probing: Pros and Cons

Passive Monitoring Systems

- Pros:
  - Do not require agent installs
  - Provide real-time data
  - Report changes immediately
  - Discover unknown and unusual services
  - Are difficult to detect and discover
- Cons:
  - Require high skill levels to interpret correctly
  - Require regular monitoring
  - Do not provide any defensive capability

Active Probing Systems

- Pros:
  - Agents provide very accurate information
  - Can provide zero-day protection against attack
- Cons:
  - Each product requires a proprietary agent
  - Agents consume system resources
  - Provide static snapshots of host condition
  - Central management systems require change management

Designing An Effective Protection Strategy

Building an effective defense-in-depth strategy to protect a large network from attacks requires the designing of protection that focuses on the goals of the attackers and attempts to thwart them.

At one time, viruses were released simply to harass or irritate people, to send e-mail directly, to make a political statement, or to prove it could be done. Cleaning up after such an attack was often as simple as deleting a few files. Modern malicious software (malware) is designed to gain control of a host so that it can be used for other purposes such as sending spam, participating in Distributed Denial of Service (DDoS) attacks, acting as repositories for illegal content, and so on. Preventing these attacks from succeeding, therefore, is more critical than it was in the past, because cleanup is expensive and data exposure can literally extinguish the life of a business. Administrators must use every tool at their disposal to win the battle.

NOTE

Cleanup has always been expensive once malware has a foothold on production systems. It could be said to be more so now, since malware authors often expend considerable effort to make it as difficult as possible to remove their creations, once installed. They do, after all, often have a financial incentive.

Secure Individual Hosts First

Since every host on the network is a target, protection strategies must be devised for every host and platform, including Windows, Mac OS, UNIX and Linux, switches, routers, and even security appliances. Every host that is IP-addressable is a potential target. Private IPs are no protection against attack. Once a single internet-facing host has been compromised, the entire interior network is exposed to risk.

Since AV protection doesn't even exist for some devices and platforms, the primary strategy must center on configuration and patch management, access control and restriction, and consistent auditing to confirm that policies are being consistently and regularly applied and verified.

Access control and restriction is an important part of any defense system. Unrestricted access to all devices on a network should never be the norm. Instead, hosts should be restricted to accessing only those other hosts that they require access to in order to perform their function.

For example, access to switch and router management interfaces should be restricted to as few networks and hosts as possible. Databases should only allow connections from trusted hosts. Even general-purpose workstations should not be able to probe every node on a network without tripping alarms and investigations.

The single most important protective measure for any host is simply not to run any services that are not required for that host to function. Workstations should never be used as servers, and should only run those services required to access and share files (Common Internet File System) CIFS, (Network File System) NFS, and so on and perform the services they are intended to supply.

Servers should never be used as workstations, and should only run those services required to fulfill their designated function.

Disable and, if possible, remove all unnecessary services. Name servers, for example, have often been the victims of spammers, who would relay their spam through an old, unpatched and unconfigured version of sendmail, because no one thought about the fact that the server came with sendmail enabled. Audits of hosts should include the identification of all enabled processes (not just running), and a review of their purpose to see if they are really needed.

Even needed services should be carefully reviewed to ensure that they do not offer unneeded functionalities or capabilities that the administrators are either unaware of or unfamiliar with. Web servers, for example, often come with sample scripts or online documentation that may represent a risk of compromise. Restrict access to them to those who need to access them, and disallow access completely to those with no need of it (either by setting correct permissions or removing them from the server.)

Review all necessary services to ensure that default passwords are not used, and that the configuration files are edited to ensure that the process behaves in an expected and approved manner. Testing should be conducted to confirm that the process does behave as expected. Often, something as simple as loading a Web page may reveal weaknesses that were not thought of, or an unanticipated means of access.

Purchase Host-based Protective Software

For those host platforms for which AV software exists, most vendors offer management software that can install, update, and monitor the condition of the software on hosts. Administrators should never assume that certain operating systems are not vulnerable to infection and therefore do not need to have AV protection installed. It's entirely possible for a host that is not vulnerable to a particular attack to make malicious files available to other hosts that are vulnerable. The non-vulnerable host then acts as the vector by which an infection is introduced to the network. (This is often referred to as "heterogeneous virus transmission".)

WARNING

Ken Bechtel puts this well: "All platforms must be protected. Malware has been written to target all operating systems, and some malware is platform independent. Javascript, macros, and some worms will just as easily operate on more than one OS platform. Servers often take on the role of `Typhoid Mary' where an infected document or PC application may be archived and lay dormant for weeks, months or even years, even if the server isn't itself vulnerable, simply by storing the infected file."

Consideration should also be given to other types of host-based protective software. Anti-spyware software, host-based intrusion detection and protection, and even access protection and buffer overflow blocking capabilities have recently been appearing in some AV software versions, and can be used to further protect a host from attack. Configuration control products should be evaluated for their ability to maintain known, secure configurations on controlled hosts. Host-based firewalls can also be used to limit a host's exposure to attack and ensure that, even if unanticipated services are enabled, they cannot be used to compromise a machine.

Carefully Examine All Points of Access to Hosts

Once individual host protections are in place, it's time to look carefully at ingress and egress points to all hosts. Here, a number of possibilities present themselves. Switch and router access control lists (ACLs) can be used to limit access to critical assets. Edge-based devices can be used to further limit access and act as choke points, forcing attacks to go through a gauntlet of checks before being delivered to an endpoint. Mail servers and Web proxies are particularly useful, allowing administrators to reject unsafe attachments, scan much of what travels across the network for viruses, and reject content based upon readily available public blacklists and other forms of reputation service. Firewalls and intrusion prevention systems (IPSes) can be used to limit access to only those services and assets that administrators determine are needed to maintain required Internet services.

Administrators should not assume that a particular host is protected. Tools such as port and vulnerability scanners should be used to determine what can be accessed on a host, and how the host presents itself to an attacker. If ports are found to be open that were not expected to be open, they should be investigated. Their purpose and necessity should be determined and compliance with policy should be confirmed. Ongoing scans should be performed to ensure that hosts remain compliant after being approved.

Malware Detection

One of the least appreciated prevention methods is early detection. If you know the nature of the threat, you can check for indicators that the threat is present. Analysts the world over understand this, but very seldom is it practiced in a corporate data environment. Often, a lack of knowledge or coordination prevents full implementation of network monitoring.

In the corporate world, computer security often overlaps other areas of responsibilities. Too often these responsibilities turn into political "territories." Unfortunately, there is no quick fix for office politics, but people do need to understand the capabilities (and necessity) of information sharing for, and with, all the other sections and teams in the company. The current trend towards the downsizing of corporate computer security departments, causing the security analysts to wear multiple hats and adversely affecting capabilities, adds to the importance of monitoring. Where other factors impact job performance, management must lend their support accordingly, with full knowledge of the situation and purpose behind the requests.

Just as anti-malware strategy in a corporate environment is often little more than unconventional warfare against rogue code, anti-malware tools are also unconventional. Rather than using only off-the-shelf software you can supplement it with (often customized) open source software. Many of the best tools for monitoring for malicious software are not commercially available from AV vendors. Some favorite tools are freely available from sources on the Internet, and we discuss some of these tools here and in other chapters in this book.

Monitoring infection logs is a must. The earlier mentioned AV consoles are great for monitoring the state of AV software and malware detection. By monitoring the detection logs (from all platforms), you can get a jump on any infection that makes it past your defenses. For example, an early detection based on an e-mail can prevent e-mail infections from spreading by disabling the user account.

Intrusion Detection

IDSes are generally either host-based or network based. Both types are further subdivided into signature detection and anomaly detection. A host-based IDS (HIDS) looks at a single system for threat signatures or for suspicious behavior. A network-based IDS (NIDS) is used for monitoring a network, seeing protected hosts in terms of the external interfaces to the rest of the network. Most of its results are achieved through network packet analysis.
A NIDS is particularly useful for detecting:

- Upcoming Denial of Service (DoS) attacks
- Scans for open or listening ports.
- Port sweeps, scanning a range of hosts for a listening port
- Specific probe/attack signatures

Tools & Traps

Intrusion Detection Signatures

Signature detection is based on known characteristics of a threat. For example, the following is a string associated with the Code Red worm. Looking for an extracted substring in Web traffic is a simple way of checking for a Code Red probe.

```
/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNN%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u6858%uc
bd3%u7801%u9090%u9090%u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a
HTTP/1.0
```

Note that this is not exactly what we mean by a signature. This is an "object" associated with the presence of the worm, which can be used as the basis for a signature (i.e., the pattern or algorithm used by the IDS).

Ingress filtering has obvious uses, but monitoring of local and outgoing traffic (egress filtering) is a major source of data on malware present within your own perimeter.

A HIDS focuses on individual systems. In general, you administer an enterprise system centrally, though there may be agent software on the local host. The HIDS monitors configurational information and suspicious activity on a protected system, and detects attacks that have circumvented outward-facing defensive systems (having been introduced from an internal network or removal media, direct tampering, and so on).

In anomaly detection, we "train" the software by developing a baseline view of what constitutes "normal" activity for the environment. You need to do this over an extended period, in order to get a picture of how activity patterns change over extended periods, according to the way in which processes and traffic fluctuate. You can also accommodate mid-to-long term changes in predominant malware types and delivery methods. Once your baseline has been established, aberrant behavior such as unusually heavy traffic is a major indicator of a problem, security-related or otherwise.

A generic or anomaly detection system can sometimes detect a new attack proactively, or at least as soon as it strikes. However, an anomaly is not necessarily malicious, and the operator is responsible for determining what is really happening.

Systems that are based on recognizing known attack signatures are less prone to seeing an attack where none exists, but are more prone to "false negatives." In other words, if an attack signature isn't in the signature database, the attack won't be recognized as such. For this reason, most Anti-virus Information Exchange Network (AVIEN) members implement mixed environments, where attack-specific (signature) detection is supplemented with anomaly detection.

Products in this area range from heavy-duty, expensive network appliances and full-scale commercial intrusion management software, to open source packages like SNORT, which I looked at in some detail in "Bots – The Killer Web App."

It's not really a binary decision when it comes to intrusion management. Intrusion detection and intrusion protection aren't alternatives, they're layers of a complete strategy. Other measures that contribute towards the prevention of intrusion include (but are by no means fully inclusive):

- Sound patch management
- User education and policy enforcement
- E-mail and Web content filtering
- Generic filtering by file type
- Malware-specific detection
- Port filtering
- Traffic analysis

SNORT

SNORT is an open source (www.snort.org) lightweight NIDS, whose current feature set is comparable to commercial IDSes, and is available for most of the common platforms, including Windows, UNIX, and UNIX-like systems including Linux and Mac OS X. Sourcefire's commercial version (the Sourcefire Intrusion Sensor) is based on the SNORT detection engine with improved reporting, policy management, interface, and support (see www.sourcefire.com).

SNORT is often thought of in its conventional role of looking for hacking type intrusions by packet logging or sniffing (comparable to tcpdump, for instance), by capturing and displaying packets or header information. However, its capabilities for protocol analysis and content
filtering can be extended far beyond simple logging and display, to pick up such potential breaches of security as port scans, probes, and overflow attacks. It's surprisingly simple to write rules to look for traffic related to malware, and Martin Overton has done sterling work in this area. Check out his papers and articles at http://momusings.co.uk/publications.aspx. There is also a whole community of people developing and sharing SNORT signatures at www.bleedingsnort.com.

Here's an example of a signature created by Martin Overton for W32/Netsky.P and used here as an example, with his kind permission.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"W32.NetSky.p@mm - MIME";
content: "X7soIUEAR4s3r1f/E5UzwK51/f4PdO/+D3UGR/83r+sJ/g8PhKLw/v9XVf9T"; classtype:
misc-activity;)
```

Here's a brief guide as to how this one works.

- *alert tcp* instructs the software to send an alert when the signature later in the rule is seen in a TCP packet.
- *$EXTERNAL_NET any* means that the rule should trigger on any TCP port. The "any" keyword could be replaced by a specific port such as 110, the TCP port used by a Post Office Protocol (POP) mail client. However, using the variable *$EXTERNAL_NET* specifies that the rule should trigger only if the offending packet comes from an external IP address.
- -> *$HOME_NET any* tells us that the target IP should be on the local network on any port. The *$HOME_NET* variable is set by the administrator and refers to a given IP range.
- *(msg:"W32.NetSky.p@mm - MIME";* specifies the message text displayed.
- *content:"X7soIUEAR4s3r1f/E5UzwK51/f4PdO/+D3UGR/83r+sJ/g8PhKLw/ v9XVf9T"* defines the detection signature.
- *; classtype: misc-activity;)* means that the event is to be logged as *misc-activity*.

The following signature is also one of Martin's, slightly modified by David Harley. This rule as it stands is capable of blocking a wide range of e-mail-borne malware, but we'll also include a number of other filename extensions you might consider blocking below. (Of course, you can do this with all sorts of tools. There is a widely used exim (www.exim.org/) script that does much the same thing, but I couldn't resist the idea of implementing generic detection in a signature IDS tool.)

```
alert tcp $EXTERNAL_NET any -> any any (msg:"Bad Extensions
Match/PCRE";pcre:"/attachment\;\W{1,}filename=["]\S{1,}[.](scr|com|exe|cpl|pif|hta|
vbs|bat|lnk|hlp)/";classtype:misc-activity; rev:1;)
```

The *"pcre"* directive indicates that Perl Compatible Regular Expressions are used. More information on SNORT in general can be found at www.snort.org, where open-source SNORT can be downloaded. Snort_inline is modified to use additional rule types (drop, sdrop, reject).

TIP

There is a little more information on rule writing, using rules by Martin and by Joe Stewart as examples, in my chapter in "Botnets: the Killer Web App." For more information on SNORT rules, see www.snort.org/docs/writing_rules/ See also 'SNORT 2.1 Intrusion Detection,' also published Syngress.

Support groups provide good intelligence. Tools like SNORT are good, but they're only tools, and only as useful as the ability of the user to interpret the data. To that end, it is very important to know what to look for. Patterns are easily found, but not as easily understood.

Subscribing to early warning systems and support groups like AVIEN, SecurityFocus mailing lists, and so on are critical. The discussions between other individuals who are addressing the same problems as you are, can often give a better picture than you could achieve alone, even in the largest corporation or government service department.

The USENET newsgroups alt.comp.virus, alt.comp.anti-virus, and so on may also provide some intelligence use. Early warning of what some of the virus authors are thinking and discussing can lead to detection of suspicious activity on your own network. Vendors also put out free e-mail newsletters, and you can often subscribe to a virus alert service.

Table 6.1 Partial List of Executable, Potentially Dangerous File Extensions

File Extension	Description
ADE	Microsoft Access Project Extension
ADP	Microsoft Access Project
.{	CSLID Codes
ASD	MS Advanced Streaming Format Description File
ASF	MS Active Streaming (Media) File
ASX	MS Windows Media Active Stream Redirector File
BAS	Visual Basic® Module
BAT	Batch File
CHM	Compiled HTML Help File
CMD	Windows NT® Command Script
COM	MS-DOS® Application
CPL	Control Panel Extension
CRT	Security Certificate
DLL	Dynamic Link Library (DLL)

Table 6.1 Continued

File Extension	Description
EXE	Application
HLP	Windows® Help File
HTA	HTML Application
HTO	Hierarchical Tagged Objects
INF	Setup Information File
INS	Internet Communication Settings
ISP	Internet Communication Settings
JS*	JScript® File
JSE	JScript Encoded Script File
LNK	Shortcut
MDB	Microsoft Access Application
MDE	Microsoft Access MDE Database
MSC	Microsoft Common Console Document
MSI	Windows Installer Package
MSP	Windows Installer Patch
MST	Visual Test Source File
OCX	Object Linking and Embedding (OLE) Control Extension
PCD	Photo CD Image
PIF	Shortcut to MS-DOS Program
REG	Registration Entries
SCT	Windows Script Component
SCR	Screen Saver
SH	Archive File
SHB	Document Shortcut File
SHS	Shell Scrap Object
SWF	Shockwave File
URL	Internet Locator
VB	VBScript File
VBE	VBScript Encoded Script File
VBS	VBScript Script File
VCS	MS Outlook Calendar File

Continued

Table 6.1 Continued

File Extension	Description
WMS	Windows Messaging File
WMD, WMZ	Windows Media File
WSC	Windows Script Component
WSF	Windows Script File
WSH	Windows Script Host Settings File

Virus Detection

Detection is only part of the malware management process. In fact, there is a whole class of measures (usually referred to as generic – we'll get on to those shortly) that generally operate at one level above malware-specific detection. In general, they detect classes of potential threats rather than instances of a specific, individual threat. Conversely, what we generally mean by virus detection is usually more than a detection control. The term is really being used as shorthand for the application of detective controls in combination with other controls, primarily recovery and preventative (see sidebar.)

Tools & Traps

Threat Management Controls

This is a simplified variation on a common model:

- **Administrative Controls** Procedures, standards, and policies
- **Corrective Controls** Measures that reduce the likelihood of recurrence
- **Detective Controls** Identify and react to exposures and breaches
- **Preventative Controls** Physical, technical, and administrative measures to mitigate exposure to malicious action.
- **Recovery Controls** Restorative measures

The AV industry may not have invented defense-in-depth/multi-layering, but it certainly seized on the model (Fred Cohen: A Short Course on Computer Viruses, published by Wiley). The popular belief that AV software only detects viruses it already knows about, has been around more or less from the start. In fact, AV systems supplement so-called signature scanning with a variety of more generic approaches, of which the best known is heuristic analysis. Furthermore, most modern commercial AV products are capable of detecting a wide range of malware, not just viruses, and may be combined with other security technologies such as the detection of spam and phishing messages.

Heuristic analysis is a term applied to a rule-based approach to diagnosis of an offending message or object. The analyzer engine works through the rule base, checking the message against criteria that indicate possible malware, assigning points where there's a match. If the score meets or exceeds a threshold score, the message is flagged as infected, infective, spam and so on, or likely to be.

The opposite of heuristic analysis in AV is not signature scanning, but algorithmic scanning, of which signature scanning is a special case. Algorithmic scanning is based on mathematically provable procedures. What is referred to in the industry as algorithmic scanning is normally understood to be based on an algorithm (other than simple string searching), which is specific to the virus it is intended to detect.

TIP

Heuristic techniques are considered in much more detail in "Heuristic Analysis: Detecting Unknown Viruses" by David Harley and Andrew Lee, at http://www.eset.com/download/whitepapers/HeurAnalysis(Mar2007)Online.pdf. For an exhaustive consideration, see Peter Szor's book "The Art of Computer Virus Research and Defense" (Symantec Press/Addison Wesley)

Generic Anti-virus

Heuristic analysis falls into the class of generic AV programs, as opposed to virus-specific detection mechanisms. Generic solutions use heuristic rule-sets as part of the diagnostic process.

For instance:

- Mail gateway filters use rules to specify what file types and file names are permitted attachments (as we've already considered with reference to a SNORT rule).
- Change detectors use the rule, "If an object's characteristics have changed, it should be treated as suspicious." Since there are many contexts in which a binary can change its checksum legitimately (self-modifying code, recompiled code, reconfiguration, run-time compression), raw change detection can exhibit a high false positive rate.
- Heuristics can enhance behavior monitoring and blocking performance and actually reduce false positives. Classic AV behavior monitoring tends to check for two types of behavior: replicative, and potentially damaging.

- Replicative code is comparatively easy to identify programmatically, especially where the code isn't significantly obfuscated.
- Some forms of damage, such as file deletion, are easier to detect programmatically than others. Detection by payload has an advantage when it comes to detecting non-replicative malware (Trojans and other non-viral programs). This approach is less effective where there is no payload, or the payload is not obviously damaging.

New malicious software is being written every day, and AV software does its best work against known malicious software. While the vendors are working to improve detection of variants of existing hostile applications, a level of uncertainty still exists. There are some basic generic security actions that can help lower your risk profile:

- **Remove 'Live' Code from E-mail, and E-mail Attachments** This approach has been very successful, and has contributed significantly to the decline of mailer and mass-mailer viruses. The drawback is it does add some constraints to user-friendly e-mail, but the benefits are quite surprising. Many believe that there is no legitimate reason to send live code. Many vendors used to send out patches and hotfixes this way, but from a security point of view it is not very efficient, and open to serious exploitation. If a file attachment was intercepted it could be infected or Trojanized, before forwarding to the recipient. This was aggravated by some mailers, which include an auto launch feature, which would run any attachment in an e-mail. Some companies learned their lesson after "Melissa," and many more learned the hard way after "Loveletter" traveled the world in a few hours. By blocking live code, "cute" games and other non-business software are also being kept out, thus reclaiming bandwidth and server storage space, not to mention user productivity.
- **Content Management** The capability to perform content management is included with many gateway-scanning products (either in the base install or as an add-in). You can also do some basic content management via sendmail scripts or Exchange server settings.
- **Turn Off Unnecessary File Shares and Apply Other Network Security Approaches** Most users do not need to share their drives, directories, or files. Even if they do, the simple step of password protecting and limiting access to a file share will often slow, if not stop, those viruses that spread via network sharing. Some users are unaware that their drive is shared out. Less technical users don't have a clue about the administrative share on each NT, 2000, and XP machine: however, virus authors are usually well aware of what *C$* stands for. Eliminating, renaming, or limiting these shares to domain administrators, closes a huge security hole.

- **Apply File level Permissions on All Public Information** While it may be that all users need to run a given executable file, it is certain that very few need to modify it. Most often they will only need to modify data, or even an INI file. In cases where a directory can't be made "read only," take the extra time to assign permissions to each file. Any binary or software package that becomes infected due to poor file level security, can have a disastrous effect with a company-wide software distribution system, like SMS or a login server. In addition, proper user and file permissions can prevent corruption of executables. Improper permissions would cause the AV scanner to prevent the virus from being appended to an executable, but allows the user to write back the modified executable, corrupting the files in the process. The time spent applying basic security policies is repaid three-fold in the time that is saved by not needing to go through restoration procedures. (It takes roughly three times the man-hours to restore systems from backups and virus clean up, as opposed to taking time to apply the security in the first place).
- **Keep Security Patches Current** The majority of malware exploits utilize known security holes. This is also true of successful hacking exploits. The sad part is that patches are readily available for most of these vulnerabilities. Customers are often critical of vendors if they don't issue hotfixes or patches to repair security holes in what is felt to be a timely fashion. Oftentimes, these same customers don't deploy the patch in a timely manner. Even when deployed, there may be some rogue users or departments who don't monitor the security risks or schedule the proper time to manage their resources. This is also another argument for creating well-defined and enforceable security practices and policies.
- **Turn Off Unnecessary Services** Not every machine needs to be running a Web server, and those machines that are not, should not be running the IIS service. Likewise, many companies do not use the Windows Scripting service, so why have WSH running as a service? Simplicity is key: the less software, the less services running, the fewer vulnerabilities are present to be exploited.
- **Restrict Mail List and CC Size** Limit the number of that addresses can be inserted into the 'TO:' or 'CC:' fields in a message. Likewise, limit the number of people who can post to enterprise-wide mailing lists. This not only helps the spread of older e-mail virus types, but also prevents accidental e-mailings to everyone in the company, and the resulting confusion and e-mail issues.

Planning, Testing, Revising

There is a saying in the military, "No plan survives first contact." An AV protection infrastructure is more than just software, and antimalware planning is a continuous, looping task. Start out simple and identify existing vulnerabilities and close them. Next, plan what to do if and when the first plan fails. As the heading implies, we have a cycle to work on, and it should be a never-ending cycle.

Planning should include input from all operational areas that are responsible for vulnerable infrastructure. Many organizations prefer to make a standing team or committee responsible for malware policy. This group should be led by someone with malware management as one of their core competencies, usually from the computer security group, if there is one. This work group should, at a minimum, have representatives from the following disciplines: Computer Security, Network Infrastructure, E-mail Administration, Server Administration, and Desktop Administration.

In smaller organizations, these five areas may be the same person. In this case, it is necessary to involve management and consult with external specialists for balance. Be careful when going to outside resources. Most commonly, people will look to their AV vendor or local security consultant for assistance in creating these plans. AV vendors may, understandably, be biased toward their own product. Additionally, security consultants and marketing agents may not have adequate knowledge or experience in AV and malware issues. Ask for proof of professional affiliations (specifically related to malware), experience dealing with malware, and endorsements/recognition from Anti-Virus industry representatives or organizations. Organizations like AVIEN, AVAR, EICAR, and Team Anti-Virus, will be able to verify the person as either a member or as a known practitioner.

Develop Contingency Plans

Know who is the primary contact/initiator and backup for each action. Activities that can be planned for, in addition to the worst case virus outbreak, should include, but not be limited to:

- Actions to take on detection of a virus that is not recognized by the primary AV software
- How to process suspected infected files
- How to respond to alerts from vendors or other external sources
- How to respond to user inquiries concerning virus alerts or hoaxes
- How to react to detection of a virus on a machine
- How to react to detection of infection or suspect infections at the gateway level
- Actions to be performed pending releases of virus detection by AV vendors

Perform an "After Action Review"

At the conclusion of each test of the plan (or the actual incident that activated the plan), assemble the team, look over the actions taken, and encourage open discussion (but not blame throwing and scapegoating.) During this review, discuss what occurred and what could have been done better or differently; consider making changes to the plan. Don't be ashamed to admit a failure, "Yeah, our primary slept through his pager" or "Because the e-mail server was overwhelmed we couldn't get the sample to the vendor." Being honest is the only way to find the flaws and fix them.

On the other hand, don't point fingers saying, "This wouldn't have happened if...." Taking this approach puts everyone on the defensive. Any ideas that may address the situation will not be voiced, or be overshadowed by the emotional defensiveness. There may be times that the way things could have been done better cannot be seen. Remember, perfection cannot be obtained, only striven for. The goal here is the minimization of impact. Only revise the plan when it improves the plan by improving a process or covering a newly discovered weakness.

Designate a Conference Room or Office as a "War Room"

This room should function as a communications center, think-tank, and meeting room. Be sure it can be dedicated to exclusive use by the outbreak management team for the duration of the outbreak. Equipment should include, but not be limited to:

- Whiteboard, chalk board, or butcher block tablet, or other device to record ideas and group discussions.
- A telephone (preferably a speaker phone) that allows conference calls, to include group members that are not able to attend physically. Try to avoid VoIP phones, as bandwidth maybe impacted during an outbreak, making this type of phone unreliable or unavailable.

It is essential to be able to secure the room. For security reasons, do not permit the casual observance of what is being brainstormed, as it may be taken out of context. Nor should you disclose vulnerabilities to third parties who do not have a "need to know." Be sure to include out-of-band communications, as traditional communications may be impacted by the current outbreak.

Some organizations that have the resources take the approach to set up an "Emergency Operations Center (EOC)," where individual representatives from the Outbreak Management team each have a "station" equipped with a dedicated phone and computer workstation. These stations are all in the same room as the conference table and whiteboard, and often a large-scale network map. The entire purpose of the war room is to act as a focal point during a situation, facilitating improved communications and cooperation between all work groups dealing with the current outbreak, and giving management a central point with which to exchange information related to the situation.

Personnel

Who should be involved with the anti-malware program in the organization? In the past, companies gave the task of AV support and maintenance to someone who had been infected many times and knew how to install, run, and update the scanning software. More companies are realizing that malware defense is proactive as well as reactive. Traditionally, this job may fall into the realm of the corporate Computer Security Unit. Unfortunately, many still feel that having some knowledge or history in traditional information security is the same as having the same amount of background with computer viruses. This is not true.

- **Designate a Corporate Anti-Malware Specialist Whenever Possible** The primary person responsible for anti-malware architecture should specialize in the field of AV, and work closely with other computer security professionals. The anti-malware professional should have a working knowledge of all levels and layers of the network, in order to understand how a malware will react on that network. The individual should have enough knowledge to be able to administer any system on the network, but doesn't have to be an expert on every platform. This individual needs access to a full complementary team. The anti-malware professional would serve as a subject matter expert, along with other expert representatives from every operating system platform, mail platform, and network infrastructure (routers, bridges, hubs, and so on) to be found in the company. This group needs to pass concerns, needs, and information to management for support. The antimalware specialist may not need to consult the group on all issues of day-to-day operations and threats, but will need to rely on them when testing products, or working on the policy planning and reaction to emergencies. If not already a member, this individual should be expected to join anti-malware-related peer groups such as AVIEN, and should be bound to a code of ethics similar to those established by AVIEN, AVAR, and the WildList.
- **The Corporate Anti-Malware Specialist Must Be Available to Everyone** One of the first things this individual must do is to establish respect in and for their area of expertise. This is not an easy task, as malware crosses many political boundaries in the enterprise. It is very difficult to get a mail admin who has been with the company for 10 years to listen to the "new guy." This can be overcome by timely and accurate communications from the specialist, as well as one-on-one time with the various work groups.
- **Management Must Give Full Support and Backing to the Specialist** You hired them to do a job, so help them to do it. There may be resistance by various work groups and "empire builders," especially in the early creations of an anti-malware program. This is where management needs to step in and address the

issue. The company's data are at risk. Without management support, the program will quickly become diluted and fragmented. Without centralized management and support, this fragmented system will eventually fail.

Look Beyond the Borders

Now that we've defined what needs to be done in the corporate environment, it's time to expand on how to protect that environment. At this point, every desktop, server, and gateway should be protected. The logs are being reviewed and used to identify holes in the defenses. The contingency and continuity plans are being practiced, and addressing the threats as they become apparent. Even so, it maybe found that the network is still under constant, if low-level, attack. What more can be done?

- **Cover Home/Remote Users** While making sure the primary in-house defenses are up to par, any corporate protective scheme will be incomplete if users are allowed to remotely connect without being protected. For employees who work remotely, make sure to have a Home Use clause with your AV provider that includes these users in the licensing agreement. This may add cost to the contract, and some users may not like having corporate dictate what they will have on their home machines. However, remember that all potential malware entry points must be covered, and that home/remote users fall into this category. Most home users are more than happy to have "free" AV software provided by the company.
- **Business Partners Cannot be Legislated For** The security practices of business partners cannot be taken for granted. It is not practical to assume that other companies or individuals follow the same standards as your business. To this end, business partners, associates, and contacts must reassure each other they have prudent protection and plans in place. If it is found that a partner is infected, offer assistance to contain the damage. After all, by helping them secure themselves, you lower the risk to your own company. If it is found that a business partner is becoming a "repeat offender," restricting access to the corporate network may be the only recourse to protect corporate resources. After a short time of inconvenience for both parties, the logic of having adequate protection should be obvious, and when the deficiency is addressed, normal business relations resume. The last resort is always a hard one for any business, especially if it's with a customer or key partner, but it must be realized that by not doing so it could cost more than what is being gained.

Documentation

Conventional wisdom holds that publishing all documentation centrally and online for the access of all is the best course of action. This can be a fatal error if taken too literally. If the policies and procedures are only available online, and that site becomes unavailable due to virus outbreak or some other DoS situation, then the documentation may as well not exist.

It is necessary for all documentation to be centrally maintained and accessible by all. A solid practice involves maintaining the documentation in three places. Two are physical notebooks, one for original documents the other to be a working copy where pen and ink changes can be made before being made official in the master notebook. The third place to keep a copy is online, for maximum accessibility during routine operations.

Careful consideration should be taken when deciding what goes into the documentation, organization is very important. A good model is organized as follows:

- 1-0 Overview
  - 1-1 Graphical High Level View
  - 1-2 AV Program visual design (line and block chart)
- 2-0 Network
  - 2-1 Network Map
  - 2-2 Network Naming Convention
  - 2-3 e-mail Servers (names, type, physical location)
- 3-0 Policies and Procedures
  - 3-1 General Corporate Policies
  - 3-2 Response Management
  - 3-2-1 Threat Analysis
  - 3-2-2 Outbreak management
  - 3-3 Lab and Support Procedures
- 4-0 Product and Technical Notes
  - 4-1 Recommended product configurations, all platforms
  - 4-2 Technical notes
  - 4-3 Internal Service Level Agreements (SLAs) and departmental agreements (use this to define duties and responsibilities to defuse political arguments)
- 5-0 Vendor related
  - 5-1 Executive Summary (use spreadsheets and quick reference for the rest of this section).
  - 5-2 Purchase orders
  - 5-3 Contracts, End User License Agreements (EULAs), and other legal agreements (like Non-disclosure Agreements [NDAs])
  - 5-4 Licenses
  - 5-5 SLAs
  - 5-6 Vendor Contacts List

For the online version, the anti-malware policies and procedures should be located with the other computer security documentation.

Copies of the policies and procedures section of the notebook (as previously outlined) should be supplied to the key players of the Malware Response Team. One of the popular methods is to use Adobe PDF format, as it is portable, easily maintained, more secure than MS-Word Documents, and has cross-platform support. By providing the documentation in this format, it allows the individuals either to print out a hard copy, or use it on a computer platform. Many key players like to have it on hand-held computers where it is readily available, while not taking up storage space.

Malware Laboratory Procedures

Not every organization needs or has the resources and expertise to staff an in-house malware lab. However, such a unit can very useful for determining how malware will behave in your specific environment, and for researching and creating non-standard ways to slow or stop its spread until vendors release official detection and countermeasures. A lab is a very serious undertaking, and could be a liability if not properly administered. Consult your corporate legal representative before implementation.

Policies and procedures for the operations of a Malicious Software and Security Lab need to have buy-in and support from the highest levels of management. This is the only way to prevent circumvention for convenience or for political gains.

Rule one for the operations of the lab must be security. The existence of the lab makes an organization vulnerable to liability issues. To offset these issues, the following policies must be practiced to show a best practice, due diligence effort.

- The lab is a secure environment. Since there are safe methods of testing AV software (such as the EICAR test file), samples are not to be given to members of the staff. The only time a sample leaves the Lab is in accordance with secure procedures (later defined), and dispatched to a defined list of recipients for virus research (such as the AV vendor's labs, or individual researchers with whom the organization works).
    - To test reliably, always start with a clean machine and network. Use Ghost or some other partition/disk cloning software and burn clean builds of each machine to CD or DVD media. Have several builds for each machine, so as to meet the requirements of any given procedure. Another option is a clean build on removable hard drives, which are re-imaged after every use. (See Chapter 9 for much more information on analysis and procedures.)
- Use a closed network, preferably in a secure area, so that it cannot be accessed from any other network, and then only use it for Lab-related functions. Don't let anyone use it for any other purpose. Utilize a non-standard gateway machine

(hosting FreeBSD or a Linux variant) if you need to move suspect files to the secure network before testing. Ensure that you physically disconnect the gateway machine and isolate the test network from the clean network after the transfer, and before beginning to test. If you must share a lab due to financial constraints, other uses of the lab must be suspended during testing. Staff without appropriate skills and expertise must be prevented from any access that might compromise the process, directly or indirectly. Alternatively, the security and malware testing may be isolated from other activities by physical means. The machines used in the testing must be physically marked to easily identify which are "hot." Under no circumstances should a network undergoing "hot" testing (testing and analysis of malware or security vulnerabilities) be mingled with "cold"/clean machines (clients or servers), or with the production network.

- Have dedicated and clearly-marked "hot" media. If you are on a closed network, there needs to be a way to get samples on (and off) of the network. Red colored floppies or distinctively marked CDs are often preferred. You must also impose a strict ban of their use anywhere else in the organization. Store samples securely in a locked drawer or safe. If you're going to use something like CD-RW (rewriteable CDs) then mark media clearly, store separately, and only use them for that purpose.
- Use PGP or other suitable, secure encryption software for transferring samples. Occasionally, samples need to be submitted to AV vendors for analysis, and sometimes so that they can supply countermeasures. You must do this in the most secure method possible. Have a machine on the dirty network with suitable encryption software installed, so that no sample leaves the network unencrypted. You will need to liaise with your AV vendor or other contacts to determine the preferred method of encryption when transferring samples. All samples should be encrypted and then transferred to removable media, so only the encrypted file is removed. This is best done on the gateway machine (see below), so that the encryption software remains insulated from contamination.
- Burn off dedicated known clean copies of common software (e.g., various Microsoft Office versions, operating systems, different versions of Internet Explorer [IE], an archiver, IRC clients, SMTP server and client software, Web servers, security software and so on), and then store the CDs in the lab area so you have it handy. Everything should be installed from CD when doing builds/rebuilds, so that there is no potential for infection. Ideally, the lab software library should normally track and reflect that used by the organization as a whole.
- Switch off, disconnect from the network, and re-clone each machine after every session. Always start clean, to minimize bias and to only replicate what is intended.

- Maintain a gateway machine. This machine should run a securable operating system. FreeBSD and Linux machines are very popular for this task, even in Windows-dominated environments, as it's easy to ensure that only necessary services are running, and potential cross-platform infections are more easily prevented. The gateway machine should have only one Network Interface Card (NIC), and should be physically switched between the two networks. This works best with visual identification to show to which network the machine is currently attached, whether by switch box, with labels, or color-coded network wires. Do not use the same user accounts on the gateway box to access both networks: have two distinct IDs. All operations on the gateway machine must be from the gateway machine. There should be no automated processes that push to or pull from the gateway machine.
- Use lab-specific IDs for testing. The login IDs on the lab machines must have no rights outside the lab, including on the gateway machine.
- Mirror the production environment, but don't limit it to known configurations. While it is best to mirror the production environment, keep in mind there are always exceptions, and it's not uncommon to have to custom build a machine to emulate a reported issue, so have a few "blanks" (i.e., unconfigured or quickly reconfigurable machines).
- When using a utility like VMWare or Plex86 to emulate multiple machines, have a method in place to sanitize the original operating system, to prevent "hidden" infections.
- Put in meticulous shut-down procedures that someone has to take responsibility for, and sign off at the end of every business day. A simple checklist of removable media secured, machines logged off/shut-down, cell phones accounted for, and so on, can save embarrassment and clean-up from infections, and demonstrate adherence to good, hygienic practice.

Summary

Malicious software defense is a must in any corporation. Effective defense must move beyond the simple solution of AV software and firewalls. A company must develop a full program for malware defense and an enforceable policy. Close coordination is necessary at all levels of the corporation, to have an effective impact on lowering the cost and damage caused by malware incidents.

Solutions Fast Track

Enterprise Defense-in-Depth

☑ The Jericho Forum advocates a strategy of "micro-perimeterization," whereby protection is implemented at multiple levels. This isn't the same as defense-in-depth, but can fit neatly into a multi-layering strategy.

☑ Passive monitoring includes IDSes and behavioral analysis systems.

☑ Proactive detection or blocking is vital. Disinfection and cleanup is often much more difficult with modern malware than it was in the past.

☑ A major defensive measure is to restrict services to the minimum required on a given system.

Malware Detection

☑ IDSes are generally either host-based, network-based, or hybrid. They use either signature detection or anomaly detection, or a mixture of the two.

☑ Intrusion management is best based on IDS, IPS, and other practices such as sound patch management and user education.

☑ SNORT is a versatile, highly configurable IDS with excellent community support.

☑ Defense-in-depth may have preceded the AV industry, but AV was certainly an early proponent of defense-in-depth strategies.

Planning, Testing, Revising

☑ Planning is an ongoing, looping process. In the context of malware management, it's best led by someone with significant experience in the field, but with input from other security personnel and other disciplines such as e-mail and networking.

- ☑ Contingency plans should be developed for a wide range of scenarios across detection, incident management, and user support.
- ☑ At the end of each test (or live incidents in which the plan is exercised), there should be an "After Action Review."

Personnel

- ☑ The person primarily responsible for antimalware architecture should be a specialist in the field.
- ☑ AV and antimalware specialists need management support and buy-in as well as expertise, if they're going to be taken seriously and function in the organization.

Look Beyond the Borders

- ☑ It's safer to let home users and road warriors have access to the corporate network, if similar protection is extended to their systems as is to office systems.
- ☑ You cannot assume that your business affiliates and partners are as well protected as you are, and have to plan for the possibility that the exchange of traffic will pose extra risks.

Documentation

- ☑ Documentation needs to be published centrally and, usually, online. However, you can't rely on its availability online during an outbreak or incident.
- ☑ A useful scenario is to make hard copy available in at least two places, one of them a "working copy" that can be amended as necessary during a test or incident. A third copy should be available online.

Malware Laboratory Procedures

- ☑ Malware samples should only leave the lab under tightly controlled circumstances.
- ☑ All possible measures should be taken to keep the test network separate from production networks.
- ☑ Dedicated "hot" media should be used to transfer files. Such media should not be re-used for more general purposes.

Frequently Asked Questions

Q: Surely security is much more than AV?

A: Undoubtedly. Even within the antimalware arena, multi-layering offers much more capable security than point solutions.

Q: Is it necessary to protect the desktop if you have the gateways covered?

A: It makes sense to concentrate protection at choke points like mail servers and proxy servers, but individual desktops still need protecting and updating to counter unanticipated threats. Redundancy is a survival factor in the world of security.

Q: Surely I don't need AV on every system?

A: There is no such thing as an invulnerable system. Even operating systems that are not generally attacked can harbor malware that affects other systems. Of course, not all platforms actually support a conventional AV solution: many vendors no longer support obsolete versions of Windows, let alone other platforms and devices.

Q: What's an integrity shell?

A: Cohen describes an integrity shell as using "redundancy to detect changes...thus a form of automated fault tolerance and change control...The only way we know of to accomplish this goal is with a hard-to-forge cryptographic checksum (CCS)." The shell he describes counters infection or corruption of system utilities by replacing them with a clean, trusted copy, but is really a specific application of change detection.

Q: Don't generic solutions generate lots of false positives?

A: Vendors hate it when you say that. One way to look at it is that if an object is misidentified as a malicious program or class of program, that's a false positive. If it's identified as potentially malicious (for example, because it's an .EXE file, or contains macros), that's not a false positive, because the software is behaving as it ought. Virus Bulletin FP tests currently make a similar distinction: a file incorrectly flagged as "malicious" is an FP, but if flagged as "suspicious" that's acceptable. (Not everyone agrees with that viewpoint. See the chapter on evaluation and testing.)

Q: Is there any point in blocking attachments by filename extension these days?

A: Yes. Mass mailers haven't disappeared yet, and various types of Trojans are still disseminated by e-mail. Malware authors don't stop using particular vulnerabilities and vectors because they're old. Rather, they add new wrinkles as they're discovered. After all, there's always somebody who hasn't got around to patching or upgrading yet.

Q: Are network shares still a problem?

A: Probably more so than they were when they were first exploited. Bots still get a lot of mileage out of them.

Q: Surely multinationals and enterprises with multiple sites can't set up a single "war room?"

A: No, but they're more likely to have a central space they can set up as such temporarily at their main office/headquarters. Often, it's enough to be able to set up a "virtual" War Room with a few lead actors meeting within a small physical space.

Q: Isn't PGP rather expensive to use for encrypting samples?

A: That depends on where you are and your exact needs. And there are freeware versions with the same core functionality (www.pgpi.org/, www.gnupg.org/). In fact, AV vendors will often accept samples as an encrypted. *zip*, but if you're going to operate a professional lab, you may as well accept the need for professional standards.

Q: Can I use flash drives and the like for transferring malware to and from the test network?

A: While you can normally transport "inert" executable files without much risk irrespective of media, there are real and hypothetical risks to using flash drives even in a production environment. You also need to be aware of other issues such as the potential for accidental execution of malicious files under uncontrolled circumstances. Best practice is to use only read-only media to prevent cross-contamination from a potentially infected network. The real issue here is that by introducing a small, even hypothetical risk, you compromise the commitment of the lab to impeccable hygiene. While it may never lead to an incident, it looks particularly bad if your procedures are audited for forensic reasons.

Chapter 7

Perilous Outsorcery

Solutions in this chapter:

- Key Concepts: Outsourcing AV Services and Risk Management
- Key Building Blocks for Managing Outsourced Security
- The Perils of Outsourcing AV Activities
- The Perils of Outsourcing Management Matrix
- Critical Success Factors for Surviving AV Outsourcing
- Putting the Pieces Together

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

Outsourcing is a commonly considered option today for Information Technology (IT) security managers in public, private, and non-profit entities, and in most areas of information management. Here, we consider outsourcing anti-malware protection as a possible response to the latest security threats caused by malicious code. And although no one will argue that it eliminates the malicious software (malware) problem entirely, managed anti-virus (AV)/antimalware services do work well for some businesses. (See Ed Skoudis' article "The pros and cons of outsourcing AV services" – http://searchsecurity.techtarget.com/tip/1,289483,sid14_gci1048857,00.html). It often makes sense to re-use working procedures across organizational borders, and to contract trained staff by hiring them from a third party, as long as the relationship is based on a clear and binding contract. Potentially, such a contract can make overall security operations less costly and more effective by using economies of scale.

But, beware. A super-cheap service may not offer, for instance, the Service Operation Center (SOC) redundancy necessary to give access to the trained, trustworthy personnel you need in order to ensure the healthy maintenance of business processes. Otherwise, the cycles that make the managed AV services "tick" may start to slow down, even to a point where AV services are effectively halted.

Tools & Traps

Suitably Qualified

In the "Information Technology Security Handbook" from the World Bank, it's pointed out (Chapter 7: Security Outsourcing) that AV services are not exactly a commodity service. As we discussed in Chapter 1, there are no universally recognized international certifications for Subject Matter Experts (SMEs) in AV. There are, on the other hand, vendor-specific certifications and more general malware-related certifications, as offered by Global Information Assurance Certification (GIAC) as well as generalist security certifications like Certified Information Security Manager (CISM), Certified Information Systems Security Professional (CISSP). However, a Critical Success Factor (CSF) in an outsourcing strategy is the presence of suitably qualified staff.

Security consultancy and managed services are still uncharted territory for many people, and it's sometimes difficult for a manager to distinguish the "charlatans, frauds, naifs and novices" from the many accomplished professionals in the field. (Partly adapted from the Information Technology Security Handbook, 2003, The International Bank for Reconstruction and Development, The World Bank, written by George Sadowsky, James X. Dempsey, Barbara J. Mack, and Alan Schwartz: see www.infodev-security.net/handbook/.)

There are fine resources on the Internet and in books on the topic of AV outsourcing, and consultancy firms are ready to guide you with questions to ask before signing the contract and finalizing an arrangement. However, very little information has been published on what happens after that point. What areas must you focus on in order to avoid some of the perils of outsourcing? In this chapter, we focus on the phase after signing the deal, helping you to spot and manage some common threats, and to avoid things becoming unmanageable.

To make AV outsourcing a success for all parties involved, we will look closely at the human CSFs and at what issues to anticipate. We also share some best practices for managed AV services. This chapter is intended to help you to deal more effectively with the impact that outsourcing AV services can have on your daily, weekly, and monthly activities. In order to deal with possible problems, which will most likely differ for each reader and will also vary over time, we address the following points, in this order:

- Key concepts of outsourcing AV services today: what, why, when, and so on
- Risk management behavior and other risk management human factors
- Operational and tactical changes based on three dynamic models that affect the work of AV staff
- Tips on how to manage the perils of outsourcing over time, based on your current and future roles and responsibilities, and by using CSFs and shared tips.

We also consider some key questions for better management of outsourced security:

- What does the term "security activities" imply for a business manager?
- What does "outsourcing AV services" imply in general for the operational tasks you are migrating?
- What determines the failure or success of any outsourced operational AV activities?
- What are common phases during outsourcing AV services for the project manager?
- What are the most common perils encountered when outsourcing AV?
- Why do more and more companies outsource AV services (despite the pitfalls)?

We also offer (free, gratis and for nothing; well, for the price of this book):

- The Perilous Outsorcery management matrix
- Tips on how to solve or survive the perils of outsourcing AV activities
- Sources of CSFs: the more you make explicit, the better!
- Opening lines of peer communication in and between both companies
- An example questionnaire for finding the right people for the right type of AV activities

Key Concepts: Outsourcing AV Services and Risk Management

We start here by assuming you have been informed of a formal contract being signed between a managed services company (and service provider) and your "normal" business; that is, one that is not itself a managed services provider. We assume that the business has selected that service provider in the course of a service procurement project, after carefully analyzing the responses to a Request for Proposal. We also assume that this contract declares, among many other things, the commitment of both companies "to undertake to fulfill as much as possible of a list of AV activities together, starting on [a specific date]." We assume that a legally binding contract has been signed by senior management, and that it has been agreed which security staff at both companies will be working, for a predefined period, to implement any needed changes. We also assume that it is known which staff are to be transferred from the company to the AV service provider at the formal start of the service, so as to manage the AV infrastructure and processes as described in the contract.

What triggered the "desire to outsource" is not particularly relevant to this topic. In most implementations of managed security services, AV is just one of the many domains owned by senior management (unless, perhaps, they work for an AV company) and may even, in many cases, be an implicit infrastructural component of IT services rather than being specifically addressed in the contract. No matter what your work in the AV area was originally, it is relevant to note that outsourcing *any* part of security to a third party will always have a big impact on your work. You need to allow time so that many people can adapt to their new setting, while malcode-related incidents, among the most prevalent security incidents for several decades, will continue to necessitate sensible actions from more and more people. In the first stages, outsourcing any AV service will therefore increase the number of time-consuming tasks to be undertaken before the work can be completed (due for instance, to loss of flexibility or unclear priorities.)

WARNING

Companies often have difficulty developing and implementing a security strategy when they do not maintain an explicit focus on business drivers. This can occur for a number of reasons, but if organizations use outsourcing as "proof" that they have a sound business vision of enterprise security, you should confirm that they're on the right track by asking for their latest approved strategy, and use that in all subsequent negotiations and activities as a "framework."

Recent surveys in financial services (2006 Global Security Survey, Financial Sector, Deloitte Touche Tohmatsu, June 2006) point to the increase in extra security-related safeguards needed to secure the flow of incidents and other relevant information across several parties:

- Bi-annual reviews and learning to deal with the metrics
- Internal audits and extra monitoring of operational staff to ensure due diligence
- Information flows that are extended to "outsiders"
- Audit teams from more than one company, posing questions about your roles and your control of leakage of customer data and so on

Again, this chapter should help you prepare what a sound answer will be, and share the basics by closer examination of the basic building blocks of AV services.

Tools & Traps

What Security Services are Being Outsourced?

Deloitte Touche Tohmatsu reported in 2006, that the most commonly outsourced services were, in descending order:

- Vulnerability management services
- Intrusion Detection System (IDS) services
- Firewall services
- AV services
- Content filtering services
- Virtual Private Network (VPN) services
- Forensic and prosecution services
- Security Training services
- Incident Response services
- Governance

However, Governance outsourcing seems to come way below the other services, which are fairly close in volume. (Deloitte 2006 Global Security Survey, June 2006)

Key Building Blocks for Managing Outsourced Security

Here are some key building blocks to enable better management of outsourced security activities, expressed in terms of questions that need to be asked.

What Do "Security Activities" Imply for a Business Manager?

Outsourcing any business function is a strategic management decision, and a hot topic in (systems) management. It is also a hot potato, because managers must have enough trust to hand over part of the corporate "crown jewels" to a third party, and it will take some time to gain mutual trust. For this reason, only some (rather than all) AV activities may be handed over in the beginning. A growing number of multinational companies take this route each year.

We enter this process after company X has publicly announced its intention to outsource IT services to company Y. In plain English, that means that the AV activities currently performed internally by staff at company X is transferred to party Y, and henceforward, AV teams will be working under different management. This applies even while it may still be unclear to operational staff what, precisely, that work is. Often, it's initially appropriate just to keep operations running and as near as possible unchanged, to allow the staff transferred to meet and get used to new management.

For our purposes, Company X is said to outsource AV services to Company Y if the Board of Enterprise of Company X makes a statement to the effect that "... managing AV infrastructures is not part of our core business activities, and because no relevant laws, internal policies, or ethics forbid us to do so, we have decided that operational AV activities p, q, and z and some of our operational security departments A, B, and C are to be transferred to an external organization. After careful analysis, we are proud to mention our business partner Y. As a consequence, some current staff will have to deal with this new model, and under project 123 and 567 we will address the new job opportunities..."

Tools & Traps

Transitioning and Transformation

It is useful to get a copy of the plans for both transition and transformation, and, using the contract as a reference, see what AV activities are in scope. To check on whether activities are relevant, work with an electronic copy to search for indirectly related AV/security terms like "security incident," "scanners," "malicious," "virus," "detection", "infection," and so on. This helps to establish a formal baseline for any discussion of any plans that relate to the scope of AV activities in the transaction.

It is not usually useful to have a formal discussion on the risks discussed here just after the contract is signed, or worse, to dispute the decision to outsource after signing, unless you have proof that the contract and current "as-is" AV defenses are endangered to a degree where it becomes worth a conflict between the "newlywed" partners X and Y. It is a good idea to inform your line manager that you are aware of some risks to your transferred work area (for instance, that the license for the AV tool will end in two months), and it is a good sign if your (new) manager asks for clarification and enters the risks you mentioned into the risk log, starts to find the risk owner if it isn't him, and so on.

What does "Outsourcing AV Services" Mean?

More specifically, what does the term imply in general for the operational tasks you have been performing?

As will be explained later, when considering which AV activities can be outsourced most successfully, the outcome of strategic decisions always depends on the type of company and how it wants to manage its major IT security risks. A bank needs different information and takes different mitigation measures regarding its AV management to a constructing firm or telecom provider.

NOTE

A bank has to be able to report certain incidents and losses to its banking authorities, to be able to stay compliant with certain banking regulations and internal policies, but a car manufacturer may decide much less information is needed, and therefore less needs to be stored.

If outsourcing AV is a business like any other business, being and staying ahead of the "bad" guys is time critical and does not depend on the business, but on the nature of your AV defenses.

Are you Owned?

Good Eggs and Bad Apples

What is a "bad guy" is somewhat subjective, but might include:

- Your "white collar" competitors
- "Black hat" hackers and intruders of all sorts

Continued

In real life, probably both types plus all intermediate shades of grey, especially if you have to look after valued and/or sensitive internal or external customer data, as financial, health care, and research organizations usually have to.

(See also the Sarbanes-Oxley (SOX) Act of 2002 (SOXA), BASEL II, ISO 17799/27001 and so on).

Also note that incident management will most likely still be a CSF for you, and although we do not deal with CSFs in any detail here, we will stress that you have to focus and keep hitting those moving targets in order to stay successful.

Outsourcing AV services implies a formal agreement as to who "manages" certain components of a company's AV services. In the contract, this is usually described at a high level, to allow room for the parties to make use of their own current best practices. It is useful to first look at the bigger picture of AV services, before zooming into a more detailed view and considering the CSFs for outsourcing AV security.

IT Security activities can be categorized into three generally linked categories: management, operational, or technical. AV services can also be categorized in this way, because these are a subset of general IT security services.

Management of IT security services is focused on managing the IT security programs (based on business strategy and projects) and the risks within an organization, such as security policy development and risk management. Operational security services focus on the security controls implemented and executed by people (not systems), such as network testing or training and awareness. Technical services are focused on security controls executed by an IT system, such as firewalls, intrusion detection, and AV. Table 7.1, adapted from NIST Special Publication 800-35 'Guide to IT Security Services', provides an overall structure by which named IT security services, including AV services, can quickly be categorized.

In outsourcing security deals, it is rare *not* to outsource operational and technical services (usually completely). Due to the nature of AV, these two types of AV services are very common as they are "tightly coupled." We assume this scenario throughout this chapter. The management services often cause the biggest problems, because they are now divided across several companies, and while managers cannot transfer all responsibilities, it is less clear who will be responsible overall and who is accountable to whom.

TIP

Legal regulation like the SOX (Sarbanes-Oxley) regulation, is a prominent example of the perils of this scenario, and in finance, many further regulations and interest groups must exist in order to safeguard "safe banking" against a Board that considers outsourcing as a denial of its overall accountability.

Table 7.1. General IT Security Service Categories

1. Management Services	These are techniques and concerns normally addressed by IT management in the organization's computer security program. They focus on managing the computer security program, and the risk within the entire to which these risk managers belong.
2. Operational Services	These are services focused on controls implemented and executed by people (as opposed to systems). They often require technical or specialized expertise, and rely on management activities and technical controls. Often regional differences are visible, but most are centered on predefined Service Level Agreements (SLAs).
3. Technical Services	Technical services are focused on security controls executed by a computer system. These services are dependent on the proper function of the system for effectiveness. There is a drive to re-use technical standards as much as possible.

What Drives the Success or Failure of Outsourced Operational AV?

It is not the contract, because the contract is more of a mission statement and a way of directing the framework for dividing the activities between parties. It is about the "what", not the "how" of allocation of responsibility. It is a complex, dynamic world in which harmful code is found, but this is not the same world that people live in. We live in the physical or real world, but malicious code is just a bunch of zeroes and ones, existing only in the virtual world of an information system. It has its own weird dynamics, and when and where these two worlds meet, that is where AV service difficulties are found and have to be managed.

For instance, after an outsourcing deal, we very often see a slowing down of the entire AV operation instead of the speedier, more efficient operation that everyone hoped for. To explain this, we share our seven "laws of outsourcing AV activities" in more or less random order.

First Law

AV services "exist" realistically if (and only if) there are agreed descriptions of the activities involved. Only then are you able to manage the virtual world of protection against harmful code as an integral part of protecting the business or the "real world" in a predictable and measurable way. Often, these descriptions are either absent or (at best) not yet migrated, slowing down the AV operation.

Second Law

Malware, malicious code, virus, or harmful code are all interchangeable terms and concepts for most managers, and so (often) are the processes related to AV services. So, in this context you can use the terms AV, anti-malware, anti-spyware, harmful code protection, malware management, and so on. This is nice, but it will not have a positive effect on managing the "perils." As described in other chapters, dealing with spyware, rootkits and so on can be very different from the techniques required for dealing with viruses and worms. During the first phases, what is described is in general, and hence doesn't give enough information to staff needing to know what is (or is not) to be done.

Adding the word "management" (like in anti-malware management services) to the outsourced activities will look more impressive for an AV manager, but makes little difference to the meaning. While we can generally stick to a term like *outsourced AV services* for the sake of clarity, it will nevertheless take quite a while for all staff to understand what that means in terms of his or her own job description, and many never reach that stage Keeping trained AV staff is not easy, and many move on to new positions before the transitional problems of outsourcing are fully under control. This often takes longer than three years, in which time most teams have acquired new members belonging to new organizational units.

Third Law

Changes due to outsourcing are similar to changes that are or have been forced upon you continuously by the never-ending race to exploit the business opportunities of novel technologies ahead of the competition. Looking at this race from a global viewpoint, we all know of some cases where people crossed ethical or legal borders, or ignored policies made to protect the information and processes of the enterprise from any accidental or deliberate misuse. Outsourced AV services deal with this murky area more often than other services, because harmful code is written with bad intentions, meaning that operational staff must stay alert and proactive so as to be aware of new risks appearing daily, as well as maintaining normal day-to-day services.

Fourth Law

Outsourcing AV involves keeping two very different business views balanced and geared together 24/7 for each AV activity:

- The "paying-for-this-AV-service" or *customer* view (also known as the "business owner" or "management") view
- The "providing-this-AV-service" or *vendor* view (a.k.a. "service provider," "technical services," "IT operations," "maintenance") view.

The divergence between these views is easily understood by the following "hired gardener" analogy.

Imagine you maintained your garden yourself for many years, with success. You have contented garden users and friendly neighbors. You mowed the grass and trimmed the hedges, but decided it would be better to contract out to a professional gardening firm, and to pay them a fixed price yearly to maintain the garden for you, to free up your time for your family. Of course, being the owner, you expect to be consulted whenever a change has to be made due to natural events. And you expect to retain your right to change the garden to your own needs, and to use it all year round as a place to relax in. Without clear arrangements as to what your gardener will do and how you control his activities, both you and your hired gardener may end up facing a number of problems and may even face substantial financial losses. If you forgot about the risks of moles ruining the lawns or of trees being chopped down because of a neighbor complaint, then do not be surprised if the flower beds are replaced with strawberry beds, because strawberries happen to be the specialty of your gardener (and much cheaper for him to maintain). All of these changes may be based and justified on that vague clause to the effect that the contractor will maintain the garden up-to-date according to his best insights and endeavors.

Fifth Law

Success in outsourcing AV depends on jointly fighting malcode-related incidents better than the competition. This applies for any outsourced AV activity, and demonstrates that the parties have synchronized their Observe, Orient, Decide and Act (OODA) loop.

This law is aimed at human decision making. Fighting malcode related incidents and preventing loss and damage is a core driver in AV outsourcing. Being able to counter malcode-related incidents effectively and in a timely fashion is a CSF. This is not only because response times are part of the metrics of most SLAs. The CSFs here are that both companies must agree on objectives, and must integrate their decision cycles and operations. This is because the main, underlying objective of the contract is not to manage the malcode incidents, but to protect the customer's business. This is essential for any security professional, because actions and decisions depend on how well you understand the situation you are facing, how effectively you are able to make decisions, and your freedom to act in time under critical and pressured conditions.

That model is best explained by the "aircraft pilot" analogy.

Colonel John Boyd introduced the concept of the OODA loop during the Korean War, based on "the ability possessed by fighter pilots that allowed them to succeed in combat." The OODA cycle shows that any entity (either an individual or organization) that can process this cycle quickly, observing and reacting to unfolding events more rapidly than any opponent, can "get inside" the opponent's decision cycle and will gain a military or business advantage.

NOTE

An overview of the process can be found at www.nwlink.com/~donclark/leadership/ooda.html. A fuller consideration of the OODA framework and its full cycle are found at Wikepedia (http://en.wikipedia.org/wiki/OODA_Loop), which is more precisely applicable for AV management, but also more complex.

In practice, we see many cases where the outsourced, trained "AV fighter pilots" fail, and the companies have to hire (back) their own fighters (for example their outsourced AV coordinator or even entire teams) simply because it turned out to be too costly to be publicly seen to be shut down once too often and the customer organization wants to engage only staff who understand the proper security context of its business needs. Many enter the outsourcing 'AV battlefield' knowing only the side they are supposed to be on. Be sure to know your objective(s) for the strategic time frame, and next to evaluate you and your project or business Strengths, Weaknesses, Opportunities and Threats (a.k.a. SWOT analysis) to maximize your success to achieve these objectives. Many AV staff tend to rush into a posture only to find out after a while things were 'not quite as expected'.

Tools & Traps

Changing Context

A possible pitfall for an IT security professional is that he or she may fail to realize that his or her context has moved or changed with the outsourcing, and hence his or her ability to make effective decisions. It pays off to start with a good observation of the new battlefield, to orient well towards the objectives of any new management, and then to make decisions accordingly. Here is a suggestion for adapting the OODA model to this scenario:

- **Observe** Scan your new environment (as-is) and gather information from it.
- **Orient** Use that information to form a mental image of the circumstances and proper context. Deconstruct old images and create new images.

(What is the change from what you did before and have to do now, and what are the options?). Different people require different levels of detail in order to perceive an event, and people who make bad decisions often fail to turn information into knowledge, lacking the proper context or time to learn from what they observe, so you need to allow enough time here before the next step!

- **Decide** Consider options and select an action.
- **Act** Carry out the arrived-at decision. Once the result of that action is observed, either stop or cycle through the four steps faster and more effectively than the "enemy."

Sixth Law

Outsourcing AV services will not get rid of current AV problems. Customer investment and input is needed in order to support the information flow to the vendor. Old problems must be understood and discussed openly by the customer and vendor, so as to work on this key and new trust/relationship. But as a success factor, it is the customer management who needs to understand that, in order to maintain the AV function, they will have to account for and stay responsible for:

- Using and interpreting the status reports delivered by the AV services vendor, and pointing to (potential) weaknesses in current systems or not living up to agreed service levels.
- Aligning the different cultures of two companies. The most common way is to translate current policies and standards from the customer into those currently in use by the vendor. Of course, a lot of miscommunication will result if no taxonomy is agreed upon when this migration begins. This is a key task to be performed by a trained AV specialist. It should be regarded as ongoing, not as a one-time job, because both parties may change their policies independently. (For example, if a company needs to store details of "major incidents," these two words will not necessarily mean the same to both parties, and hence using them without definition in an SLA will give rise to conflicts later.)
- Managing the vendor relations with their own security team, as well as the contracted team, to maintain appropriate responses to new risks and to encourage detailed investigations to prevent losses (such as an order to do a Root Cause Analysis, or an order to mitigate a hiccough in the OODA cycles).
- Preventing new losses, because no AV solution can be 100 percent effective. In the case of a massive infection, the customer's in-house staff will still need to cooperate by leaving their machines running and allowing remote access, so as to

allow the reinstallation of applications. The rational aim here is to be able to get back to normal business as soon as possible, rather than looking to penalty clauses for restitution.

- Keeping up with, – or better, keeping ahead of, all of the ongoing organizational changes (for the vendor as well as the customer), because that is part of the information flow towards the vendor's operational staff. The customer must thus keep track of its own key people who are appointed to manage the new relationship, and must inspect the quality of all activities and manage the relationship for two organizations instead of one, and maintain communication and information flows.

Seventh Law

Outsourcing AV services means managing a one-to-many partner relation, not just one single vendor. Since the late 1980s, when AV became a common enterprise security requirement, AV vendors have sold their tools with support (e.g., help desk, Web pages, alert services on e-mail, SMS, pager and so on) directly to the customer. Since then, they have continued to offer more support and services to customers, in line with the evolution of e-commerce, Internet, malware, and several layers and levels of AV protection. The big difference with full outsourcing of AV activities today is the scale of what is done by a third-party vendor. How well can the traditional AV tool vendor's work with and support parties acting as an intermediary between the customer and the tool vendor? At the dawn of AV, only a small percentage of functions were transferred and characteristically, only a few functions were implemented, compared to the formalized infrastructures of today. Nowadays, 80 percent or more of functions are commonly migrated, and many subcontractors may be linked into the chain. If that is the case, each party has to be willing to align its procedures for the customer's benefit, or else the problems of outsourcing will become increasingly intractable.

What Common Phases does the Project Manager Encounter when Outsourcing AV Services?

The following phases are common drivers for change in the formal outsourcing of managed AV services. They can be seen more as the strategic cycle that drives the operation, as opposed to the operational tasks that staff must do in an hourly, daily, weekly, monthly, and bi-annual rhythm. Here are some typical timescales for those phases for a company with some 10,000 users

1. **Strategic Reorganization Phase** Decisions made on which IT security activities (and which people) need to stay in-house (3 to 6 months).

2. **Scoping AV Outsourcing Activities Phase** Maintenance of AV infrastructure and support framed by senior management, with budgets and profits set for the next four phases (1 to 3 months)
3. **Request for Proposal Phase** Liaise with AV-managed service providers for a definite selection of framework and set starting date for the contractual phase (the so called Business Commencing Date, the legal commencement of the outsourcing arrangement). At this point, any missing or unclear definitions will yield perils (6 to 9 months, ending with a signed contract).
4. **Transition Phase** People and tools are transferred without modification, to the direct management of the vendor (6 to 12 months).
5. **Transformation Phase** People and tools implement change under the management of the vendor, so that the benefits declared in phases 1 through 3 start to become apparent for both parties (6 to 12 months, where staff changes dictate the pace of this phase rather than changes in tools and technology).
6. **Operational Phase** Ongoing until the end of the contract, which is longer than phases 4, 5, and 6 if all goes well (meaning that phases 4 and 5 are expected to be completed at least a year before the end of the contract). If not, loop back to phase 1.

WARNING

Bigger companies mean longer implementation periods, and you will need more resources to get to the next phase. The trick is to be sure not to use up years for a phase so that, before you know it, budgets have been vaporized. The minimum timescales presented here seem to work for small businesses. These periods are derived from actual experiences in large companies.

In this section, we deal only with the last three phases. So, for our imaginary multinational company, a typical full cycle, where one will meet and have to mitigate the effects of some (if not all) the perils addressed here, typically takes 3 years or more. Every 6 months or so, significant updates, upgrades, and configurational tweaks to the AV tools are needed to stay successful in fighting the "bad" guys, so no matter how well the processes of AV outsourcing are done, each project that is kicked off into the next phase is likely to encounter one or more essential upgrades of AV tools. On top of these issues, and apart from other project pitfalls that must be resolved, some tactical initiatives are likely to be implemented before a stable operation is in place and working as planned.

Tools & Traps

Top Gear

It may help to see all of the building blocks we describe in this chapter as part of a "gear box" of a car to get the outsourced security services running. (These phases can be seen as "shifts" in the "gearbox" of AV outsourcing, an analogy that is used throughout this chapter to illustrate the working of the complex processes involved and how to fix problems as well as possible.)

The main "gear wheels" are, in fact, the key concepts that we shared here. Some gear wheels will look old and rusty, some very new, some parts are usually missing, and some wheels are not spinning because it is unclear who has to do what.

To outsource AV more successfully, all staff and management will have to put effort in and not be frustrated if outsourced AV work is not working as usual. Managers will have to "top up" with the right oil (e.g. add money, trained staff, set objectives and allow architectural changes) to cope with the transitional problems, and IT security staff will need to decide where to stop, reverse, or speed up some operations if that makes the AV engine run better.

We can't resist quoting Seymour Papert here: "I believe that working with differentials did more for my mathematical development than anything I was taught in elementary school. Gears, serving as models, carried many otherwise abstract ideas into my head." ("Foreword: the Gears of My Childhood," in "Mindstorms: Children, Computers and Powerful Ideas.) We cannot comment on how useful knowledge of the LOGO programming language may be in the field of AV outsourcing.

What Are The Most Common Problems Seen During AV Outsourcing?

With this understanding, we can move on from the perils of outsourcing to decide how best to deal with them. Bigger companies mean bigger projects, or more time spent on dealing with these problems, but you are likely to face one or more of these perils (one security expert with experience in the outsourcing of AV, winked and referred to this as a form of "malicious management" as opposed to "malicious code management.")

Miscommunication Between Customer and Vendor

Customers and vendors may have very different views of what the vendor is expected to do. This can derive from:

- A contract in which essential customer requirements aren't specified, or aren't well-specified, maybe because the customer didn't realize they were needed, or because the customer assumed that crucial areas would be covered, or because issues informally/verbally agreed in the initial negotiations somehow failed to be addressed in the contract.
- A contract in which the customer's requirements at the time were pinned down very precisely, but which lacks a mechanism for accommodating changes in the threat landscape.
- Failure to resolve possible areas of dispute over terminology and concept.
- Sloppy agreements that allow vendor "weaseling." "That's not our problem. You need to educate your end-users (all 1 1/4 million of them...)"; "That's not in the contract. Would you like us to quote for a change in specification?"
- Wishful thinking: "It's not our problem any more: it's the service provider's..." versus. "That's not our responsibility. We're supplying a service. We're not here to absolve you of all risks and accountabilities."
- Internal miscommunication between the vendor, customer top management, customer financial and contracts managers, customer tech personnel, project management, and the other subcontractors involved. This can derive sometimes from deliberate fostering by the vendor ("He's asking too many awkward ques tions. Talk to someone at board level instead."). Sometimes it's the result of empire building and politicking within the client organization. "It's not for IT to question how we frame our contracts." "Operational staff won't like us dropping that requirement, but what they don't know can't hurt them." Sometimes it's simply because operational requirements are specified by managers a million miles from the coalface (www.usingenglish.com/reference/idioms/at+the+coalface.html), with no real grasp of the operational issues.

Tools & Traps

The Joys of AV Outsourcing

David Harley lists a number of real-life instances of such problems. For instance:

- A vendor that refused to use all the capabilities of the AV software they administer, having arbitrarily decided that certain types of potential malware and spyware were just spam. "You didn't buy the spam module

Continued

of the service, so we can't activate the detection for potentially unwanted applications." (This may have been an attempt to conceal the possibility that the vendor's "wrapper" program had no way of hooking into that functionality in the core software.)

- A vendor who claimed that the service had not failed to detect a particular (and damaging) instance of malware because "That's a worm: the contract only covers viruses."

Names have been omitted to protect the guilty.

Lack of Responsive and Flexible Threat/Change Management Mechanisms

Symptoms of this deficiency include insufficient involvement of the customer in decision-making and post-incident analysis phases, and a temptation to cover up an inconvenient incident, or even to apply spin and claim credit for "rescuing" the organization from an incident they mismanaged. Many deals start with clearly outdated AV tools, but with no common cross-vendor plans to upgrade as a mandatory requirement, before shifting to the next planned phase. The customer and/or vendor often shift up too brusquely, or even in an unplanned, almost accidental manner. In some cases, management even shifts more than one gear at the same time, which is a sure way to stop or even break your delicate AV gear box!

Procurement and Tendering Conflicts

These can derive from a procurement process that seriously restricts the contenders to one or two heavyweights, where a leaner, more specialized group might offer better service. They might also arise where contenders are restricted to a catalogue of approved providers chosen without reference to their suitability for the job in hand. Invitations tend to demand specifications that are framed at a high level by senior management, and without reference to groups that are most able to determine the technical aspects of the solution requirements. It is not uncommon to see that new departments and staff are needed to deal with this part of the work, where old contracts are not migrated in time, nor renegotiated so as to deal with new situations, or with the current or imminent phase of the outsourcing. When incidents need immediate resolution, these teams can turn out to be a major cause of slow down in that ad hoc defense process. For example, during a nasty but small and targeted Trojan attack to steal money from a bank's internet customers, a cleaning tool was needed from the contracted AV supplier to help customers under attack to clean their potentially infected home systems. Time was lost establishing which contract managing party was responsible for resolving the issue, and legal parties seemed not to understand the urgency, focusing instead on the legal risks involved in the sharing of customer data with the vendor. One AV supplier created the necessary tool and offered to help because they were not hampered by a formal relationship, leaving everyone puzzled as to what to expect next, and what needed to be changed in the contracts as a result of learning this lesson.

A Vendor-Centric Worldview

"We can't do that: it would compromise our ability to meet SLAs." Further examples from Harley of inability to understand customer requirements based on a customer-centric view of what constitutes a problem include:

- A service provider refusing to use beta definitions for a mass mailer during a major mail-storm because any problems with the definitions update would lay them open to penalty clauses for other performance shortfalls. This clearly prioritizes the convenience and profit of the vendor over the primary functionality of the service, and could have been avoided by a clearer "exception negotiation" mechanism.
- A service provider unable to appreciate that a standard feature of their service creates problems for the customer (e.g., the generation of automatic alerts to the apparent sender of a virus-infected message).

Overestimation of a Vendor's Competence

Once the choice is made, the customer has invested more than cash into the transaction. (In fact, this is one of the bases of the continuing success of a 419 scam, when "good money" continues to be thrown after "bad money," because the victim cannot acknowledge that they are being duped.) Of course, we don't intend to suggest that outsourcing should be directly compared to advance fee fraud. Nonetheless, it's not unusual for an outsourcing organization to fail to get past the "denial" phase in the Kübler-Ross cycle (http://changingminds.org/ disciplines/change_management/kubler_ross/kubler_ross.htm). Hopefully, your outsourcing will not launch you into a "grief cycle" comparable to the psychological trauma of death, but here are some factors to watch out for:

- Verbal reassurance that a service "is able to do that" when it turns out it can't (especially when reference to that capability somehow disappears from the contract).
- A psychological need to rate the vendor's expertise over that of your in-house personnel because it cost $0.25m for their initial feasibility study and $8m for the contracted service.
- The customer's inability to properly evaluate the initial risk, the range of solutions available, or the competence and experience of the supplier, so that undue weight is given to what is essentially a sales pitch.
- Emphasis on "Swiss army knife" solutions. Some providers tender for a wide range of security (and other) solutions, but have no in-house expertise specific to a

specialized field. Suppliers whose expertise is in setting up a "one size fits all," low maintenance service, may not have corresponding in-depth knowledge of the problem the services are intended to address.

- Services based on a poorly understood, minimally configured set of open source products and services. One more example: a vendor trying to compensate for a serious underestimate of his or her own costs, by substituting an unsupported open source scanner for a full-strength industrial solution.

Buying a Solution on Financial, not Functional Grounds

- "What can you offer us? Which is cheapest? We'll take it."
- "Of course it's good. Look how much it's costing us..."

The Perils of Outsourcing AV Activities

Outsourcing a current in-house AV activity will always generate some problems, because transferring an AV activity from one to another company, on top of your presumed existing difficulties (why else are you outsourcing?), means that operations will be affected and people need to adapt. Many companies hope that farming out AV to a third party will free up in-house personnel for other work, but will AV activities be carried out to the same standard by the third-party implementers as by an in-house team? (See "Fact, Fiction and Managed Anti-malware Services" by David Harley, 2003 Virus Bulletin conference proceedings.)

Many common and necessary AV activities are addressed in other chapters of this book, and we assume that you know what they are about and how to do them, technically speaking. The main AV activities that are potential candidates for outsourcing should, ideally, have been described. It should be fully understood what will be the impact on the work of at least two people in the two companies.

The next potential problem is the need to agree to changes to the level (standard, norm, or quality) you need to perform a given AV activity in the new context. Unclear SLAs are SLAs that do not specify the Level exactly. It is tempting for the customer to demand the best quality ever, and for the vendor to confirm or proclaim they will provide it; however, failing to quantify that level will give you headaches. So you need to look at or obtain descriptions of AV work that look like "deploy any publicly available AV update to counter a new threat within 24 hours for more than 98 percent of the systems to be protected."

For considering these aspects of quality, you need a checklist to specify the same, lower, or higher standard than before outsourcing. It is common to change this level as the service is outsourced, and that is fine, as long as it remains clear to all affected parties.

And finally, you will need very careful resource planning during all phases, and make enough time to deliver explicit (new) descriptions of any work transferred, as procedures must be implemented in two places to ensure that a critical activity doesn't stop working altogether.

The perils of AV outsourcing are thus almost always organizational by nature, and are based on human factors. Outsourcing AV means that people from different companies with independent management, different missions, different cultures and varying experiences have to start to work together to manage a complex and dynamic security domain. Next, your personal success will depend on how quickly you can spot any imbalance in this crucial customer-vendor relationship with respect to each outsourced activity, and how quickly you can make stabilizing adjustments to counter such imbalances. For that purpose, we present later on a Roles and Relationships (R&R) matrix, as well as a questionnaire to be adapted for use with respect to your own organizational changes.

Why Do More and More Companies Outsource AV Services?

Given the problems cited above, it is not obvious why companies want to outsource some or all in-house AV activities and infrastructures, and decide to make room for a third party. But more and more companies do so because:

- It offers better control over most AV activities. By managing vendor(s) using formal contracts and services, management can be freed from dealing with many technical details, while retaining overall control.
- It makes economic sense. Customer companies are buying because the outsourcing company is selling the idea that it will cost them less than keeping an in-house team active (at least, that's the management theory.) Managers like the idea of managed AV services in just the same way as they like paying for clean water, 24/7. By analogy to an infection with dirty water from a badly maintained source (instead of clean water), the decision to outsource AV looks like a no-brainer. The argument is that the costs of seeking new sources and maintaining pipes are too high for a normal company. But harmful code is associated with virtual risks, and is not particularly like water provision. Pollution incidents are very common, their causes are not ultimately curable, and losses are unpredictable and much more dynamic. If water providers were liable to problems comparable in variability, frequency, and impact, there would be far fewer of them.
- It offers more flexibility. Selecting a service provider, and allowing the vendor to deal with delivering AV services or functions in its own way, allows both the customer and the vendor to operate more independent of actual locations. As countries where labor is available at low cost are sometimes politically less stable, switching to another region at need is now possible, and it is easier to replace a team that fails to meet SLAs or to perform, provided that AV services are not dropped when new competitors are invited. This benefit relates also to the OODA cycle we referred to previously.

- There are functional benefits purely for the customer, and others applicable to both the vendor and customer: trained staff with up-to-date knowledge of the latest threats can be re-used. (See: http://searchsecurity.techtarget.com/tip/1,289483,sid14_gci1048857,00.html, by Ed Skoudis.) After all, security staff in most non-specialized organizations have problems keeping up with the rapid advance of harmful code and its complex countermeasures. A more robust, affordable AV infrastructure can be built with multiple, parallel, or regional SOCs, virtual servers can be deployed to share costly infrastructures, and on top of that, many AV service providers can offer a more solid infrastructure, which translates into operations/performance benefits for all, however, not immediately, as each change will cost time and money to overcome the problems.
- AV services are an integrated component of other outsourced IT services and management infrastructures. It is often risky and costly to keep the AV infrastructure separate, if most other IT services are outsourced.
- There are risk management benefits for the customer's (senior) management. Today, companies spend a lot of time proving that their security services are compliant with respected standards (like ISO/IEC 17799 or ISO/IEC 27001). It is easier to impose a process for compliance or certification using a compliance model like the Plan – Do-Check-Act cycle on your vendors, than to start checking your own staff. (This cycle will be included in the AV gear box, because it drives the ITIL management processes just as it does OODA.)

AV services are always associated with symptoms of bad code and bad ethics, and as a cost factor. But for decades now, AV is a must-have, and is seen as "sort of not sexy." Managers will love it if they don't have to deal with operational and technical AV issues themselves.

Are You Owned?

Plausible Deniability

Security managers in both companies must keep in mind that someone could hypothetically hack both organizations via the shared network, or spread malware from one company network to the other. While the customer and vendor are committed to work together on AV activities, senior management will do their utmost to keep a clear perimeter demarcation (even in the coupled networks), so as to avoid becoming responsible for any AV risks they are not in a position to manage. This may result in additional technical issues, and can stop AV functions from working when access is denied to do the work for this reason. Managers tend to react stereotypically if confronted with a specific AV function in order to stay disengaged for as long as possible. For instance, "we need to upgrade a scanner to a new version to stay compliant with the SLA as follows" is likely to generate a sequence of responses along the following lines:

- Deny the reality and ignore the risk. If that doesn't altogether succeed (because policy, law, controls and so on militate against inaction), then...
- Deny a personal accountability and responsibility. If that doesn't altogether succeed, then...
- Transfer your part of responsibility to another internal party. If that doesn't altogether succeed, then...
- Transfer that part of responsibility to another external party. (If that is partially successful, the same cycle will start in the other company), but if that doesn't altogether succeed, then...
- Mitigate the risk for your part. If that doesn't altogether succeed, then...
- Accept that you own that risk and that you need to see to it that others manage all related activities for you. Next, search for the Critical Success Factor needed to manage that risk, as it is now part of a managerial responsibility.

It is at this point that we can implement many of the techniques and technologies in the book in practice with the best chance of succeeding.

- Technical benefits for service provider and customer (e.g. better AV defenses). Many managed AV vendors are better able to deploy multiple scanning technologies, or blocking rules integrated to single layered defenses (see above), which gives provider and customer an advantage, because they are better equipped to implement multiple signature-update cycles. Malware often bypasses a certain configuration or type of AV tool, but not all, so using multiple engines ensures a better, multi-layered protection methodology.

WARNING

AV services fall into three distinct layers or types of infrastructure: e-mail-centric solutions, the perimeter defense layer (gateways to Internet and other companies), and the desktop/server layer. It is often found that layers are added over time, but not integrated with each other, and even split across competing AV service providers. You must avoid introducing too many vendors by outsourcing bits and pieces over time, even when the supporting environments are already outsourced. AV services work best when implemented end-to-end.

The 'Perilous Outsorcery' Management Matrix

We now present a high level framework that we call the "Perils of Outsourcing Management Matrix." This framework, presented as a set of tables or matrices, is not a "magic cook book" from which one can pick and choose recipes. Each migrated AV function is part of a complex chain of services, and you must still add many personal judgments and adapt it to fit what is most applicable to your own current situation.

The First Dimension: Use The Job Descriptions, Roles, and Functions of People You Meet

The seven following typical internal roles in our fictional multinational help you to identify (your own) role(s) and later help to identify the likely CSFs for that role:

- General line manager and senior manager
- Security administrator, security manager, and information security professional
- IT service delivery manager, IT support staff
- Human resources professional, educationalist
- Security and risk compliant manager (e.g., auditor)
- Enterprise or regional policy maker
- Vendor contracts manager
- Project manager (and team) to implement the changes around the AV outsourcing project
- End user at the customer site

On top of this, six typical external roles for our fictional multinational exist:

- Vendor of anti-malware and/or AV services
- Vendor of anti-malware and/or AV scanners/packaged software for AV software, AV tools
- Staff member from law enforcement agency or government
- Member from branch-specific organizations
- AV researcher, consultants, or other professional from another external body
- The company's customer

TIP

It is good practice to ascribe the vendor roles (external role number 1) so that they map to the same internal roles, mirroring each person to his counterpart in the client organization. People do not work alone, but in teams, so reading a business card or a two-minute introductory chat is not enough to assess whether an activity is a fit or mismatch for you or another member of your team. You will have to do some background checks on employees to make sure they meet the new business needs of both companies. For that purpose, an AV services-related questionnaire will help you to classify who is best for what job, and a sample questionnaire is presented later.

The Second Dimension: AV Function Types from Risk and Systems Management Perspectives

Successful AV outsourcing will have an impact on all roles above, but should not negatively impact the customer's relationship with its own customers, or the quality of the work done. To manage the quality of the services, another dimension is added to the work as a check to see if AV activities are broken down in sufficient detail to make sense to all involved staff.

Ideally, each basic AV activity that is outsourced fits into one of these two types (seen from a risk management angle):

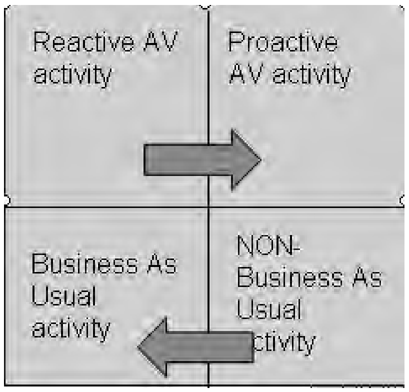
- **Proactive** AV activity, including issues like policies and standards creation and enforcement, an awareness memo, system and network administration, or deployment of new AV tools, all intended to prevent losses due to harmful code entering the business.
- **Reactive** AV activity, including issues like incident management, report analysis, monitoring to limit losses due to harmful code detected entering (or having entered) the business.

Likewise, each basic AV activity also belongs to one of these two types of AV systems management:

- Business As Usual (BAU) AV activity, including issues like monitoring for IT support calls, common incidents handling to be covered in the new SLAs.
- Non-Business As Usual (Non-BAU) AV activity, including issues like handling an emergency outbreak, an AV vendor stopping support, or no "natural" current defense due to new dissemination factors that current AV tools cannot handle.

Figure 7.1 shows the simple classification for a basic function, for example `Update the AV tool'. The arrow indicates a way to improve that service's overall effectiveness over time (phase 5,6).

Figure 7.1 Classification for a Basic Function



TIP

If you come across an activity that seems to fall into all four quadrants, then that activity is insufficiently detailed and should be discussed with stakeholders in order to avoid the perils of miscommunication. Add the underlined words to that vaguely defined activity and then get back to the people involved to ask for clarification of the objective and the SLAs. If the issue is still unresolved, go to management so they can see that their success is dependent on each quadrant of that function.

The Third Dimension: Type of Governance Role Using The RACI Model

The most critical component for successfully navigating the perils of outsourcing is the governance dimension. It deals with the scoping of AV activities in the context of outsourcing to a new "owner" of each function. This is to make clear to both the provider and the customer who should (or should not) be doing what, once the contract has been signed. Much confusion can be prevented by avoiding multiple overlapping or failure to allocate responsibilities for all outsourced AV work under the contract. Here emerge the biggest potential problems, because so many vague terms and differing policies and guidelines, originating within each company in the partnership, plus the legal terminology, must and will be mixed all the time with many complex and indirectly interrelated AV functions.

Few projects and companies succeed in doing everything right in a single project cycle. One frequent reason for this is that the current roles have not been well defined in a common model for change management, or for accommodating the losses that happen to occur because of wrong decisions.

The most common way to describe the process where a company has to migrate activities and roles/responsibilities from one party to another is to apply the acronym RACI to the migration of each outsourced AV activity. These four letters stand for:

- **Responsibility** Identify the person(s) having responsibility and will be expected to participate actively in the AV activity, to the best of their abilities.
- **Accountability** Identify the person who is ultimately responsible for the output (performance) of AV activities, and whose position must stay in-house (accountabilities cannot be outsourced).
- **Consultation** Identify the person(s) who have particular expertise to contribute to making specific decisions about an AV function (that is, their advice will be sought), or who must be consulted for some other reason before a final decision is made. For example, contract management is often allocated to a consulting role for outsourcing projects, but security experts often seem too hard to find.
- **Inform** Identify the person(s) who are directly affected by functions and decisions, and therefore need to be kept informed, but who do not participate in the effort to define the migration of the AV activity.

TIP

One important lesson during the outsourcing project is that you should not leave the assignment of responsibilities to chance. Organizations avoid innumerable conflicts by taking the time to make deliberate and accurate choices about who will be responsible for what. The number of all basic AV activities that fit nicely into the matrix of Table 7.2 is generally high: in fact, it's more likely to be over 100 for our average 10,000 user company than less than 10. Low numbers here indicate only that high-level AV functions are being discussed, indicating a lack of the detail necessary to consider, for instance, metrics or staff allocation.

An Example of the "Perils of Outsourcing" Matrix

By mapping the RACI elements to each AV activity, using colors to indicate personnel in different companies that occupy that role, and assigning the RACI letters and the names of the department\functions\group, it becomes clear what the impact is on each person, and which issues can be addressed where.

As standard tools cannot handle a three dimensional table well, it is more practical to add the second dimension as a new row (as shown in this table for a specific AV activity), and discuss it with all affected people (according to role or function title):

You should not be disappointed if things are not working very differently after a year, and in fact should be glad if they still work at all. AV is very dynamic (making it complex to outsource), and numerous critical success factors must be kept in focus. The more exotic the demands, the less likely the chance of quick and eye-catching successes, Also, follow the Keep It Simple Stupid (KISS) principle and design a gear box with as few gear wheels as possible, and have trained staff (not many, but competent) to shift the gears.

Table 7.2 Sample RACI Matrix for a Single Basic Activity; Metrics can be Added According to Department Name/Function and Job Title (Role)

AV Function	Dept. ID or Group/IT Function	Trigger	Get Sample	Sharing Results	Store Results
Send a sample for expert analysis after reported incident. Detailed SME task, so no multiple responsibilities					
Type					
Non-BAU (no SLA) and proactive (results needed to prevent recurrence)					
Senior Security Manager (Internal)	Security Officer	A			
Line Manager (Internal)	Credit Services Program Manager	I	A	A	I
Security Manager (at AV service provider X)	Outsourced IT security X	R	R	R, C	R

Table 7.2 Continued

AV Function	Dept. ID or Group/IT Function	Trigger	Get Sample	Sharing Results	Store Results
AV Security Professional at service provider	Outsourced AV team X	C	C	C	C
AV Lab Tech supporting vendor Y's AV tool	IT support Lab Y		I	R	
Security and Risk Compliance Manager (Internal)	Legal and Risk Management	R	C	I	A, C

To validate a matrix for an outsourced AV activity quickly, the (sub) tasks should have precisely one A and at least one R; letters "A" and "R" cannot be put in one cell together. At least one R belongs to the vendor if the activity is to be categorized officially as being outsourced. The other letters (C and I) are optional, and can be added whenever needed for more quality, clarity, or to allow others to adapt to the new way of working more quickly.

It is useful to make up the matrix not only for the imminent scenario, but also for the one that applies currently. In many companies, some basic AV activities are already transferred. Shifting them again will give rise to conflict in vendor relations that can be very hard to overcome. Nobody likes their work taken away without any warning. AV staff remaining in-house will, however, have a clearer view of any changes to their roles, and can discuss that impact with the vendor managers.

Critical Success Factors for Surviving AV Outsourcing

Here we address the most important CSFs. The excellent publication " The Critical Success Factor Method: Establishing a Foundation for Enterprise Security Management" (Richard A. Caralli et al, July 2004) defines critical success factors (CSFs) as "key areas of performance that are essential for an organization or manager to accomplish its mission or objectives." The authors make clear that any AV operational activity must be tightly couple to one or more operational (business) goals in order to obtain or keep management support for what will become your AV gear box, and hence your way to succeed in outsourcing AV.

So the obvious question is: do you know your mission and business objectives, and what you need to accomplish that? If you don't know, ask your manager for more information. Managers should implicitly know and consider the key areas when they set their goals, and direct operational activities and tasks that are important to achieving those goals. But operational staff are usually less aware of the (often complex) hierarchy of goals and missions.

Once the attached key areas of performance are made explicit (this is another CSF called 'defining metrics' which we talk about later) these CSFs provide the common point of reference on AV activities for the two (or even more) involved organizations.

Both customer and vendor staff must somehow understand the CSFs for any outsourced activity or initiative undertaken, to ensure consistent high performance in key areas: otherwise, the organization will probably not be able to achieve its goals. Consequently, both may fail to accomplish their mission, AV outsourcing will fail and malware management may even be brought back in-house at some point.

Sources of CSFs: the More Explicit, the Better!

CSFs are generally allocated (see Richard A. Caralli et al) within the sphere of influence of one particular manager. There are many levels of management in a typical organization, each of which usually has a very different operating environment. For example, an executive-level manager may be focused on the external environment in which their organization lives, competes, and thrives. In contrast, line-level security managers (you?) may be concerned with the operational details of the organization and are therefore focused on what they need to do to achieve their internal, operational goals.

Rockhart (Rockhart, John F. & Bullen, Christine V. A *Primer on Critical Success Factors*. Cambridge, MA: Center for Information Systems Research, Massachusetts Institute of Technology, 1981) defined five specific sources to scan for CSFs for an organization, and these CSFs should be aligned with both the customer and vendor(s) as follows for each manager:

- The industry in which the organization competes or exists
- The understanding of the organizations' peers
- The general business climate or organizational environments
- The problems, barriers, or challenges to the organization
- The layers of management

Internal CSFs are those CSFs within the span of control for a particular manager. In contrast, *external* CSFs are those over which a manager has very little control. For example, in the AV services industry, an internal CSF could be "managing the monitoring security operation," while an external CSF may be "license fee costs."

Categorizing a CSF as either internal or external is important, because it provides better insight for managers in setting their goals, and affects how they respond to new events. For example, a manager can set very specific, achievable goals that complement the achievement of internal CSFs, because the manager has control over them. However, if a manager has an external CSF, he or she must set goals that aim to achieve the internal CSF and to minimize any impact on operations that may result, because the external CSF is not fully in his or her direct control.

It is not too hard to see that AV services do depend on all five CSF types, and, depending on what layer of AV management you have in place, it should be clear what your external CSFs are (not your direct responsibility) and which are your internal CSFs. The two must-have operational CSFs are from NIST (NIST Special Publication 800-35 "Guide to IT Security Services"):

- SLAs document and articulate by whom and how AV services will be handled, in sufficient detail to show:
  - AV service provider responsibilities (service tasks, documentation, service support, reporting requirements)
  - Customer responsibilities
- Metric(s) for each SLA to verify and check if the SLA is met, exceeded, or unmet. (For further guidance on developing metrics, see NIST Special Publication 800-55, Security Metrics Guide for Information Technology Systems.)
  - Specific objectives, metric(s), service level, service level assessment
  - Agreement to account for changing requirements (costs, period of performance, dispute resolution, remedies for non-compliance, maintenance of agreements)

Open Peer Communication Lines Between Both Companies

At a minimum, appointed personnel should not only be familiar with AV-related risk and performance information, but should be able to discuss it and share information freely between customer and vendor peer. So once it is clear who is who, it should be okay to contact your peers and share so as to get the obstacles out of your way. A number of items are well known for triggering information flows and discussions between the customer and vendor (based on the "Information Technology Security Handbook, 2003" – The International Bank for Reconstruction and Development/The World Bank; Chapter 7, Security Outsourcing; George Sadowsky, James X. Dempsey, Barbara J. Macklan, Schwartz:):

- Employment, employment law, and Human Resources (HR) management issues that may predict conditions under which insiders may harbor a grudge against their employer(s)
- National and local computer crime laws
- AV vendor support contract issues
- Encryption products, technologies, and limitations
- All issues relating to viruses, worms, and other harmful code, as well as detection software

- Transmission Control Protocol (TCP)/Internet Protocol (IP) fundamentals and issues of Virtual Private Networks (VPNs) and firewalls
- Patching and (reporting) unpatched systems or exploits.
- Awareness and educational issues, including sharing materials and services, level of employee training and expertise. Also, how to keep all personnel updated and assure at least a year of experience for all AV and SOC personnel.
- Issues of incident response and forensic investigations (often assigned to yet another provider)
- Security issues peculiar to replacing old hardware and software
- Best practices, formal risk assessments, and methodologies (like the Information Technology Infrastructure Library (ITIL); see below)
- Insurance issues. Get a written statement of ultimate responsibility for failures. If the implemented hardware or software exposes your data to the outside world, or unexpectedly crashes systems during peak business hours, this is not a good time to discover that you have agreed that some partners have no liability.)

TIP

It really helps to think, whenever an AV issue is brought up in AV service team meetings, how that issue fits into the agreed method(s). If it has been agreed to use the ITIL framework for security management and the Dynamic Systems Development Method (DSDM) to deal with the transformation as core methodologies, best practice is to seek what is already agreed in that context, and work down towards AV operations. This is simply more direct than working upward from the Operations team to find the managers dealing with these common issues.

In a real example, it turned out that the AV teams were worrying about the End of Life of the primary AV product and how to mitigate the consequent risks, as their CSF was based on the number of uncontrolled incidents. But, using ITIL, they discovered that they were neither responsible nor accountable with respect to timely upgrading, but the Change Manager was. And because he owned this CSF, he dealt with that issue using a global transformation plan.

ITIL is a de-facto standard for IT Service Management (ITSM) developed in the late 1980s by the United Kingdom Central Computer and Telecommunications Agency (CCTA).

Use a Questionnaire to Match People to AV Functions

The AV-related questionnaire is a good way to verify AV functions, and to do some background checks on employee AV skills. Thus, you can make sure your team can meet the SLAs for any new AV activities. The questionnaire will also help you to see whether a person is better suited to more of basic type of BAU or Non-BAU activities, and/or to proactive versus reactive roles.

For example, if you are an AV researcher trained in specifying new countermeasures for your company, your training is likely to be very different to that of your AV systems manager or a person that is trained to deal with end-user support. The outcome of the questionnaire may well mean that some people do not fit well in the new structures, or that you need to ask for some training.

It is also logical that staff and management are reviewed because of outsourcing, and therefore face changes in their roles and activities. Being forced to move on and to meet your new manager makes most AV people nervous when they face outsourcing. To counter this trepidation, you can start to fill in an AV questionnaire for each member of your new team, and do so again after a year or so to see how scores have improved. Each answer will give credit points and with that, after scoring, an AV service team can be made from existing staff, because it is easy to see who is best suited for what.

A sample AV questionnaire is found at the end of this chapter, as an example of what to ask when you have to empower an AV operations team for a customer and work for an AV service provider.

TIP

If you are transferred with the function to the new provider, it pays off to know as soon as possible if a certain activity fits your current or new available role, or that of the other candidates. If there is a mismatch, one can plan either AV-related training or select another person. (Mismatches appear to be very common and, in general, if the boss says you have to migrate, you probably have to start 'transforming' or adapting somehow, or quit. In the European Union, labor laws for outsourcing offer some protection to preserve your work, but the effectiveness of such protection depends greatly on protocols behind such a labor contract/ labor counsel deal.) To reflect the current projects status, repeat the questionnaire after a year or so: first, update the questions, and then send it out again to the teams. Once the new team stabilizes, it is time to update the inventory or, if there is still no data there, start a detailed research on the AV inventory in and out of the scope of the agreement. Be careful to report any boxes not in scope to both the vendor and the customer.

Align as Soon as Possible with Monitoring Services (SOC) and Incident Management Teams

A key area that everyone will focus on is Incident Management, simply because malicious code incidents have been the most public incidents in ICT for many decades, and most managers will measure overall performance by assessing incident handling effectiveness. Each company will have (had) some internal AV monitoring before outsourcing their AV services, so the first discussion that needs to be resolved is the response to or recording of these two simple questions:

- How do we all share information that we have a malware incident (or worse, a malware disaster)?
- Once we know that (our) human intervention is required, who will do what? How can we do so, being careful to minimize losses for all parties involved?

Oddly enough, these questions need to be asked over and over again, whenever changes occur, including new hardware and software, new services, new products, and new staff.

AV services are not just like water from the tap, or as regulated in the same way as filling in a tax form. Many SLA's may not specify properly what an incident is (and outbreaks are not the same as many detections per second.) So, who is allowed to push the alarm button and who is expected to do so?

TIP

In many cases, the actual monitoring lies in many hands (e.g., different providers) and correlation of systems from multiple sources becomes a major task. Once you have some idea of who is who, who are your peers, and how the management layers are laid out in both companies, it pays to start to work on a list of CSFs for each person in the relevant teams you encounter. Use the output from the previous section by asking what each person wants to achieve. Next, transform these answers into CSFs. Often, using just the AV SLA and its metrics will help to show what the individual does to fulfil operational unit CSFs, and how that supports the other, higher enterprise CSFs, motivating everyone to do their part.

Having multiple SOC's and support teams, monitoring and alerting tends to get very complex, making it difficult to see who is doing what and why, Because much depends on individual cases, we won't address this in detail. The best tip we have is that if the project teams start to work on the processes (or on the determination of how to implement malicious code incident management processes) with key players in incident management, you should be there yourself as well as later during the transformation phase, to ask the right questions above.

Outline the AV Infrastructure (as Seen by the Customer and the Vendor) and Discuss Differences

A high level review of the current AV architecture and technologies used is needed, not only to understand the technicalities, but also to separate your data and network from those of the other party in the agreement. After the teams address the two questions above, you can approach the best team members to start to build the hardware catalogue. In most deals, vendors are primarily expected to manage the AV scanning services or infrastructure, but the contract is not necessarily exact as regards the inventory. Using the questionnaire approach again, send a simple spreadsheet to the AV service provider team appointed to manage and monitor the AV scanning services, and have them send back the statistics regarding any unique machines, type of machine (notebook, desktop, file server or application server, infrastructure component), and statistics regarding:

- Exact level of operating system (e.g. WinNT 4, SP4), and within that the level of AV scanner (e.g., Symantec Corp. Edition, build 10.1.5000)
- General AV infrastructure components (e.g., AV monitoring consoles, quarantine servers)
- Unknown/other devices. This is a key item to quantify in each company. There are inevitably some boxes just sitting there, but the AV risks of these are not known or clearly owned. In any large company, you find hardware that is awaiting replacement or testing is not serviced by a known party, and so on. Knowing that these devices are there, it helps to ask questions during real incidents. To engage all and share these numbers is vital in order to enable a response to any change request or incident. One real example concerned an instance where outsourced staff had to work with two desktops. One special set was needed to monitor and test quarantines that were collecting malware from many places, partly based on incidents from machines not serviced by this company, but by the competitor. One was a "normal" workstation that had AV running, but the other system had AV that had not been supported for many years, and constituted a major risk. So, it took a while to identify the owner of that risk and initiate proper risk management.

Next:

- Ensure that you ascertain the functionality of every group of machines in place, to be certain that you and all associated AV staff understand how necessary that group of machines is to the customer.
- Get a written statement of vendor responsibility for failures.

- Ensure that due care has been and will be taken in developing, testing, and deploying the technology being added to your systems, especially if it is proprietary in nature.
- Understand whether the serviced technology actually helps to prevent problems from occurring, or only detects problems after they have happened (e.g., virus prevention versus virus detection.)

Align or Prepare the Reporting on Compliance Issues with Outsourced AV Services

The compliance process is another loop that is to be entered, or a gear wheel to keep spinning albeit one that revolves far less frequently than the ITIL or OODA cycles, which are more continuously cyclic. This cycle is more commonly known as the Plan-Do-Check-Act cycle. Often, the trigger for entering this loop is a complaint from the customer's end users or, more positively, a requirement to check whether objectives are being met. This action fits best in the tactical area, so slots between the strategic and operational tasks as described before.

As can be seen from the risks mentioned, audits and compliance checks serve a very important business purpose for everyone involved. Not only as a means of demonstrating to senior management that many risks are "controlled risks," but also to show that AV processes are not violating laws or simply ineffective. They also help to improve the image of the companies involved, to strengthen any newly weakened links, and to improve efficiency on the part of both the customer and the vendor. Why? Because often the world has changed, but the agreed procedures that have little use today have not been updated, yet lack a review trigger. Yet IT managers will not usually cheer when auditors announce an audit, as it means work not planned for, or an additional risk if something or someone appears not to be working as anticipated.

It is very wise to prepare for compliance reports and embrace compliance whenever possible. First, because it will help to show what information is considered vital, and second, because any sound outsourcing deal will include at least a few lines on compliance requirements, and will demand that some security-related information is shared with customer auditors to assure them all is working as planned.

So, it is not a question *whether* you will enter the "hamster wheel of compliance," only *when!*

WARNING

It is beyond this chapter to detail every possible compliance check and reporting requirement. It is sufficient to understand that any such audits can be and will be carried out both by the vendor audit teams and by the customer, and that any mismatch will trigger a fierce discussion as to who will have to do what. Often the process of resourcing the auditors is not made explicit (e.g., nobody's CSF).

The R&R matrix will again be very helpful here for finding the managers that need to be engaged. If this is not yet clear, a spin-off of the audit will most likely be that the matrix is a CSF for one or more managers to create. (See "Embracing Compliance," ISSA Journal l, April 2007, article by A. Atri.)

TIP

In large service companies with a dedicated security and compliance function it is not uncommon to start with a so-called pre-audit. For example, the security manager at the servicing company may do a check prior to a formal audit. That check helps a lot in understanding the weakest links, the CSFs, and the metrics that auditors will also want to look at. It also helps to improve the OODA loop and the ITIL-relevant tasks (e.g., the ones tied into incident management and change management), and will strengthen the complete AV gear box. If not already scheduled as part of the outsource deal and plans, it is good to plan these audits for at least your own parts in the AV gear box.

Putting the Pieces Together

From the previous chapter, you have seen a long list of building blocks and issues, ranging from project phases, pitfalls, and a matrix of R&R, to CSF and the ITIL, PDCA, and OODA gear wheels. So, how do you fit all of this together so it works for you?

First, assume we have no AV services split up, so the customer and vendor still work virtually as one and the same entity, working with the same models and having the same goals and mission.

Second, add the operational CSFs for your AV activities by finding all the managers willing to "oil your AV gear box."

Once all the AV activities are described and accepted, you have a working setup that will be able to ride the rough roads of anti-virus management, assuming a 100 percent transparent and cooperative relation between vendor and customer, because both serve the same mission. The objective now becomes that for each company, at a minimum, the gear wheels for the PDCA, OODA and ITIL loops are running at the same speed, work 24/7, and the overall processes are moving in the same direction in order to achieve clear business objectives.

For simplicity, we can assume that the basic AV security activities are the usual supporting activities, for instance processing credit card data in a bank or selling tickets for an airline.

In reality, AV outsourcing splits at least one or more AV functions across two parties (vendor and customer). It is important to remember that for business management, security activities are always seen as supporting business operations. The same is not true for the vendor, whose business objective is actually to deliver the demanded service. This is simultaneously a good thing for AV staff because of the better commitment, and a bad thing because of the energy needed to keep the two different objectives aligned. Misalignment is, in fact, one of the major risk areas.

In an ideal world, what the vendor does should work without any customer interaction. It is vital to align the OODA, ITIL, and PDAC loops securely at all times for both parties. Otherwise, some of the outsourced activity will have a tendency to slip, slow down, or even stop.

TIP

This is also seen in the OODA cycle Orientation, which step will slow down if the parties differ in their previous experiences, differ in their cultural or ethical behavior, do not share the same previous experiences, or both get different new information (from different systems) on how and what they are doing. The answer is to share more instead of less, in order to keep the outsourced AV service up in the air, and to meet and communicate at all levels (from senior managers and air commodores to operational AV fighter pilots).

Slowdowns are common, and if that happens, parties are usually committed in some way to a different set of gear wheels, SLAs, and security controls. For example, they use very different AV management tools and change management or incidents models, have different operational goals, or have split up some AV functions differently according to country, site, or stakeholder. Then you will need even more oil to prevent basic AV functions like incident reporting and timely updating from becoming a huge problem. The entire AV process will become an operational risk on its own if the AV services start to slow down, or even stop completely because staff are unclear on what (not) to do.

It is not always clear if the overall benefits outweigh the risks. So it is up to you and all involved with outsourcing AV to combine all the parts into one function and keep monitoring the overall performance. It is, however, easy to see that adding more parties, off-shoring, and so forth will lead to exponentially more work to keep all running up to the same speed. More rather than less work is then the outcome, there are more decisions to take, involved staff will have a harder time to keep up to speed, and the overall process (serving the mission of the companies) may be hindered by harmful code. Quite a challenge!

TIP

If you have an operational problem with meeting an SLA with respect to predefined response times, it helps to first look at the timing constraints that all companies impose on that activity. For instance, it may well be that demanding a solution to a new malware attack implies the assistance of the AV researchers of an AV company. Such a company will have its own OODA and ITIL cycle, which will slow down the response. Such effects must be made an explicit part of the SLA to ensure the success of the agreement.

Roles and Responsibilities

This list is (adapted) from page 2-1 of NIST Special Publication 800-35 "Guide to IT Security Services." The list of participants who will select, assess, implement, and/or manage an AV security service will depend on the type and scope of the service, service arrangement, and type of organization. Key players can include the Chief Information Officer (CIO), contracting officer, contracting officer's technical representative, IT investment board, IT security program manager, IT system security officer, program manager/procurement initiator, and privacy officer, among others.

Chief Information (Security) Officer	The CI(S)O is responsible for IT planning, budgeting, investment, performance and acquisition, and oversees senior organization personnel in obtaining efficient and effective IT (security) services.
Contracting Officer	The Contracting Officer enters into, administers, and/or terminates contracts and makes determinations and findings related to those contracts.
Contracting Officer's 'Technical' Representative	The Contracting Officer's Technical Representative (COTR) is a qualified employee appointed to act as the Contracting Officer's technical representative in the management of the technical aspects of a security related contract.
IT Investment Board (or equivalent)	The IT Investment Board manages the capital planning and investment control process.
IT Security Program Manager	The IT Security Program Manager develops or applies enterprise standards for IT security, and leads in the introduction of an appropriate, structured methodology to help identify, evaluate, and minimize IT security risks. These managers coordinate and perform system risk analyses, assess risk mitigation measures, and build a business case for the acquisition of appropriate security. They also support senior management in ensuring that security management meets the organization's needs.
IT System Security Officer	The IT System Security Officer ensures the security of an information system throughout its life cycle.
Program Manager (Owner of Data)/ Acquisition Initiator	Program Managers play an essential role in security because they have been involved in strategic planning initiatives of the IT security service and are intimately aware of functional service requirements.

Continued

Privacy Officer	The Privacy Officer ensures that the service meets existing privacy and data protection policies and standards
Other Participants	The list grows with the complexity of the acquisition and management. The system certifier and accreditor will make critical decisions nearer to the end of the procurement process, so should be included earlier in the acquisition to ensure that critical issues are addressed and resolved early in the process. System users may help the program manager to determine the need, develop and refine the requirements, and assess the delivered system. Information technology, configuration management, design/engineering, and facilities groups may also be represented.

Sample AV Skills and Experience Questionnaire for an AV Service Provider.

In this example, the maximum points possible per question were given as they were used in an actual case. Feel free to change them as well as to drop or add questions to suit your case. Of course, neither the points one gets for the questions nor the total score are meant to be made public, so as to avoid scores that are put in there just to look good. And, of course, this is personal data, and hence must be dealt with according to privacy rules, data protection legislation, and so on. Be sure before you start that you have a written agreement from your manager and all other parties including HR, that asking for such data is okay. Of course, ensure that all results are stored in a secure place!

TIP

It is possible some people will send you answers that give them a score of almost zero points. This suggests that they have been assigned to do this work, but have no clue why or how, and it would be useful to discuss this openly with them and with management.

0.	**General Contact Details AV Service Team Member**	**RESPONSE** with maximum score (weight points are from real example)
	Name :	
	Customer e-mail address if any:	5
	Vendor e-mail address: Location maildrop code: Country: Local Time zone: Cell phone: Work phone:	
	Certifications (like CISSP, CISM, CISA, GIAC GSEC, MSCE and so on: you might want to weight some certifications more than others):	5/each
	Vendor employee total of paid hours per week	
1.	Please indicate your regular working hours and days when we can contact you for AV service issues (by telephone under normal circumstances)	
2.	Do you object that your contact information are shared with:	
	A) Existing Customer AV service contacts ?	Y/N
	B) Current Customer AV service vendor's contacts?	Y/N
3.	If you have a back-up person working for AV services for the Customer, what is the name and e-mail address of this person?	5
4.	How many years experience (counting back from [current date]) do you have doing AV services in a large network (e.g., 10.000+ machines coupled to an internal network and the Internet)?	5/yr
5.	Please describe your security function and check each role that fits you	10 each
6.	How many hours per day or week did you work on AV services on average, over the past 6 months?	1 point per hour

7.	How familiar are you with the AV service products below used on the Customer site?	
	A. Customer's primary AV tool	10
	B. Customer's secondary AV tool (if any, repeat if needed)	10
	C. Generic blocking (e.g., filter rules that block certain file types in Customer network or mail flow)	10
	D. Other AV service products (more generally said other AV, anti-SPAM or Anti-Spyware products not part of the contract).	10 per product
	Please specify such other product names	
8.	Are you a member of and/or do you use the information of any of these AV service related organizations (Specify M for Member or U for "I often use their info feeds" or N if you don't use this resource)	Member = multiply score by 5
	A. Anti-Phishing Working Group (APWG) www.antiphishing.org/	1
	B. Anti-Spyware Coalition (ASC) www.antispywarecoalition.org/	1
	C. Anti-Virus Information Exchange Network (AVIEN) www.avien.org/	4
	D. Common Malware Enumeration (CME) http://cme.mitre.org/	2
	E. Computer Antivirus Research Organization (CARO) www.caro.org/	1
	F. Computer Incident Advisory Capability (CIAC) www.ciac.org/ciac/	1
	G. Cooperative Association for Internet Data Analysis (CAIDA) www.caida.org/	2
	H. Distributed Intrusion Detection System (DShield) http://dshield.org/	1
	I. European Institute for Computer Antivirus Research (EICAR) www.eicar.org/	1
	J. Internet Storm Center (ISC) http://isc.incidents.org/	2
	K. SANS Institute www.sans.org/	2
	L. FI-ISAC www.nvb.nl/index.php?p=11523	1
	M. United States Computer Emergency Readiness Team (US-CERT) www.us-cert.gov/	1

	N. Virus Bulletin hwww.virusbtn.com/	2
	O. WildList Organization International www.wildlist.org/	1
	P. Other: pick the most valuable (examples)	
	ISSA NL	1
	Idefense,	1
	Secunia	2
	Customer alert services	1
	ISCA	1
	Microsoft bulletins	1
9.	What is the geographical and Business Unit scope of your AV service responsibilities?	region = 15 (nations, more than 20 locations), local = 5
10.	Do you have an actual view of all new, in-the-wild malware that is hitting the Internet?	5
11.	Do you get customer malcode-related alerts (e.g., from retained security staff, CISO, end users of helpdesks)?	5
12.	Do you have a real-time view of all harmful code that is hitting your part of the Customer network?	5
13.	Can you share a report of all harmful code that hit your part of the Customer network in the past 24 hours?	5
14.	Can you share a real time report of the deployed signatures files of all AV service scanners in your part of the Customer network from the past 24 hours?	5
15.	How many people work on AV service in your region/your people network?	
	A. Estimated number working today at Customer	5 each
	B. Estimated number working today at Vendor	5 each
	C. Estimated number outside Vendor and Customer	5 each
16.	How many boxes do you monitor for AV service activity/services?	1 per 100, max 100

17.	What % of these are running current versions of the scanner and OS systems software, and are fully supported by the vendors and its servicing parties/owners?	
18.	Do you manage Customer AV quarantines? If Yes, specify the vendor of the software	
	A. Customer primary AV product	5
	B. Customer secondary AV product (if any, repeat if needed)	5
	C. Other AV service Products (more generally, other AV, anti-SPAM, or Anti-Spyware products not part of the contract).	5 per product
19.	Do you have platinum vendor support, with access to AV vendor's experts, for the AV service tools mentioned?	5 per product
20.	Do you have the privileges to modify current Customer AV service infrastructure configuration settings (are you a domain AV service Admin)?	20
21.	Have you ever worked to contain an active infection in more than two physical locations (an active infection is defined by the moment in time when 25 machines detect the same AV in an area)?	5
22.	Are there any critical /major issues within your area you wish to share? For example:	
	A. Technically (e.g. cannot download updates or patches, no Customer direct network connection) Please name the most critical one.	5
	B. Organizational (e.g. no management support AV service tasks , no back up , lacking resources)	5
23.	Is your focus mainly on Prevention AV damage or mainly on Reactive response to actual AV incidents in Customer (indicate %)?	R = max 10 (BAU), P = max 10
	Count BAU AV service (normal tactical operations)	
	Count Non-BAU AV service (expert, pro-active level)	
	TOTAL COUNT	

Summary

Outsourcing isn't rocket science. In fact, it's often not scientific at all, but something much more subjective. It's more of a way of tightrope walking with two ropes instead of one. So, day-by-day you pick the most stable rope, even if both are still too wide apart for comfort.

The lines can be seen as symbols of the respective information security strategies of the service provider and the customer. At the start, they are poorly aligned, but after a while, the two ropes will be stable and will even be stronger than just a single rope.

It's a matter of common sense to keep alignment with management and projects and demonstrate added value. So if outsorcery is a black art, it is at least possible to summon some spells, formulae, and models to take much of the uncertainty out of the process.

Solutions Fast Track

Key Concepts: Outsourcing AV Services and Risk Management

- ☑ Initially, outsourcing AV services will increase rather than decrease the number of time-consuming tasks that have to be done.
- ☑ A Deloitte Touch Tohmatsu report in 2006 indicated that the top four commonly outsourced security services were, in descending order:
  - ☑ Vulnerability management services
  - ☑ IDS services
  - ☑ Firewall services
  - ☑ AV services

Key Building Blocks for Managing Outsourced Security

- ☑ It is recommended to get the plans for transition and transformation and, using the contract as a reference, see what AV activities are in scope. It's not usually useful to start discussing the risks or questioning the decision to outsource after the contract has been signed.
- ☑ If an AV service is a business like any other bought-in service, staying ahead of the "bad" guys is time critical. It does not depend on the business, but on the nature of your AV defenses. Incident management is likely to be a CSF.

- ☑ Security activities can be allocated to three linked categories: management, operational, or technical. Operational and technical services are often outsourced together, as they are generally tightly coupled in deployment.
- ☑ The OODA framework is based on a model that was first applied to fighter pilots in the Korean War, consisting of four steps that are repeatedly cycled through: Observe, Orient, Decide, and Act.

The Perils of Outsourcing AV Activities

- ☑ Unclearly defined SLAs are a major source of AV outsourcing problems.
- ☑ Outsourcing problems are almost invariably organizational, and based on human factors.
- ☑ Companies outsource for a number of reasons, including perceived advantages of high-level control, economic factors, and flexibility, as well as functional benefits.

The 'Perils of Outsourcing' Management Matrix

- ☑ The matrix presented here works along two dimensions:
  - ☑ Job descriptions, roles, and functions
  - ☑ Type of AV function from risk and systems management perspectives
- ☑ Each basic AV function belongs to one of two types of AV systems management:
  - ☑ BAU functions like support and routine incident handling
  - ☑ Non-BAU activity, such as emergency outbreaks, withdrawal of vendor support, or a change in the threatscape that nullifies current AV tools.
- ☑ The RACI migration model identifies the individual(s) who have Responsibility and Accountability, are qualified to offer Consultation, and who need to be Informed about the migration.

Critical Success Factors for Surviving AV Outsourcing

- ☑ CSFs are key areas of performance, essential to achieve objectives
- ☑ Rockhart defines five specific contexts to scan for CSFs: the industry in which the organization competes, the understanding of its peers, the general business climate, the problems/barriers/challenges, and layers of management.
- ☑ SLAs specify by whom and how services will be handled; metrics for each SLA verify performance.

Putting the Pieces Together

- It is fundamental to the process to align the OODA, ITIL, and PDAC loops.
- Where an SLA meets an operational problem, it's worth looking at the timing constraints imposed by all parties.

Frequently Asked Questions

Q: What steps do I need to take to maintain necessary incident information flows?

A: Six monthly reviews, establishing useful metrics, internal audits, monitoring of operational staff performance, extension of information flows to "outsiders" (customers, in particular), and allocating auditing duties to more than one company.

Q: Do you think that AV outsourcing is a good idea?

A: It can work better than haphazard in-house AV management. What you mustn't do is:

- Expect that just the act of signing a contract will solve any problems you have right now with your AV infrastructure
- Expect an immediate and painless improvement in performance. To accomplish the equivalent of a well-tailored in-house solution requires meticulous planning and management, and even then, you should expect some hiccoughs while "bedding in."

If you have an AV function that is working adequately at the moment, it's often worth running it as-is for a while and transferring staff and responsibilities gradually.

Q: Our company outsourced all our AV two years ago and it just worked.

A: Consider a claim like this (based on a million posts to alt.comp.virus, SecurityFocus and so on). "I changed to WonderScan AV and I haven't had a single virus since." What is the poster really telling us?

- No one is sending him viruses any more.
- WonderScan isn't detecting the viruses that are being sent.
- WonderScan is detecting and eliminating viruses as it goes along, with perfect transparency.

Now extrapolate this to a managed AV service.

- The virus problem has gone away.
- The service isn't catching viruses.
- The service is catching viruses and dealing with them without bothering you with the details, as it's supposed to. Or is it?

If it is, you're buying a service without monitoring its effectiveness. Don't mistake "no obvious problems" for "no problems." Remember that malware infestation isn't the only problem (and it often isn't obvious.) For example, how do you know you're not losing customer information through false positives and inappropriate generic blocking?

Q: Why can't I just leave it all to the service provider?

A: Because if you don't have a good incident information flow, you don't know that the service is being run economically and effectively. You may not even be able to assess any negative impact on your business processes. You also need a realistic appreciation of the technicalities. David Harley quotes the example of a provider who supplied monthly reports with an impressive total of e-mail-borne viruses. However, the total was a little less impressive when the breakdown by name was checked, and turned out to include several thousand messages carrying the EICAR test file, sent for testing purposes. Furthermore, because the service had no quarantine function, there was no way of validating heuristic detections and likely false positives.

Q: Can you summarize the "laws" governing success and failure in operational outsourcing?

A: Succinctly:

- The functions involved must be defined formally
- Most managers have only a generalized knowledge of malware and AV service processes, so terminology can be a very imprecise guide to an agreement.
- Outsourcing changes are similar to other changes due to response to competitive pressures.
- AV outsourcing entails balancing very different views of the transaction; customer versus provider.
- Success depends on jointly fighting malware better than the competition, demonstrating that both parties have synchronized their OODA loops.
- Outsourcing does not get rid of your current AV problems. You need to establish excellent communication with the provider and pin down the details of the transfer, including a full consideration of the current status of the service.
- Outsourcing AV is a one-to-many partner relationship, not just a one-to-one vendor/customer relationship.

Q: Where can I find more general information on Roles and Responsibilities?

A: NIST Special Publication 800-35 "Guide to IT Security Services" Roles and Responsibilities (http://csrc.nist.gov/publications/nistpubs/)

Chapter 8

Education in Education

Solutions in this chapter:

- User Education from an Educationalist's Perspective
- Security in Education
- Not Exactly a Case Study: The Julie Amero Affair

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

In the first section of this chapter, David Phillips, a security specialist for the UK's Open University with longstanding experience in virus research, considers the problem of user education from the point of view of the educationalist. Since he also presents a highly rated Open University course on computer security threats and defenses called "Vandalism in Cyberspace," he is well qualified to consider this viewpoint. Since David is too modest to go into details about the course itself, we've included some information in a sidebar.

The section on security in education, by David and Judith Harley, builds on a shorter contribution to the second edition of Terry Freedman's "Coming of Age: An Introduction to the NEW World Wide Web" (http://web2booklet.blogspot.com/2006/10/draft-table-of-contents-16.html; www.terry-freedman.org.uk/db/web2/. Judith teaches Information and Communications Technology (ICT) at a secondary school in the UK, and David offers security consultancy to that sector from time to time. Neither claims expertise as regards the US educational system. However, David takes the opportunity here to consider some not-very-US-specific issues thrown up by the Julie Amero case, which was generating a great deal of controversy at the time this chapter was written.

Tools & Traps

Vandalism in Cyberspace

"Vandalism in Cyberspace: Understanding and Combating Malicious Software" is a ten-week course run by the Open University as part of its Technology Faculty's Relevant Knowledge program (http://tscp.open.ac.uk). It's presented online via a protected Web site, which houses the learning and assessment materials. It is described as "an introduction to the downside of computing," focusing on spam, hoaxes, and malicious software (malware). Most relevantly to this chapter, the course is intended not only to teach protection from such nuisances, but also to demystify them by teaching, for instance:

- The differences between different types of malware
- How those types of malware work
- What motivates spammers and how it affects Internet users
- The motivation behind malware authorship
- The impact of malware on business

- A summary of anti-malware legislation
- Systematic countermeasures

You can get information on this and other Open University courses at www.open.ac.uk. David Phillips also presented a paper at the Virus Bulletin conference in 2003, examining the approach taken in this course towards tackling the malware education issue. ("Malware: Educating the Users!," Virus Bulletin Conference Proceedings, September 2003)

The Open University is the largest University in the UK, and was a pioneer in distance learning, attracting candidates for first and advanced degrees who are unable to attend a conventional university, either because of accessibility issues or because they don't have the pre-undergraduate qualifications often required. See:

- www.open.ac.uk/new/what-is-the-ou.shtml
- www.open.ac.uk/about/ou/p3.shtml
- http://www.open.ac.uk/new/introducing-ou.shtml

User Education from an Educationalist's Perspective

The antivirus industry has developed a somewhat pretentiously entitled tripartite (three-part) model for categorizing the kinds of damage malware can cause. This model groups damage under three main headings: Availability, Integrity, and Confidentiality:

- Availability relates to individuals' ability to access their system and data
- Integrity relates to confidence in the data being in a state you intended
- Confidentiality relates to secure information and user information

Are you 0wned?

Damage: the Tripartite Model

Here is a list of the potential types of damage under each of these headings.

Availability

- Deletion of data or programs
- Renaming of files or programs (so that you cannot find them)

Continued

- Encryption of files, disks, or system areas (to make these inaccessible)
- Unauthorized calls to operating system software (e.g., via a program that starts up your network connection without you requesting it)

Integrity

- Corruption of system programs and system areas (changing programs to do tasks other than those for which they were originally created)
- Modification of data stored on your computer
- Corruption of programs or data files

Confidentiality

- Password theft
- Forwarding of personal files to a third party
- Forwarding of confidential information (e.g., bank account details) to a third party

Security professionals often take this model as a starting point for the protection of their organization. This is often because it is a good basis for a business case to put before the board in order to secure funding. Convincing the board to spend profits on protecting the work environment with no obvious (or at least immediately visible) gain to the company's well-being isn't easy. From the point of view of the security industry it may be working as it should, but one factor left out of the security equation is "the user." Many times I have seen presentations that fail to take this in to account. You can put all the software and hardware security measures you can afford into place, but without proper training, the users will find a way to bypass your expensive, 100 percent secure system.

In an article in Virus Bulletin in January 2006, Vesselin Bontchev wrote: "Users are just as stupid and ignorant as they were five years ago. In fact, I have a message for anybody who still harbours any illusions that user education can work to solve the virus problem on a large scale: face it, folks, if user education was going to work, IT WOULD HAVE WORKED BY NOW!"

- Is Vesselin's statement correct?
- With any statement like this you have to consider a number of factors:
    - Who is the writer?
    - What is his or her perspective?
    - What is the context of the statement?

The writer is a respected professional in the antimalware community, but I believe his perspective is from the point of view of a professional researcher in a technical specialty, not that of an educator. This leads us to suspect that the antimalware industry often has a paternalistic perspective on the issue. "We supply the software: your job is to install it. Do not fiddle and we will protect you." The trouble is that in many cases, if you do not fiddle with the settings the vendor has kindly set as a default, some of your software may no longer perform as it should.

The final point is that without reading the full article, you can never fully understand in what context he was making this statement.

You may wonder why I have started this section in this manner. As educators we are taught never to take statements on face value: you have to consider many factors before you accept the statement. Therefore I now want to examine the issue of how we can educate the users in protecting themselves, if this is possible at all.

First, I want to make a point about what we are trying to achieve. The computer industry runs many training courses that are aimed at company staff. An education establishment runs courses to teach its students. I feel this marks a boundary of difference between the two ideals. Training does not teach users what they need to know, so much as it conditions them to interact appropriately in the environment in which they find themselves having to work.

A critical factor here is that the user, in the context of their job, is trained in an environment that, in many cases, they find themselves having to be in. Some may be there purely by choice, but many are in some sense forced to be there by the need to earn a living.

In the home, however, the environment in which the user learns changes. They are learning because they are interested in activities they choose to take part in, without that element of compulsion. When you get this change from "have to" to "choose," the learning needs to change, too. The user now learns or seeks to be taught, not trained, in their choice of activity. They want to understand the issues well enough to enable them to use their home systems safely.

Therefore, the question you should ask is this: "Do I want to train my users in computer security drills, or do I want them to learn how to protect themselves, which in turn will protect the company?"

In education, one of the key factors in enabling home users to learn is to make it personal. That is, make them see the issues that may affect their personal and social environment. If you want the end-user or staff member to learn, you should consider taking the problem out of the work place and into the home. This, in turn, may change the staff members' attitude from, "Oh, another thing I have to learn at work" to "Oh, this will sort out the problems when I download my latest album, or when I chat to Aunty Jean in Australia."

Changing a person's perception of *why* they are learning can have beneficial effects too. When you make an issue personal, it can make that person want to learn, rather than need to learn. Something fuels their enthusiasm to learn the subject matter when they are personally engaged with the subject manner. If you increase a person's enthusiasm for a subject, this will have the knock-on effect of making it easier to learn and increasing their will to learn. This can be the opposite of learning in the work environment, where people are obliged to learn so they do it, in a manner of speaking, under duress, which has the effect of making learning more difficult.

If you make it personal and the user learns how to protect their computer system in the home, this will have the effect of increasing your protection in the work place. If you have best practice working in the users home environment, they will bring that working practice in to the work place.

How do you make it personal? First, let's look at some security criteria you may wish to apply to your company.

Tools & Traps

Key Building Blocks of a Security Policy

- Assess the risks
- Establish policies and procedures
- Create an outbreak response plan
- Deploy appropriate tools (anti-virus, personal firewall, IPS, anti-spam)
- Define update strategy
- Document the policy
- Foster user awareness and education
- Review the policy regularly

Most awareness programs appear towards the end of a security policy. Unless you have all the other points in place, you have nothing to teach the staff. When you get to "User Awareness," consider what you want to achieve. You need the staff to apply commonsense to your environment to keep your data and company safe and trading.

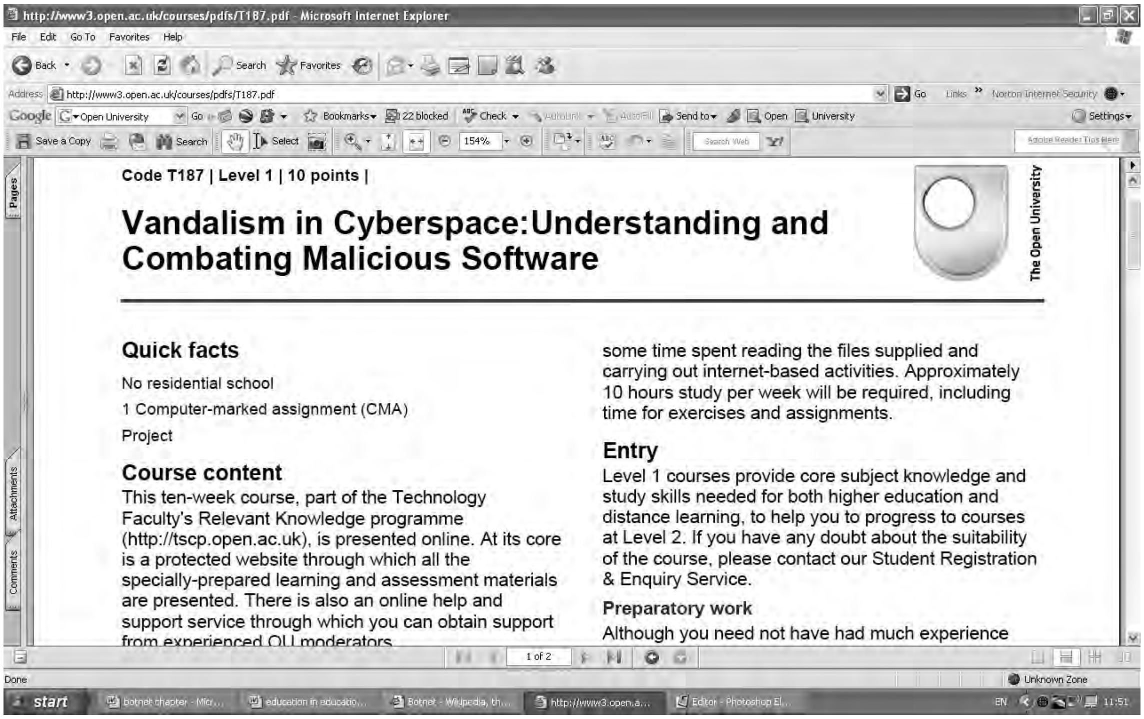
Keep your policies simple. If you over-complicate the issues, your staff will give up and turn off. All your security software and updates may be in place, but that will not give you total protection from unthinking staff. If you have forced them into training that is over-complicated and too low-level, they will switch off. They will absorb little or none of the training and will not be able to apply that training in the work place.

Personalizing the content in ways in which they can identify conveys the impression that the company is offering training and willing to teach them ways to protect their home environment. This will grab their interest.

Below are a couple of ways to introduce the personal touch to user education (taken from the Open University course. T187, Vandalism in Cyberspace). This is a 10-week course aimed at the home user, and in the first week it is centered on personalizing the learning curve. Put the issues in a context that fits their own home environment. Grab their interest so that as they progress. They will add better security to their home, and they will probably then apply those lessons to their work environment.

Grab their attention in the first week—even the first hour—of the training session.

Figure 8.1 Vandalism in Cyberspace


http://www3.open.ac.uk/courses/pdfs/T187.pdf - Microsoft Internet Explorer
File Edit Go To Favorites Help
Back Search Favorites
Address http://www3.open.ac.uk/courses/pdfs/T187.pdf
Go Links Norton Internet Security
Google Open University Go Bookmarks 22 blocked Check AutoLink AutoFill Send to Open University Settings
Save a Copy Search Select 154%
Pages Attachments Comments
1 of 2
Done Unknown Zone
start botnet chapter - Micr... education in educatio... Botnet - Wikipedia, th... http://www3.open.a... Editor - Photoshop El... EN 11:51
The Open University

Code T187 | Level 1 | 10 points |

Vandalism in Cyberspace:Understanding and Combating Malicious Software

Quick facts

No residential school

1 Computer-marked assignment (CMA)

Project

Course content

This ten-week course, part of the Technology Faculty's Relevant Knowledge programme (http://tscp.open.ac.uk), is presented online. At its core is a protected website through which all the specially-prepared learning and assessment materials are presented. There is also an online help and support service through which you can obtain support from experienced OU moderators

some time spent reading the files supplied and carrying out internet-based activities. Approximately 10 hours study per week will be required, including time for exercises and assignments.

Entry

Level 1 courses provide core subject knowledge and study skills needed for both higher education and distance learning, to help you to progress to courses at Level 2. If you have any doubt about the suitability of the course, please contact our Student Registration & Enquiry Service.

Preparatory work

Although you need not have had much experience

Some True Stories

The key component of computer systems is not hardware or software but "liveware"; the people who use the technology. And we sometimes forget, when reading media stories about virus attacks and other malware exploits, that their main effects are experienced by and impact upon people rather than computer systems. So here are some real-life stories, gathered from our collective experience of the effects of malware on colleagues, friends, and family. All of the stories are true, but names and locations have been changed.

The Grandmother

Alice is 75, left school at 16, and has had no higher education. She worked for some years as a secretary, and later as a school playground supervisor. She is a quiet, rather timid person who lives with her husband in the Midlands. Her husband is technophobic: he dislikes and rather fears "gadgets." They have a mobile phone, but never switch it on; it's for use only in emergencies. Her children have grown up and moved away. One of her sons lives and works in the Middle East.

Some years ago, when visiting her grandchildren in the South of England, she was fascinated by the way the children accessed the Internet, went on eBay (an online auction site), used instant messaging, and sent and received e-mail. "Why don't you get a computer, Grandma," they said, "then we could send you e-mail." Her first reaction was that she was "too old" for this stuff, but eventually her daughter and son-in-law bought her a computer and installed it for her, with a free dial-up account to give her an e-mail address and Internet access.

At first, Alice was delighted with her new computer. She used it to play games like Solitaire and for e-mail. (Her husband used it only to play music CDs.) Checking her inbox for new e-mail was, for a time, the high point of Alice's day. When she received an e-mail, she printed it out, composed a reply (on paper), then typed the reply and sent it off. Although her grandchildren teased her about this, she said it was the only way she could work: the idea of composing on a screen was anathema to her. She used the Web very rarely, because she was conscious of the phone charges clocking up.

After a few weeks, Alice began to receive e-mail messages from complete strangers, most of them offering goods or services of no relevance to her. She found some of the messages disgusting. She was getting spam; unsolicited e-mail. Some messages had a link on the bottom of the message saying "Click here to unsubscribe from this mailing list" or some such instruction. She did this whenever the opportunity presented itself. But instead of fixing the problem, it got worse. Her mailbox was rapidly overwhelmed with spam. Sometimes she received over 50 in a single day. She had difficulty locating the few legitimate e-mails she was receiving in the middle of all this junk. Instead of being a pleasure, e-mail became a nightmare and she stopped using it.

Eventually, her grandchildren discovered the problem, and one of them cleaned up her mailbox during a visit and explained some of the basic principles of dealing with spam, including the rule that one should never click on a "click to unsubscribe" link, because that merely confirms to the spammer that your e-mail address is valid.

Alice now uses her computer a lot less than she used to. To her, it now feels somehow "contaminated."

The Sister

Gordon is a Scot who has a sister, Janet, who is severely agoraphobic, that is, afraid of open spaces. As a result, she is very introverted and rarely goes out. She lives alone on a housing estate. She dislikes using the telephone, and is an awkward conversationalist on it. So Gordon had the idea of buying her a computer and showing her how to use e-mail. She took to it immediately, and for the first time in years Gordon and Janet were able to communicate easily with one another. Among other things, Gordon used to send her family photographs as e-mail attachments. Then one day Janet received an e-mail, apparently from Gordon, with an attachment. She clicked on the attachment and her computer was infected with a virus; one of the variety that ransacks victims' address books and dispatches virus-infected e-mail to addresses it finds in them. In the end, Gordon realized what had happened, had his computer disinfected and arranged for Janet's to be cleaned up also. But Janet was traumatized by this experience, and now refuses to read even the text of an e-mail message. Gordon has found that the only way he can communicate with her is to put his entire message in the "Subject" line of his messages.

The Father

Tom is an academic with teenage children. The family has a broadband connection to the Internet. One of Tom's sons is very interested in music, and used the file-sharing program Kazaa to download and share music files across the Internet. Tom knew nothing of this, but noticed that the speed of the family's broadband link seemed to have decreased. Thinking that perhaps his laptop (which his son used) had a virus, he brought it in to his department's anti-virus expert and asked him to check it out. His colleague discovered that Tom's laptop was configured as a Kazaa server (i.e., as a machine that served music files on a large scale), and also that his son's file-sharing habit had resulted in the machine becoming riddled with "spyware," which are programs that covertly monitor what the user is doing and report back to some outside location. At the same time, Tom read media reports that the Recording Industry Association of America (RIAA) (the trade body that represents record companies) had begun prosecuting individuals who were illicitly sharing music files on a large scale. He was, frankly, terrified by this, and mortified by the fact that he had had no idea what was happening on his laptop.

The Young Girl

Rachel is ten years old. She campaigned vigorously to be allowed to have her own e-mail address. Her father signed up for an AOL account, mainly because the company's advertising material emphasized facilities for implementing "parental controls" that he assumed would protect her from reaching undesirable Web sites. But within a few weeks, his daughter began receiving spam messages. He only realized this when she asked him, "Dad, what is Viagra?"

The Self-employed Professional

Hazel is a self-employed journalist, and a single mother with three lively daughters. She earns her living by doing freelance editing and some PR work for local companies and organizations. She has a PC and a broadband link to the Internet. Although the computer is absolutely essential for her work, she has little technical knowledge and because she has an Arts background, none of her friends are "techies." In the last two years, she has been infected with a number of computer viruses. She has no idea how they got into her computer (though her daughters also use it for e-mail, instant messaging, and file sharing), and had no idea where to look for help. In one case, she wound up being helped by someone who formatted her hard disk, thereby wiping out all the program and data files. Luckily, she had backup copies of some of her work on floppy disks, but she still lost a lot of material and several weeks' earnings as a result.

The Unwitting Spammers

Alison and Mark are a young professional couple who have recently moved to the US. They are not really computer-literate, but use e-mail a lot to keep in touch with family and friends in the UK and elsewhere. They were delighted to find that the neighbourhood in which they lived had broadband access, so they signed up. They were then surprised and alarmed to discover, within a few weeks, that their Internet Service Provider (ISP) threatened to terminate their service because they had been identified as spammers. This seemed so preposterous that at first they concluded it was a case of mistaken identity, but the ISP remained obdurate. Eventually they consulted a lawyer, who employed a technical consultant (at considerable cost). The consultant quickly discovered that the couple had not installed a firewall when they hooked up to broadband. (Why should they? They weren't techies and their ISP had said nothing about firewalls.) The inbuilt firewall in their copy of Windows XP was turned off by default. What had happened was that their computer had been penetrated by a spammer, who had covertly installed a Trojan (i.e., a program that turned their computer into a relay station for spam). And Alison and Mark had known nothing of all this, even though it was happening under their noses.

And the Point is...

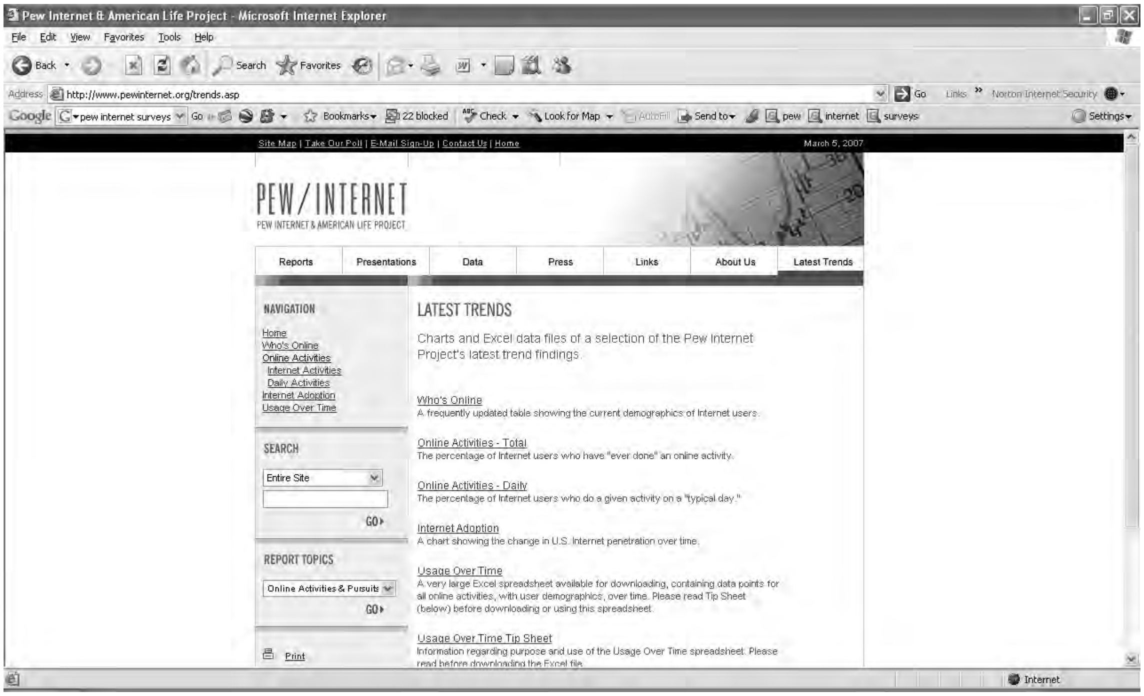
The above are all true stories. They illustrate two important points.

- The old saying that "ignorance is bliss" doesn't apply to networked computing. Many of these stories show that ignorant or naive computer users are vulnerable to exploitation and security breaches. The aim in creating the course was to help people become more informed, and therefore less vulnerable.
- The second point is that malware can have devastating effects on people. It can make them feel depressed, frightened, ashamed, and/or guilty. It can also (as in Hazel's case) cost them money and cause great frustration.

There is also some evidence that these "downsides" of networked computing are beginning to discourage people from benefiting from all the advantages of the technology. The Pew Internet Surveys provide a comprehensive ongoing survey of Americans' Internet use. The survey recently reported that:

- Twenty-five percent of America's e-mail users say they are using e-mail less because of spam. Within that group, most say that spam has reduced their overall use of e-mail in a big way.
- Malware has taken the shine off the Internet for many people, and that could be worrying if it became a long-term development, especially if you believe, as I do, that the benefits of the Internet greatly outweigh its disadvantages.

Figure 8.2 PEW/Internet Surveys



Pew Internet & American Life Project - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.pewinternet.org/trends.asp
Go Links Norton Internet Security
Google pew internet surveys Go Bookmarks 22 blocked Check Look for Map AutoFill Send to pew internet surveys Settings
Site Map | Take Our Poll | E-Mail Sign-Up | Contact Us | Home
March 5, 2007
PEW/INTERNET
PEW INTERNET & AMERICAN LIFE PROJECT
Reports Presentations Data Press Links About Us Latest Trends
NAVIGATION
Home
Who's Online
Online Activities
Internet Activities
Daily Activities
Internet Adoption
Usage Over Time
SEARCH
Entire Site
GO
REPORT TOPICS
Online Activities & Pursuits
GO
Print
Internet

LATEST TRENDS

Charts and Excel data files of a selection of the Pew Internet Project's latest trend findings

Who's Online
A frequently updated table showing the current demographics of Internet users.

Online Activities - Total
The percentage of Internet users who have "ever done" an online activity.

Online Activities - Daily
The percentage of Internet users who do a given activity on a "typical day."

Internet Adoption
A chart showing the change in U.S. Internet penetration over time.

Usage Over Time
A very large Excel spreadsheet available for downloading, containing data points for all online activities, with user demographics, over time. Please read Tip Sheet (below) before downloading or using this spreadsheet.

Usage Over Time Tip Sheet
Information regarding purpose and use of the Usage Over Time spreadsheet. Please read before downloading the Excel file.

Where Do You Come In?

When you come to consider your user training program, you may look at the examples and feel only the father is of interest to you, as he could easily be one of your employees. However, making the issues personal in relation to the father example can have the desired effect for the company. If the father learns that the company is going to show him how to protect his family, he will probably be interested in doing the training. He is then more likely to apply the same level of care and caution to his work environment. In some ways, what the employer is trying to achieve is difficult to achieve in the standard work/training environment, but by teaching users to protect themselves in the home, the company benefits from increased security awareness among its staff, and thereby increases its own level of protection.

I am not saying you should send all staff back to college. All companies have their own training strategy. You may use a variety of techniques such as Computer-based Training (CBT), Web-based training (WBT), and hybrid approaches. What we are asking you to consider is the approach you take to educate your staff.

At the beginning of this section we quoted Vesselin Bontchev's claim that user education doesn't work This point of view is very common among "old guard" antivirus researchers, and even in other sectors of the security industry. In the September 2006 issue of his "Cryptogram" newsletter, Bruce Schneier said: "I've met users, and they're not fluent in security. They might be fluent in spreadsheets, eBay, or sending jokes over e-mail, but they're not technologists, let alone security people. Of course they're making all sorts of security mistakes. I too have tried educating users, and I agree that it's largely futile."

Is it really the case that we have not been approaching this issue in the correct manner? Or should more companies try to supplement more technical countermeasures by engaging their staff more directly into the fight against malware?

Tools & Traps

What is Training For?

What are enterprise-training programs really intended for? In many cases, we would suggest that management has no more faith in the ability of end-users to "get" security than Bontchev or Schneier. At any rate, the bulk of the security budget tends to go to technical solutions, not direct security training.

It makes sense to provide training as a supplement to technical countermeasures. Certainly, none of the contributors to this book are likely to insist you have to rely either on technical solutions *or* on user training and education. Although, to quote David Harley, "No one knows if user education works, because no-one's ever done it properly yet." Still, there's no doubt that many employees do benefit from good IT security training, and their adherence to sound computer hygiene can raise security for other people and their systems, as well as their own. And in that sense, training in the operational use of security software and procedures, even if essentially rote learning, can have a significantly beneficial effect.

But training is also a risk transfer mechanism. If you're assumed to have been trained to act appropriately in a situation of exposure to risk, it's possible that if a breach occurs, it will be seen as your fault. The company's own responsibility for security breaches is perceived to be mitigated by the fact that they provided training. Of course, that may apply irrespective of how adequate that training is, if it exists at all. But risk transfer is an attractive (to the employer) feature of basic training, helping to compensate for the cost of providing it.

Education is a very different issue. In security, as in the school system, education is (or should be) about considerably more than carrying out procedures without real understanding. Not that it's necessary (or even, perhaps, desirable) to train the average employee up to expert standards. However, to acquire enough knowledge to think for his or her self, to extrapolate from one context to another and make a judgment about an uncertain situation is a very valuable trait, as long as it's backed up by appropriate and useful resources. "You're trying to influence attitudes and approaches, not create security experts." (Netfocus 2006 Workshop notes on User Education: David Emm and David Phillips).

In the early 1990s, most computer users were in the work place. Now many homes have at least one personal computer. In the 1990s, we created a strategy in the work place for staff training, but the landscape has changed. The threats are not just inside the company's perimeter; they can reach the workplace via an employee's home system. The nature of the threats to your company has changed, but has your training strategy kept pace with that change? If you want to prove the experts wrong about the unteachability of end users within the company, your enterprises strategy has to change to meet the new challenge: the home user.

Remember that a high percentage of home users are also computer users in the work place. Bad practices in the home will, in the absence of appropriate countermeasures, appear in the work place.

Overall points to consider when educating staff:

- Make it personal. End-users tend to learn more from examples they can relate to their own experience.
- Make it informative but fun.
- Keep it simple. If you turn all your staff into security gurus, they'll be too paranoid and beset by nightmare scenarios to function at full efficiency. They only need realistic information about what is out there posing a current threat, and an appreciation of the impact it may have not only on the company, but also on their own home-computing environment.
- Keep written policies and guidelines as short as possible. Simple and short. Staff are likelier to read and remember it. But make sure that there is more detailed

information available for occasions when it becomes necessary. Not too detailed, though. Otherwise, people will freeze when a scenario arises that doesn't map to the documentation.

- Make sure there is a clear line of communication when they need person-to-person support.
- Teach them how their machines are protected, but also how to verify that protective software is running. Do not just show them how it all relates to their work machine, but also give them information on how it relates to home systems. If they check their defenses at home, they will be likelier to check at work too, and report things that don't seem quite right.
- Encourage them to report anything strange appearing on their machine. This requirement should be reflected in your security policy. If you make it scary, they will not report anything. Above all, don't make the classic error of punishing them for reporting a problem. Even if the problem is their fault, would you rather they reported it or tried to conceal it? Encourage reporting with support, not a whip. There's certainly a place for the whip in security administration, but sometimes the carrot is a better first resort.
- Foster an atmosphere of openness among users. The last thing you want is to drive security breaches underground. So, encourage users to report alerts, suspicions, and problems and avoid making users feel stupid or culpable if something goes wrong or if they have made a mistake.

We are not claiming we can educate all users into a state of perfect security, but if the correct approach is taken, we can educate a high percentage of them. In the 1990s and the early years of the current decade, we have been trying to educate on the same premise that we did when the malware problem was novel. The baseline has moved outside the company's perimeter, and a company's educational strategy needs to adapt to meet that change. Otherwise, an uncomfortably high percentage of end-users will remain in a state of ignorance.

Security and Education in the UK

The last thing we want to do is frighten anyone off using the marvelous array of tools and resources made available by computers and networking, and we don't think all educators need to be security gurus. However, the Wild and Woolly Internet is not a place for the complete security tenderfoot, even in a well-protected working environment. Included here, therefore, are a few issues that deserve some thought from anyone who has to use computer and network technology in an educational context. Knowing about them may enable us all to work a little more safely. This section also examines the relationship between different agencies in maintaining security in education. Teachers with a minimal ICT background may find it helpful in understanding security as a whole school issue, but we anticipate (or at least hope) that anyone (certainly anyone with children of school age) will find it of use and interest. (Please forgive the inevitable UK bias in the detail here. The principles don't differ much across other jurisdictions.)

It would be nice if information security was taken care of behind the scenes (in education as in industry and other parts of the public sector), so that everyone else could get on with the real business of education. Unfortunately, any teacher using ICT needs background awareness of security issues, not only for their own safety on-line, but also as part of their pastoral duty of care towards students. Teachers of ICT, Business Studies, and so on, need a deeper level of awareness than average, and to be able to relate their knowledge to the curriculum.

Evaluating Security Advice

Finding security advice is easy. Just google "security" or "bot" or "rootkit" and watch those page hits roll. The problem, as with all the information available on the Web, is sifting fact from fiction. Much security information ranges from paranoia to wishful thinking. How do you evaluate the (often conflicting) advice from all those stakeholders in the security business? Vendors and consultants should know what they're talking about (removes tongue from cheek...) but their advice is colored by their field (and level) of expertise and the nature of their product range. Much vendor information is put there by marketing, not the research team, and they have a vested interest in selling you what they have, even if it's not what you need.

There is an enormous range of security resources available in the public sector, but the quality of the information is highly variable. Unfortunately, some of the best resources have restricted availability, and some of the publicly available material is so simplified as to be useless, if not actually dangerous. We've tried to address this problem by listing some resources (not all of it specific to education) in the resources section of this book. In an attempt to keep the book as up-to-date as possible at the time it hits the bookshelves, the resources section will be one of the last parts of the book to be finished, so excuse us if we aren't too specific in this chapter about what we're putting in there.

Information Sharing and the WARP factor

Of course, you are probably (hopefully, even) not going to rely on getting all your security information from this book. Information sharing between incident response teams (Computer Emergency Response Teams [CERTs] Computer Security Incident Response Teams [CSIRTs]) has benefited further education for many years. For instance, as JANET'S remit has spread from higher education to school education, so has the scope of its CERT evolved, and the need to trade information with other teams and sectors has increased.

NOTE

Here's the explanation for some of those acronym:

CERT: Computer Emergency Response Team (CERT is actually a trademark of CERT/CC, the CERT Coordination Centre (www.cert.org).

CSIRT: Computer Security Incident Response Team (www.cert.org/csirts/)

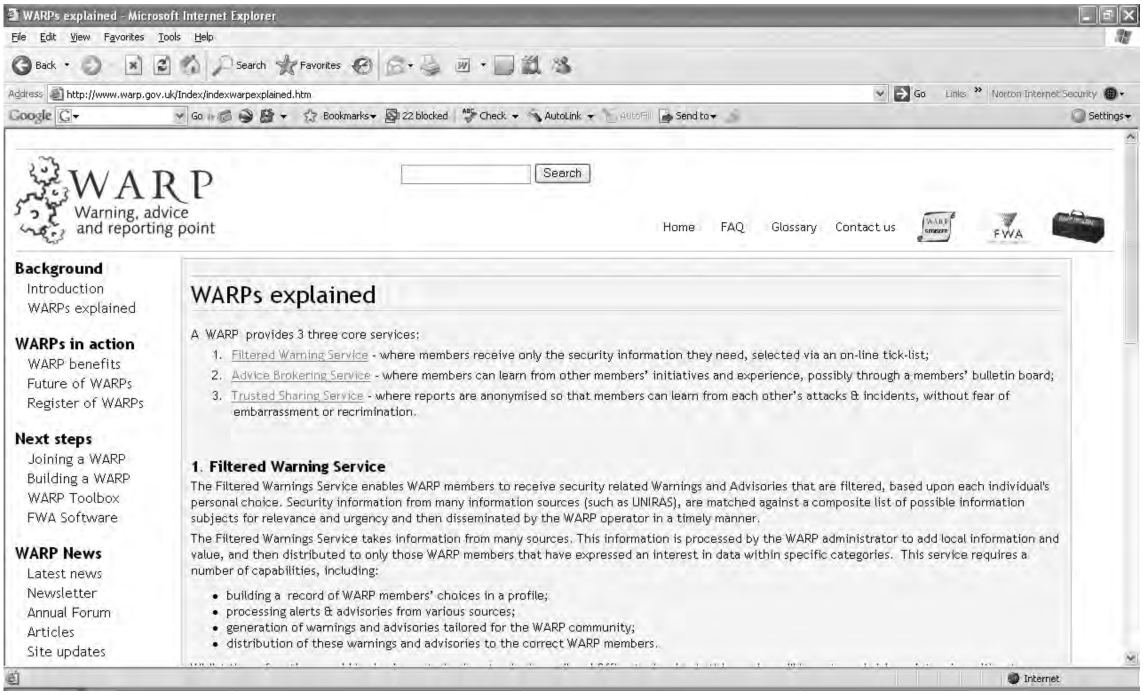
JANET-CERT (www.ja.net/cert/) is the security coordination team for JANET, the UK's Joint Academic NETwork (www.ja.net), which connects the UK's education and research organizations to each other and to the rest of the world.

Many universities have their own CERTs, and there are national CERTs such as AusCERT (www.auscert.org.au/).

More recently, the UK's National Infrastructure Security Co-Ordination Centre (NISCC), now the Centre for the Protection of National Infrastructure (CIPNI -www.cpni.gov.uk/ came up with the idea of scaled-down response teams called Warning, Advice and Reporting Points (WARPS) in sectors that couldn't afford large, full-time security management resources.

This type of resource is growing into an excellent means of sharing security information such as alerts about new threats and vulnerabilities, and obtaining and sharing advice and experiences. They also serve as initial reporting points for problems and incidents that the whole community can benefit from hearing about. Manpower and resources can be shared across schools, authorities, and regions, and information exchanged can be tailored so that it's relevant to the community. This is much healthier than having a junior technician spending hours a week drowning in irrelevant alerts and patch information, or ignoring the risks altogether. The process of implementing a registered WARP is somewhat bureaucratic, but the long-term benefits could be considerable. Much more information about WARPs is available at www.warp.gov.uk. The site includes a pointer to the first education WARPs to register, and access to the WARP toolbox, a cornucopia of documentation resources that will be of interest even to security minded schools and authorities that decide not to go for formal registration.

Figure 8.3 WARPs Explained


WARPs explained - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.warp.gov.uk/Index/indexwarpexplained.htm
Go Links Norton Internet Security
Google Go Bookmarks 22 blocked Check AutoLink AutoFill Send to Settings
WARP
Warning, advice and reporting point
Search
Home FAQ Glossary Contact us
FWA
Background
Introduction
WARPs explained
WARPs in action
WARP benefits
Future of WARPs
Register of WARPs
Next steps
Joining a WARP
Building a WARP
WARP Toolbox
FWA Software
WARP News
Latest news
Newsletter
Annual Forum
Articles
Site updates
Internet

WARPs explained

A WARP provides 3 three core services:

1. Filtered Warning Service - where members receive only the security information they need, selected via an on-line tick-list;
2. Advice Brokering Service - where members can learn from other members' initiatives and experience, possibly through a members' bulletin board;
3. Trusted Sharing Service - where reports are anonymised so that members can learn from each other's attacks & incidents, without fear of embarrassment or recrimination.

1. Filtered Warning Service

The Filtered Warnings Service enables WARP members to receive security related Warnings and Advisories that are filtered, based upon each individual's personal choice. Security information from many information sources (such as UNIRAS), are matched against a composite list of possible information subjects for relevance and urgency and then disseminated by the WARP operator in a timely manner.

The Filtered Warnings Service takes information from many sources. This information is processed by the WARP administrator to add local information and value, and then distributed to only those WARP members that have expressed an interest in data within specific categories. This service requires a number of capabilities, including:

- building a record of WARP members' choices in a profile;
- processing alerts & advisories from various sources;
- generation of warnings and advisories tailored for the WARP community;
- distribution of these warnings and advisories to the correct WARP members.

Tools & Traps

How To Be Warped

WARPs tend to follow this organizational pattern:

- The lynchpin of the group is an operator with some security knowledge who is well placed and equipped to communicate with the group members. Generally, these are expected to number up to 100 or so, in order to maintain a personal, community orientation. However, there is flexibility in that WARPs are often seen as interconnected organizations, so that a large community like the UK's National Health Service, for example, can be served by WARPs within WARPs. More typical WARP communities include small businesses, local government, special interest groups, and so on.

Continued

- The operator uses these community links to pass on warnings and advice to community members by a variety of routes (e-mail, text messages, RSS, and so on). This is primarily security information, but there's no reason why other information can't be exchanged as appropriate. Direct information and help and advice sharing between members are also encouraged.

WARPs have three main functions:

- Distribution of security warnings
- Response to appeals for help
- Provide a trusted mechanism for reporting problems and security breaches

They cannot usually provide the range of expertise and 24-hour coverage of a full-blown response team, but they can filter warnings of threats, vulnerabilities, patches, and so on, so that only those relevant to their community are passed on. Also, advice, information, and experience are of particular value when shared between sites with similar working environments and interests. Administration, staff, and resource costs can be reduced significantly when shared between sites on a part-time or volunteer basis.

An in-house team like a school's Information Technology (IT) support unit can do a similar (and sometimes very effective) job without affiliating or even being aware of the WARP initiative. However, extended networks of WARPs and other response teams can exchange information on a much larger scale with trusted sites, rather than having to choose between fragmented and contradictory information resources.

The Myth of Teenage Literacy

There is a myth of teenage Internet-literacy, and most parents and many teachers have been sucked into it. The entire younger generation, it seems, emerged from the womb already capable of programming a DVD recorder, and by the age of six already programs in C++ and assembly language and hacks into the CIA.

However, we have worked together for a while on pupil/teacher awareness of security issues. In fact, David Harley and Eddy Willems presented a paper based on our combined research into pupil awareness at the 2005 Virus Bulletin conference, which is available at www.smallblue-greenworld.co.uk, by kind permission of Virus Bulletin ("Teach Your Children Well: ICT Security and the Younger Generation": David Harley, Eddy Willems and Judith Harley). I'm afraid we're unconvinced that the younger generation is any better at practicing safe hex than it is at safe sex. The ability to use a search engine, e-mail, or instant messaging, or even to blog on MySpace, are not evidence of comprehensive computer knowledge, or of an ability to compare and evaluate google search results.

Tools & Traps

Security and Students

Our Virus Bulletin paper was sparked off by an article by Eddy Willems ("The End of Cybercrime?": Virus Bulletin, August 2004), in which he discussed some informal research in Belgian schools, including talking to a small group of children about the following group of questions.

- Do you know what a virus is?
- Have you ever been affected by a virus?
- What are the effects of a virus?
- Do you click on every link in an e-mail and open every attachment you receive?
- Do you use an easy password?
- How can you surf on the Internet safely?
- What do you think of virus writing or hacking?

The follow-up study expanded the range of questions and tasks.

- Name at least three types of "viruses" and give an example of each type.
- How do viruses get passed onto your computer?
- Why would someone send a virus across the Internet?
- What must individuals and firms always do to be prepared for the threat of viruses?
- What does the term "social engineering" mean?
- Name three types of Web scams

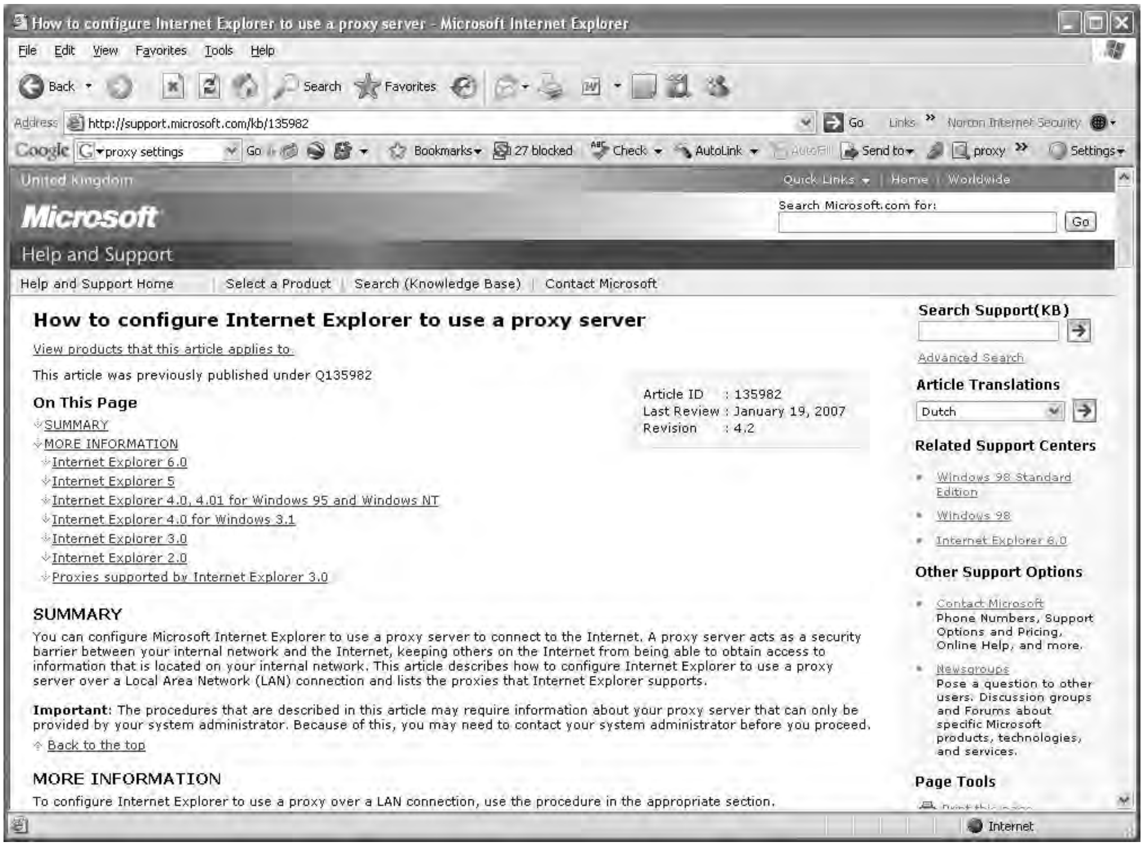
The students were also asked to create a newsletter to write about all of the above to be presented to new staff in a company. The rest of the study considered this material and workshop/Q&A presentations to and by classes about security-related issues, especially around malware and e-mail abuse.

Teaching Security in the Classroom

Teachers in the UK, as in many other parts of the world, have only limited control over what they teach, even in areas in which they have unusual expertise. In the study mentioned above, a brief informal survey of teaching materials current in the UK showed a wide range of accuracy, relevance, and understanding of the current threat landscape. Examples include such statements as, "For virus-scanning programs to be effective, their list of known viruses needs to be updated at least once every six months," and that "micro viruses are increasingly the most common form of virus." It's not unusual in any field for non-specialists to mistake outdated information for authoritative data. However, it remains all too possible to teach security using information that will score highly in an examination, but is so inaccurate that it makes the student more at risk in the real world.

Children and teenagers are often encouraged by their parents and others to regard themselves as more knowledgeable in the use of technology than they really are. Security vendors, the media, law enforcement, IT semi-literate civil servants, and so on, have talked up the expertise of the hacker and virus writing communities, and therefore increased the glamour of illicit activity. How knowledgeable children and teenagers really are depends on a number of factors: we suspect that for many young people, their expertise is focused on recreational and social use (and the researching of ways of bypassing school proxy servers).

Figure 8.4 The Proxy Server Problem


How to configure Internet Explorer to use a proxy server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://support.microsoft.com/kb/135982 Go Links Norton Internet Security
Google proxy settings Go Bookmarks 27 blocked Check AutoLink AutoFill Send to proxy Settings
United Kingdom
Quick Links | Home | Worldwide
Search Microsoft.com for: Go
Microsoft
Help and Support
Help and Support Home | Select a Product | Search (Knowledge Base) | Contact Microsoft

How to configure Internet Explorer to use a proxy server

View products that this article applies to.

This article was previously published under Q135982

Article ID	: 135982
Last Review	: January 19, 2007
Revision	: 4.2

On This Page

- SUMMARY
- MORE INFORMATION
  - Internet Explorer 6.0
  - Internet Explorer 5
  - Internet Explorer 4.0, 4.01 for Windows 95 and Windows NT
  - Internet Explorer 4.0 for Windows 3.1
  - Internet Explorer 3.0
  - Internet Explorer 2.0
  - Proxies supported by Internet Explorer 3.0

SUMMARY

You can configure Microsoft Internet Explorer to use a proxy server to connect to the Internet. A proxy server acts as a security barrier between your internal network and the Internet, keeping others on the Internet from being able to obtain access to information that is located on your internal network. This article describes how to configure Internet Explorer to use a proxy server over a Local Area Network (LAN) connection and lists the proxies that Internet Explorer supports.

Important: The procedures that are described in this article may require information about your proxy server that can only be provided by your system administrator. Because of this, you may need to contact your system administrator before you proceed.

Back to the top

MORE INFORMATION

To configure Internet Explorer to use a proxy over a LAN connection, use the procedure in the appropriate section.

Search Support(KB)

Advanced Search

Article Translations

Dutch

Related Support Centers

- Windows 98 Standard Edition
- Windows 98
- Internet Explorer 6.0

Other Support Options

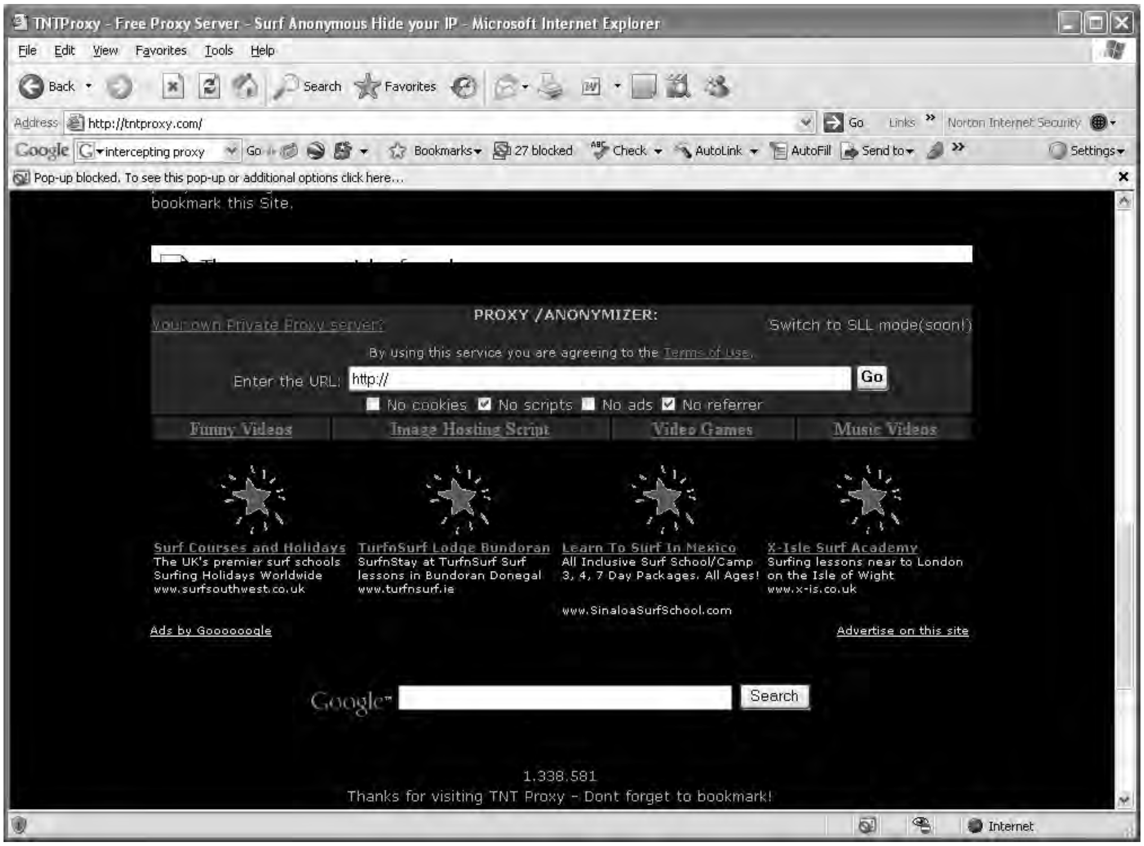
- Contact Microsoft
  Phone Numbers, Support Options and Pricing, Online Help, and more.
- Newsgroups
  Pose a question to other users. Discussion groups and Forums about specific Microsoft products, technologies, and services.

Page Tools

Note

Most Web browsers can be configured to use a proxy server. Many schools and businesses use this facility to filter Web requests so that students or employees are only allowed restricted (filtered) access to Web sites. However, there are many ways for even an unprivileged user to bypass such a mechanism and use a different proxy (e.g., an open proxy on a local robot (bot)-compromised PC, or one of the many proxy bypass sites to be found on the Web). Other possibilities include accessing sites that can't be accessed directly using copies cached by search engines, text translation services that pass requests as if from the remote IP, not the local IP, or using a shell account for text-based browsing.

Figure 8.5 Type 'n Go Using a Proxy Bypass Site


TNTProxy - Free Proxy Server - Surf Anonymous Hide your IP - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://tntproxy.com/
Go Links Norton Internet Security
Google intercepting proxy Go Bookmarks 27 blocked Check AutoLink AutoFill Send to Settings
Pop-up blocked. To see this pop-up or additional options click here...
Internet

bookmark this Site.

Your own Private Proxy server? PROXY /ANONYMIZER: Switch to SLL mode(soon!)

By using this service you are agreeing to the Terms of Use.

Enter the URL: http:// Go

☐ No cookies ☑ No scripts ☐ No ads ☑ No referrer

Funny Videos | Image Hosting Script | Video Games | Music Videos

Surf Courses and Holidays
The UK's premier surf schools
Surfing Holidays Worldwide
www.surfsouthwest.co.uk

TurfnSurf Lodge Bundoran
SurfnStay at TurfnSurf Surf lessons in Bundoran Donegal
www.turfnsurf.ie

Learn To Surf In Mexico
All Inclusive Surf School/Camp 3, 4, 7 Day Packages. All Ages!
www.SinaloaSurfSchool.com

X-Isle Surf Academy
Surfing lessons near to London on the Isle of Wight
www.x-is.co.uk

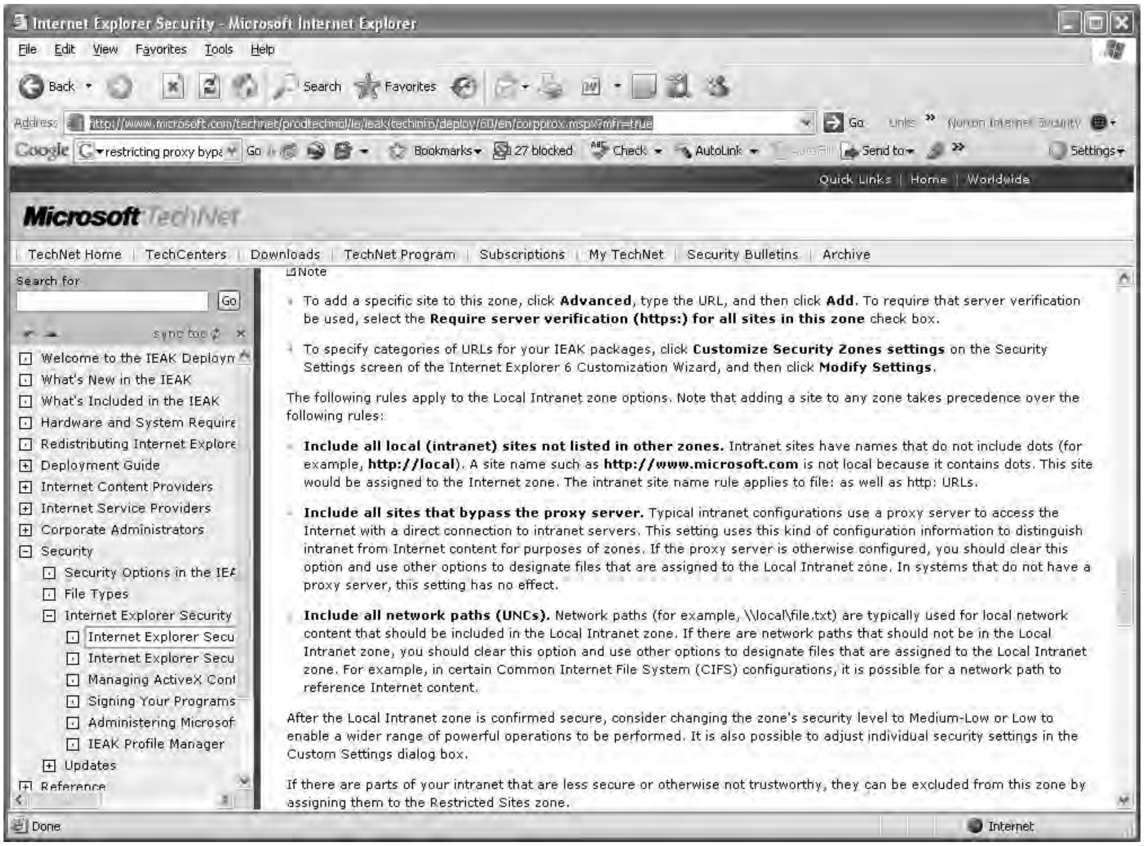
Ads by Goooooogle Advertise on this site

Google Search

1,338,581
Thanks for visiting TNT Proxy - Dont forget to bookmark!

We see less evidence of a widespread aptitude for the use of business applications, programming languages and so on, or most relevantly to the current topic, security, an area that we already know to be rich in misconceptions and misinformation. Sadly, bad news is good news. An article in the Times Educational Supplement ("Cat and Mouse Mischief" – March 2nd 2007) takes the "Junior hackers run rings round the security industry" approach ("Pupils are finding ever more ingenious ways to look at forbidden Web sites and wreak havoc on school computer systems.") and quoted a 17-year-old "computer expert who advises software companies on security" as saying that some pupils have the skills of "someone who has been working in the industry for three years or more".

Figure 8.6 You Can Filter Proxy Bypass Sites, But How Do You Keep Track of Them All?


Internet Explorer Security - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.microsoft.com/technet/prodtechnol/ie/ieak/techinfo/deploy/60/en/corpprox.mspx?mfr=true Go Links
Google restricting proxy bypa Go Bookmarks 27 blocked Check AutoLink Send to Settings
Quick Links | Home | Worldwide
Microsoft TechNet
TechNet Home | TechCenters | Downloads | TechNet Program | Subscriptions | My TechNet | Security Bulletins | Archive
Search for Go
sync toc
Welcome to the IEAK Deploym
What's New in the IEAK
What's Included in the IEAK
Hardware and System Require
Redistributing Internet Explore
Deployment Guide
Internet Content Providers
Internet Service Providers
Corporate Administrators
Security
Security Options in the IEA
File Types
Internet Explorer Security
Internet Explorer Secu
Internet Explorer Secu
Managing ActiveX Cont
Signing Your Programs
Administering Microsof
IEAK Profile Manager
Updates
Reference
Done
Internet

Note

- To add a specific site to this zone, click **Advanced**, type the URL, and then click **Add**. To require that server verification be used, select the **Require server verification (https:) for all sites in this zone** check box.
- To specify categories of URLs for your IEAK packages, click **Customize Security Zones settings** on the Security Settings screen of the Internet Explorer 6 Customization Wizard, and then click **Modify Settings**.

The following rules apply to the Local Intranet zone options. Note that adding a site to any zone takes precedence over the following rules:

- **Include all local (intranet) sites not listed in other zones.** Intranet sites have names that do not include dots (for example, **http://local**). A site name such as **http://www.microsoft.com** is not local because it contains dots. This site would be assigned to the Internet zone. The intranet site name rule applies to file: as well as http: URLs.
- **Include all sites that bypass the proxy server.** Typical intranet configurations use a proxy server to access the Internet with a direct connection to intranet servers. This setting uses this kind of configuration information to distinguish intranet from Internet content for purposes of zones. If the proxy server is otherwise configured, you should clear this option and use other options to designate files that are assigned to the Local Intranet zone. In systems that do not have a proxy server, this setting has no effect.
- **Include all network paths (UNCs).** Network paths (for example, \\local\file.txt) are typically used for local network content that should be included in the Local Intranet zone. If there are network paths that should not be in the Local Intranet zone, you should clear this option and use other options to designate files that are assigned to the Local Intranet zone. For example, in certain Common Internet File System (CIFS) configurations, it is possible for a network path to reference Internet content.

After the Local Intranet zone is confirmed secure, consider changing the zone's security level to Medium-Low or Low to enable a wider range of powerful operations to be performed. It is also possible to adjust individual security settings in the Custom Settings dialog box.

If there are parts of your intranet that are less secure or otherwise not trustworthy, they can be excluded from this zone by assigning them to the Restricted Sites zone.

NOTE

Here are some examples of those "ingenious ways" of bypassing school and office firewalls: [xx] denotes an URL hosting a proxy bypass server. (These are real but edited examples.)

(1) im tryin to find a Web site that will let all us at our school get on Web sites that have been blocked

we did have [xx] but some1 got it blocked

please either build a new one or send me some more to my hotmail adress above

pleaseeeeeeeeeeeeeeeee

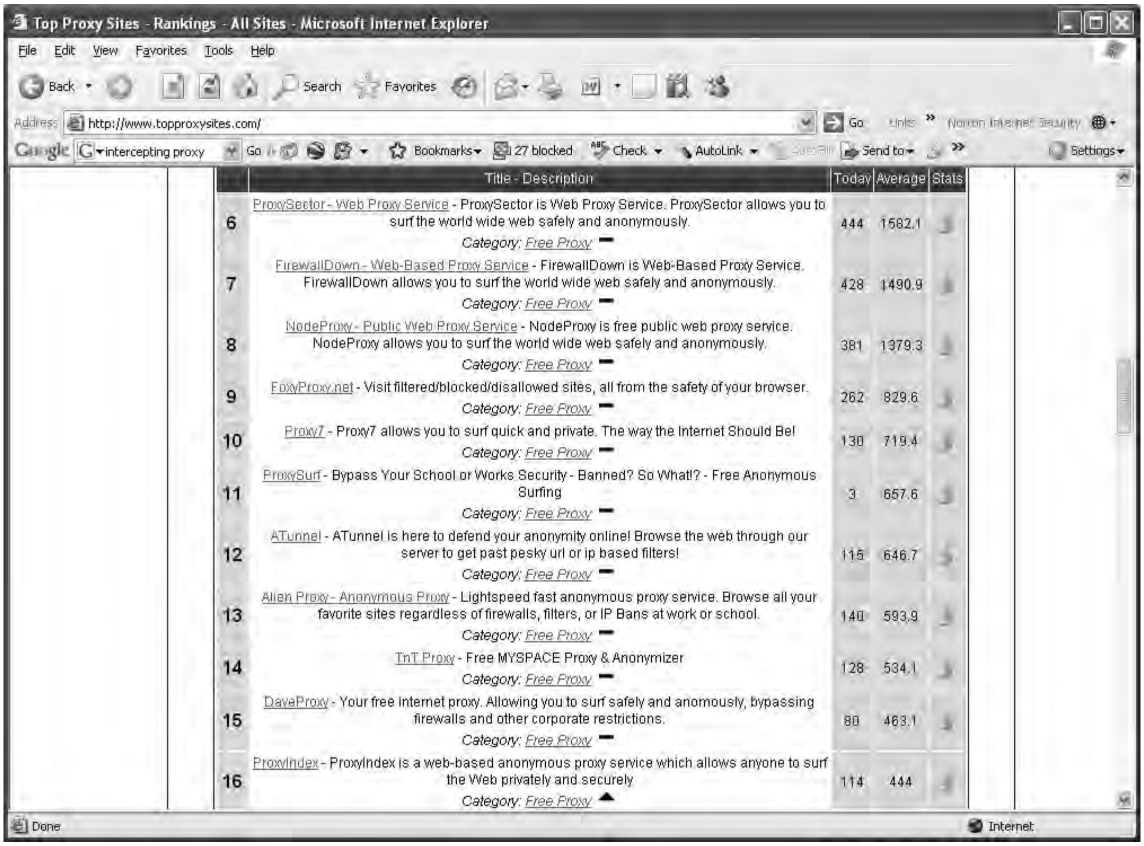
(2) how do I block a firewall?

(3) hello whats the latest Web site that you can use to hide your actions against Websence?

Clearly, schools are fertile breeding grounds for skiddies (script kiddies) and other wannabe hackers. How many of these achieve real expertise is largely a matter of conjecture and anecdote, rather than stringent analysis, though Secunus Software, (see Figure 8.8) a major player in UK education and other public sectors, were quoted in the same piece as estimating that around 4 percent of pupils deliberately misuse school systems.

If these bypass mechanisms (sometimes called Common Gateway Interface [CGI] proxies or circumventors) are such a problem, why aren't there more effective attempts to regulate them? At the very least, shouldn't we be seeing more panic? For instance, Wikipedia tells us that "The use of circumventors is usually safe with the exception that circumventor sites run by an untrusted third party can be run with hidden intentions, such as collecting personal information, and as a result users are typically advised against running personal data such as credit card numbers or passwords through a circumventor." In fact, there are many hypothetical attacks made possible by proxy avoidance. The fact that we don't see more of them is probably due to the fact that there are usually easy, less site-specific ways of accomplishing the same criminal ends, but that doesn't mean there won't be further explorations of this attack vector.

Figure 8.7 Anyone Who Can Google Can Find Proxy Sites!



Top Proxy Sites - Rankings - All Sites - Microsoft Internet Explorer

http://www.topproxysites.com/

	Title - Description	Today	Average	Stats
6	ProxySector - Web Proxy Service - ProxySector is Web Proxy Service. ProxySector allows you to surf the world wide web safely and anonymously. *Category: Free Proxy*	444	1582.1	
7	FirewallDown - Web-Based Proxy Service - FirewallDown is Web-Based Proxy Service. FirewallDown allows you to surf the world wide web safely and anonymously. *Category: Free Proxy*	428	1490.9	
8	NodeProxy - Public Web Proxy Service - NodeProxy is free public web proxy service. NodeProxy allows you to surf the world wide web safely and anonymously. *Category: Free Proxy*	381	1379.3	
9	FoxyProxy.net - Visit filtered/blocked/disallowed sites, all from the safety of your browser. *Category: Free Proxy*	262	829.6	
10	Proxy7 - Proxy7 allows you to surf quick and private. The way the Internet Should Be! *Category: Free Proxy*	130	719.4	
11	ProxySurf - Bypass Your School or Works Security - Banned? So What!? - Free Anonymous Surfing *Category: Free Proxy*	3	657.6	
12	ATunnel - ATunnel is here to defend your anonymity online! Browse the web through our server to get past pesky url or ip based filters! *Category: Free Proxy*	115	646.7	
13	Alien Proxy - Anonymous Proxy - Lightspeed fast anonymous proxy service. Browse all your favorite sites regardless of firewalls, filters, or IP Bans at work or school. *Category: Free Proxy*	140	593.9	
14	TnT Proxy - Free MYSPACE Proxy & Anonymizer *Category: Free Proxy*	128	534.1	
15	DaveProxy - Your free internet proxy. Allowing you to surf safely and anomously, bypassing firewalls and other corporate restrictions. *Category: Free Proxy*	80	463.1	
16	ProxyIndex - ProxyIndex is a web-based anonymous proxy service which allows anyone to surf the Web privately and securely *Category: Free Proxy*	114	444	

File Edit View Favorites Tools Help
Back Search Favorites
Address Go Links Norton Internet Security
Google intercepting proxy Go Bookmarks 27 blocked Check AutoLink Send to Settings
Done
Internet

Perhaps the one reason the world in general is so calm about the problem, is that even security workers in education aren't sufficiently aware of the problem, or believe that the use of blacklists for blocking proxy bypass sites and open proxies afford sufficient protection, especially when tunneling through HTTP over SSL/TLS, SSH (Secure Shell) and DNS (Domin Name Service.) The use of circumventors for anonymization is, in some contexts, a positive. And there are supplementary technologies such as the use of intercepting proxies, where connections through Network Address Translation (NAT) are intercepted and redirected to an authorized proxy irrespective of client-side configuration and intervention. Interceptors characteristically work by switching and routing Hypertext Transfer Protocol (HTTP) traffic through the local or approved remote proxy.

Duty of Care

In some respects, young people are still more vulnerable than adults, even adults who are not particularly computer literate. Security in schools aside, most home-use security issues apply to some extent to children as well as their parents, and in some contexts, they are far more vulnerable. They are, for instance, far likelier to visit specifically risky environments (e.g., chat rooms, instant messaging, music file exchange) as well as vulnerable to specific attacks from predatory pedophiles. Their inexperience may also make them more vulnerable to exploitation by criminals and vandals seeking access to sensitive data.

Of course, older people have, despite all those books about "The Internet for the Terminally Antique," a wide range of expertise and experience, from career IT professionals to accomplished silver surfers, to panicking descendents of Ned Ludd.

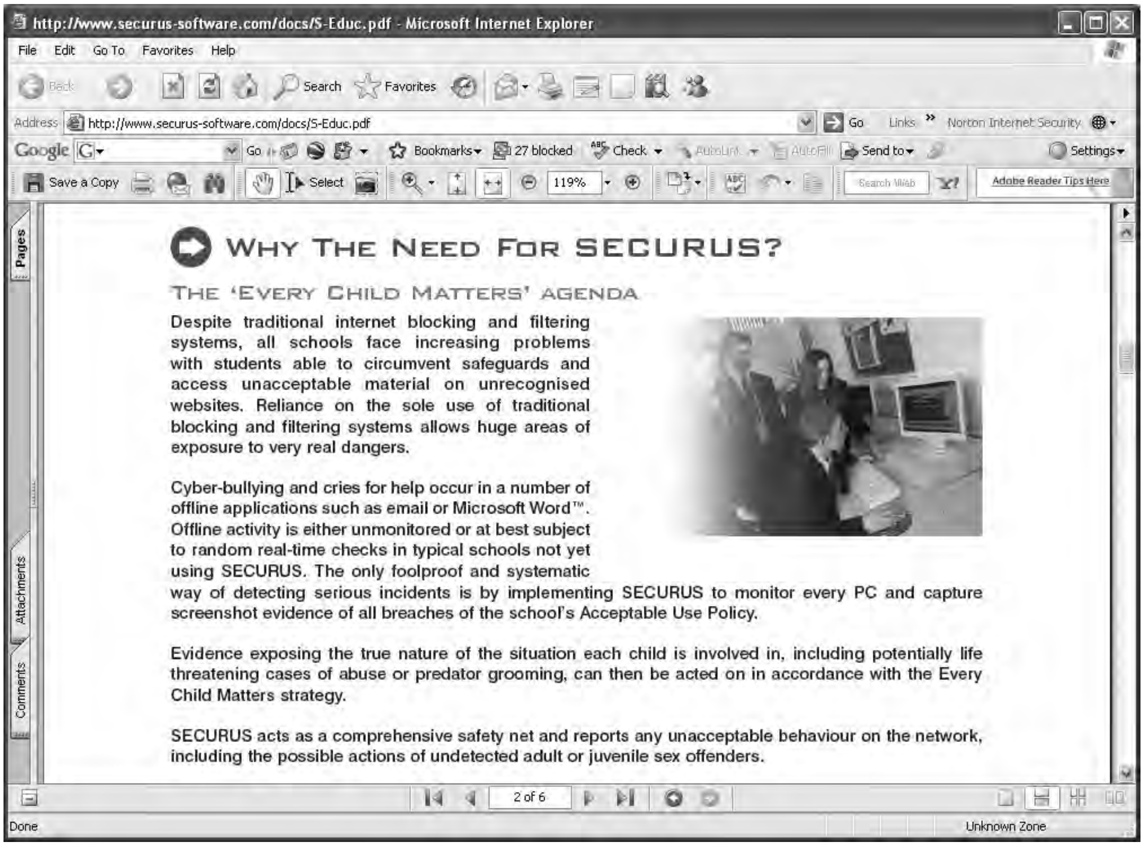
NOTE

Ned Ludd was the mythical leader of the Luddites, a movement of English textile workers in the early nineteenth century who protested against technological changes taking place as a result of the Industrial Revolution, for example by wrecking the machinery that they feared would rob them of their jobs.

Children may not have the moral or ethical grounding to address security issues or inappropriate behavior, because their elders haven't been able to communicate moral guidance in computing environments. In fact, older generations may not actually be better equipped to apply conventional morality to online transactions. However, technophobia might in itself have a "moral" influence. (These issues are discussed at some length in "Viruses Revealed" by David Harley, Robert Slade, and Urs Gattiker – Osborne, 2001.)

Nor should it be assumed that pre-adults don't need to know about criminal activity such as fraud, which is primarily aimed at adults. Some of the children in these studies were old enough to take part-time jobs and some were on the verge of leaving school. According to UK culture and law, they were two years from being an adult, but were certainly exposed to a wide variety of malicious software, phishing e-mail, and so forth. They might already have access to their parents' credit cards and have their own bank accounts and cash cards, though the facilities extended to this age group are usually limited.

Figure 8.8 Securus and "Every Child Matters"


http://www.securus-software.com/docs/S-Educ.pdf - Microsoft Internet Explorer
File Edit Go To Favorites Help
Back Search Favorites
Address http://www.securus-software.com/docs/S-Educ.pdf Go Links Norton Internet Security
Google Go Bookmarks 27 blocked Check AutoLink AutoFill Send to Settings
Save a Copy Select 119% Search Web Adobe Reader Tips Here
Pages Attachments Comments
2 of 6
Done Unknown Zone

WHY THE NEED FOR SECURUS?

THE 'EVERY CHILD MATTERS' AGENDA

Despite traditional internet blocking and filtering systems, all schools face increasing problems with students able to circumvent safeguards and access unacceptable material on unrecognised websites. Reliance on the sole use of traditional blocking and filtering systems allows huge areas of exposure to very real dangers.

Cyber-bullying and cries for help occur in a number of offline applications such as email or Microsoft Word™. Offline activity is either unmonitored or at best subject to random real-time checks in typical schools not yet using SECURUS. The only foolproof and systematic way of detecting serious incidents is by implementing SECURUS to monitor every PC and capture screenshot evidence of all breaches of the school's Acceptable Use Policy.

Evidence exposing the true nature of the situation each child is involved in, including potentially life threatening cases of abuse or predator grooming, can then be acted on in accordance with the Every Child Matters strategy.

SECURUS acts as a comprehensive safety net and reports any unacceptable behaviour on the network, including the possible actions of undetected adult or juvenile sex offenders.

NOTE

"Every Child Matters" is a UK government initiative concerned with ensuring the general welfare of children and young people from birth to 19 years of age. The aims of the project are children should be safe, healthy, enjoy and achieve, make a positive contribution, and achieve economic well-being (www.everychildmatters.gov.uk/

Surfing the Darkside Economy

Once upon a time, the Internet and its precursors constituted, except for the military bits, an academic's playground. Trust was taken for granted and to some extent we're still paying the price (and the security vendor subscriptions) for that presumption of pre-fall innocence.

Since the end of the 1980s, the take-up of the Internet by the rest of the world (and its transmutation into the biggest shopping mall in the solar system) has introduced bigger snakes into the Garden of Eden.

More recently, there were virus writers, malicious hackers, spammers, con artists, money-launderers, and so on. They didn't have a lot to do with each other, and many of us didn't have to bother with some or all of them. Nowadays, bandits on the information superhighway work on more-or-less commercial financial models. Phishing isn't just about counterfeit banking Web sites. It's about viruses and Trojans that open up the victims' systems for use to distribute spam and scam e-mails, or to store stolen information. It's about tricking computer users into allowing their bank accounts to be used for money laundering. (These mechanisms are considered more closely in other chapters.) However, if you have some understanding of how different kinds of cyber-villains work together, and can see the interdependencies, you will be better able to protect yourself and those with whom and for whom you work with.

We might even understand better when the IT support team says, "We can't let you do that: it's a security risk." But that might be too much to ask. There's a common conception that the IT team are the group best able to assess risk. Sometimes this is true, and sometimes it's far from true. The team probably doesn't have an in-house security guru (tell them and the higher echelons of educational administration about WARPs!). Sometimes, they may say yes or no because it's politically expedient, or they're panicked by media misinformation, or it's just less bother to give that answer. An understanding of how the dark-side economy works, and of security countermeasures, will put you ahead in that argument.

Duty of Care Issues (Again)

We do know educationalists and security gurus who are already torn between elation at the potential of the technology available, and despair at the amount of new things to learn. Some of them have probably already winced and skipped ahead to the next chapter. And, of course, you can engage with many of the tools presented in this book quite safely without knowing the difference between a rootkit and a bot, or a cracker and a mule driver. (The more so if you use platforms other than Windows, especially older and less secure flavors of Windows.) But schools and authorities have a duty of care to pupils and to staff, an obligation to protect them as best they can from the bad guys and from themselves.

That doesn't just mean investing time and capital into the best technical defenses you can afford and scrupulous patch updating. It means including security awareness across the curriculum, not just in ICT and Business Studies. In fact, most of the ICT textbooks we've seen are firmly locked into the early 1990s when they talk about security issues, to the extent that the security-savvy teacher may spend sleepless nights worrying about whether improving their knowledge might put pupils at a curricular disadvantage.

It also means teaching good citizenship: netiquette, netizenship, encouraging discussion about ethical issues as the curriculum gets more IT-intensive. A healthy dose of skepticism wouldn't hurt. And while requiring the entire school to sign up to acceptable use policies, codes of conduct, and so forth are at least a way of demonstrating due diligence, they can also be a valuable educational tool in their own right.

Cross-Curricular Security

Effective and secure use of ICT across the curriculum requires the maintenance of:

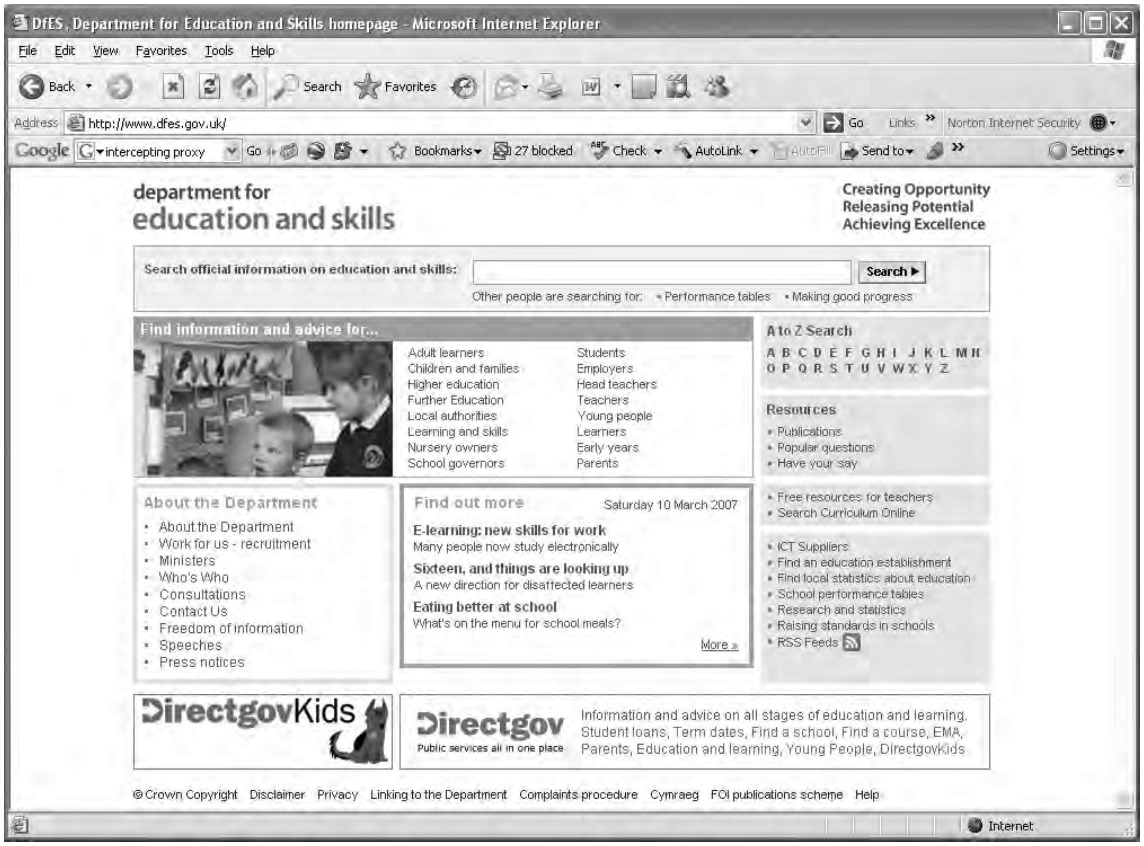
- Peer dialogue between specialist subject teachers where subject matter may cross over.
- Dialogue between senior management and subject specialists to maintain an overview and reduce the risk of multiple wheel variant reinvention.
- Agreement between the IT department, senior management, and teaching staff on responsibility and the accountability of departments, staff, management, and the student.
- A consistent system for applying sanctions against misuse of school systems and resources. For instance, it's not a good idea to sanction by withdrawal of access privileges if that withdrawal means that resources across the curriculum become inaccessible, unless there is agreement across teaching departments, IT support, and senior management. This may require not only agreement in principle, but also a mechanism for discussing individual cases.

One specific approach to pastoral care in the IT context is to integrate teaching online citizenship and awareness of security issues into a wider program of training, policy, and dissemination of information within the school and to parents (and other interested parties). Such an approach demonstrates the school's intent to apply "due diligence," to protect pupils from bad influences as well as from themselves, and to maintain good lines of communication with pupils and families.

This approach should also reflect the fact that on-line safety is not the sole responsibility of one group, whether it is parents and students, teaching staff, management, or IT staff. In the UK, there are many other stakeholders with a share in the responsibility:

- Local authorities and consortia
- The Joint Academic NETwork (JANET) supplying connectivity to schools as well as to Higher Education and Further Education establishments
- The British Educational Communications and Technology Agency (BECTA)
- Even Parliament. Indeed, in the UK, the government regulates education very tightly through the use of a National Curriculum, albeit filtered through the Department for Education and Skills (DfES).

Figure 8.9 DfES Web Site



DfES, Department for Education and Skills homepage - Microsoft Internet Explorer

Address http://www.dfes.gov.uk/

department for education and skills

Creating Opportunity
Releasing Potential
Achieving Excellence

Search official information on education and skills: Search

Other people are searching for: • Performance tables • Making good progress

Find information and advice for...

Adult learners	Students
Children and families	Employers
Higher education	Head teachers
Further Education	Teachers
Local authorities	Young people
Learning and skills	Learners
Nursery owners	Early years
School governors	Parents

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

Resources
- Publications
- Popular questions
- Have your say

- Free resources for teachers
- Search Curriculum Online

- ICT Suppliers
- Find an education establishment
- Find local statistics about education
- School performance tables
- Research and statistics
- Raising standards in schools
- RSS Feeds

About the Department
- About the Department
- Work for us - recruitment
- Ministers
- Who's Who
- Consultations
- Contact Us
- Freedom of information
- Speeches
- Press notices

Find out more Saturday 10 March 2007

E-learning: new skills for work
Many people now study electronically

Sixteen, and things are looking up
A new direction for disaffected learners

Eating better at school
What's on the menu for school meals?

More »

DirectgovKids

Directgov Public services all in one place

Information and advice on all stages of education and learning, Student loans, Term dates, Find a school, Find a course, EMA, Parents, Education and learning, Young People, DirectgovKids

© Crown Copyright Disclaimer Privacy Linking to the Department Complaints procedure Cymraeg FOI publications scheme Help

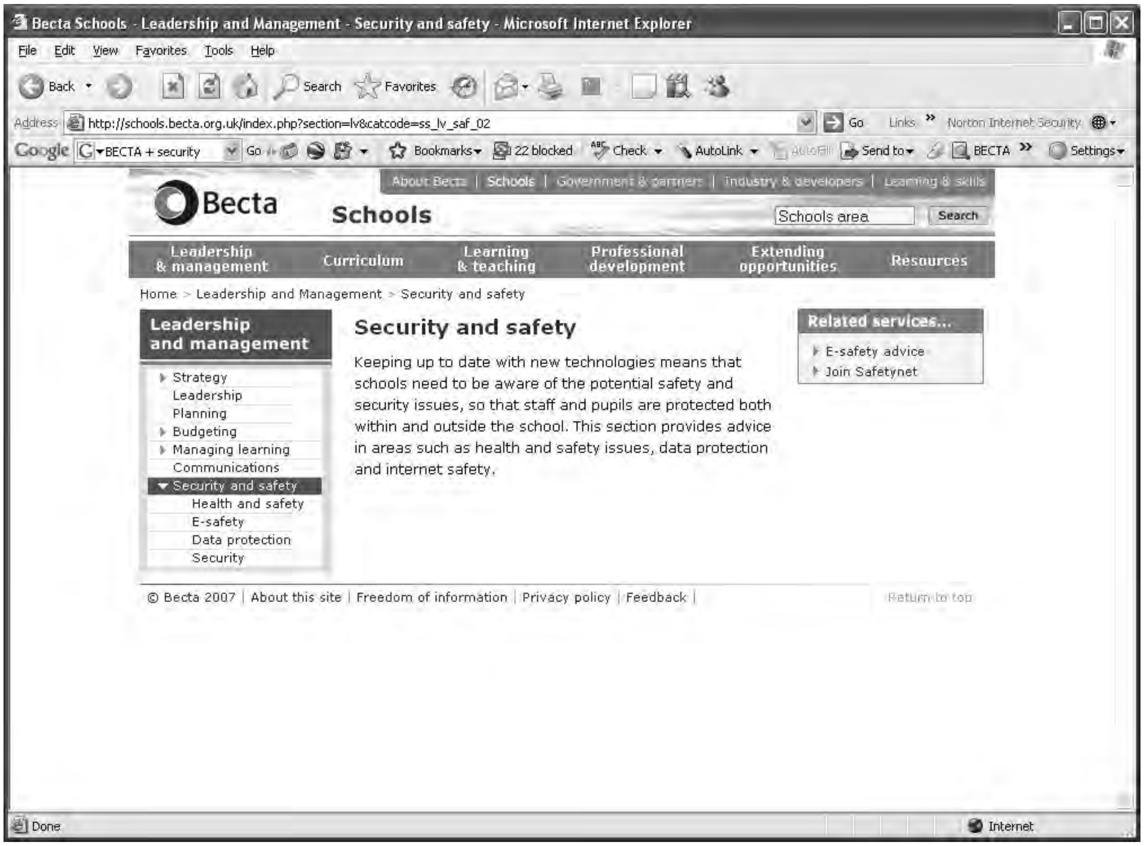
The names of the stakeholders may change, like their roles and responsibilities, according to which part of the world you live in. But the crucial point is that responsibility is shared, not devolved. Unfortunately, the legal policy and standards frameworks that support education security constitute a complex, many-branched tree. A well-conceived and integrated pastoral and outreach program can offer a realistic picture to participants. It needs to be simple enough for non-experts to follow, to know what is expected of them and what protective measures are in place, not only within the school, but also at network levels. These can include:

- Technical measures: (e.g., local authority firewalls, content filtering and Web caching, local measures against malicious software, and so on). While some of these technologies can, applied over-protectively, be a serious nuisance to the technophile educationalist, calling attention to the fact that they're there has two advantages:
    - They reassure parents and teachers that defensive measures are in place against the wild and woolly Internet

- They remind students that attempts to misuse the available resources (or indeed to subvert or bypass protective measures) are likely to have been anticipated.

- Policies and agreements: not only the sort of hoops that organizations must jump through to access remote resources (e.g., connectivity agreements, Acceptable Use Policies, security policies and so on), but also school and LEA policies.
- Legal issues such as copyright and intellectual property issues, data protection, computer misuse, child protection (again), abuses of e-mail, and so forth. Again, BECTA has a number of articles relating to these issues in the other European countries regulated by EC directives, which often have very similar legislation. Because much relevant legislation is state-specific rather than federal, we won't attempt to cover it here.

Figure 8.10 BECTA: The British Educational Communications and Technology Agency


Becta Schools - Leadership and Management - Security and safety - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://schools.becta.org.uk/index.php?section=lv&catcode=ss_lv_saf_02 Go Links Norton Internet Security
Google BECTA + security Go Bookmarks 22 blocked Check AutoLink AutoFill Send to BECTA Settings
About Becta | Schools | Government & partners | Industry & developers | Learning & skills
Becta Schools
Schools area Search
Leadership & management Curriculum Learning & teaching Professional development Extending opportunities Resources
Home > Leadership and Management > Security and safety
Leadership and management
Strategy
Leadership
Planning
Budgeting
Managing learning
Communications
Security and safety
Health and safety
E-safety
Data protection
Security
Related services...
E-safety advice
Join Safetynet
© Becta 2007 | About this site | Freedom of information | Privacy policy | Feedback |
Return to top
Done
Internet

Security and safety

Keeping up to date with new technologies means that schools need to be aware of the potential safety and security issues, so that staff and pupils are protected both within and outside the school. This section provides advice in areas such as health and safety issues, data protection and internet safety.

Technical Areas Checklist

We won't attempt to cover all the conformance issues that may apply, but here are some technical issues that you might want to run past your management and IT support teams, if they haven't already run them proactively by you.

There aren't usually any absolute rights or wrongs to these. There is a continuum between near-perfect security ("deny all," in security system administration parlance) and absolute but unsafe usability ("allow all"), and the institution has to find a comfortable place on that continuum. Sometimes it comes down to a compromise between safe systems that aren't of much practical use, and the need for experimentation and research. Striking that balance requires discrimination between risks that must be avoided and risks that can be accepted.

NOTE

Robert Slade has a couple of quotes that summarize this dichotomy neatly.

Jeff Richards' Laws of Data Security are (1) Don't buy a computer (2) If you do buy a computer, don't turn it on.

While a second quote, from John Parks, is to the effect that "A ship in a harbor is safe, but that is not what ships are built for."

Mail filtering. This remains a hit and miss technology. Opinion varies according to context as to whether it's better to risk getting the occasional scam, spam, or virus message rather than risk losing legitimate messages to an overzealous filter. Clearly, if recipients are well aware of the issues around phishing, 419s, e-mail worms, and so on, a laissez-faire approach is less risky. However, restrictions on outgoing e-mail in particular do reduce the risk of deliberate misuse of school resources, as well as reducing the knock-on effects of security breaches such as virus infection.

- It's difficult to argue against the need for Web access across the curriculum in the 21st Century. However, issues of duty of care, conformance, and so forth make it inevitable that access to dangerous areas be blocked where practical (e.g., pornography, scam sites, hacker and cracker resources). More contentiously, IT teams or third-party providers may have arbitrary blocking policies that conflict with the needs and aims of teaching staff. In situations like this, communication and negotiation are vital.
- Many organizations regard it as part of their duty of care to monitor possible misuse on the part of staff and students, using caching and usage monitoring, among other approaches. (This is probably not contentious in principle: if sites

don't do it, it's probably a resource issue rather than any sort of ethical reluctance.) The main issue here is interpreting data correctly: passively receiving inappropriate mail, getting unseemly pop-ups, and so forth, is not, in itself, necessarily an indication of improper conduct by the recipient. There have been a number of recent cases that may suggest otherwise, and we'll consider those shortly. There is also a "duty of care" to ensure that those being monitored are aware of the monitoring and, even more importantly, what activities are being checked for and what policies are in place.

- External mail accounts pose particular dangers, both as channels for external malicious activities, and as channels of deliberate misuse. Access to external e-mail can be restricted by such measures as firewall configuration, blocking known Webmail pages such as hotmail.com and gmail.com, as well as by limiting the user's ability to install or configure applications and services on the systems they use. These measures can interfere with legitimate activity, especially if there is no mechanism for allowing justifiable exceptions.
- Instant Messaging takes many forms, and is subject to many of the abuses that e-mail is prone to, often in an aggravated form. For instance, IM worms can spread at a speed far beyond that attainable by e-mail worms. In addition, Internet Relay Chat (IRC) is often used to control robot networks (botnets), networks of infected machines used to transmit malicious software, spam, hacking attacks, and so forth. Managing these risks adequately requires substantial knowledge, but IT teams need to be aware of them. It's certainly not recommended that schools allow the same unrestricted access that home users, especially teenagers, are used to.
- MySpace and similar sites are fun and a good way to get acquainted with the technology, but can attract predators. Staff members also need to be aware of the risks of blogging and other identifiable use of the Internet outside the school perimeters. I'm not aware of an instance where a teacher has been fired for criticizing their employer in a blog ("doocing"), but firewalls have ears.
- Skype and other consumer Voice over Internet Protocol (VoIP) applications can pose some security and confidentiality issues, and might sit uneasily with school policies. Like Webcams, access to telephony devices is best restricted to, or controlled by, teaching and IT staff. The ubiquitous Universal Serial Bus (USB) port brings other risks, too, such as the introduction of malware and the stealing of data through a multiplicity of electronic devices with data storage and transfer capabilities (pen drives, MPG players, smart phones and so forth).
- Remote access is best done through a Virtual Private Network (VPN) administered by the IT team. Remote Access applications over a minimally secured Web link can bring tears of rage to a security manager's eyes.

Finally, here's an absolutely secure approach to all internet access, courtesy of Marcus Ranum, who is sometimes described as the inventor of the firewall. (Used by kind permission.)

Figure 8.11 The Ultimate Firewall/IPS

NOTE

Marcus includes the following installation instructions on his Web page (www.ranum.com/security/computer_security/papers/a1-firewall/):

"For Internet use install the firewall between the demarc of the T1 to the Internet. Place the jaws of the firewall across the T1 line lead, and bear down firmly. When your Internet service provider's network operations center calls to inform you that they have lost connectivity to your site, the firewall is correctly installed."

Draconian IT support teams will love it.

Not Exactly a Case Study: The Julie Amero Affair

Julie Amero, substitute teacher, was convicted in Norwich, Connecticut in January 2007, of offenses related to "risk of injury to a minor, or impairing the morals of a child." The conviction relates to events that took place in October 2004, when the teaching machine in the classroom where she was taking over a seventh-grade language class showed pop-up pornographic images. Potentially, the charges for which she was convicted carry a maximum sentence of 40 years imprisonment.

This isn't a case study as such: I seem to be one of the few people in the security industry who *hasn't* offered expert opinion, in or out of court. It is a fascinating case, and if you want to know more about it, the first places to go might be the trial transcripts at http://www.norwichbulletin.com/apps/pbcs.dll/article?AID=/20070225/NEWS01/702250334 and the Amero defense blog at http://julieamer.blogspot.com/ or some of the comments on the issue from professionals in the field, posted at http://www.securityfocus.com/columnists/434 and http://sunbeltblog.blogspot.com/2007/01/is-this-miscarriage-of-justice.html.

Many in the security and forensics industry sectors have argued against the conviction in this particular case, based on arguments such as the assertion that the prosecution's case relied on inadequate forensic imaging and analysis and misleading expert testimony. I'm not going to attempt to evaluate the applicability of arguments made against the conviction or second-guess the results of the recently announced new trial. Instead, I simply want to reiterate that responsibility is shared between the teacher, the school, and the employing authority (i.e., school district, local education authority, and so on, according to where you live.)

Substitute teachers and other staff who supplement core teaching staff, such as learning support and other teaching assistants and auxiliaries, are rarely at the top of the list for in-service training, even where such training includes substantial security-related content.

NOTE

The terms "supply teacher" and "cover teachers" are used in the UK, though not always interchangeably. Supply teachers are qualified to teach; however, cover teachers aren't, necessarily: they may simply be brought in to "babysit" in the absence of a qualified teacher. Definitions are less rigid in the US, where local authorities may have much more autonomy.

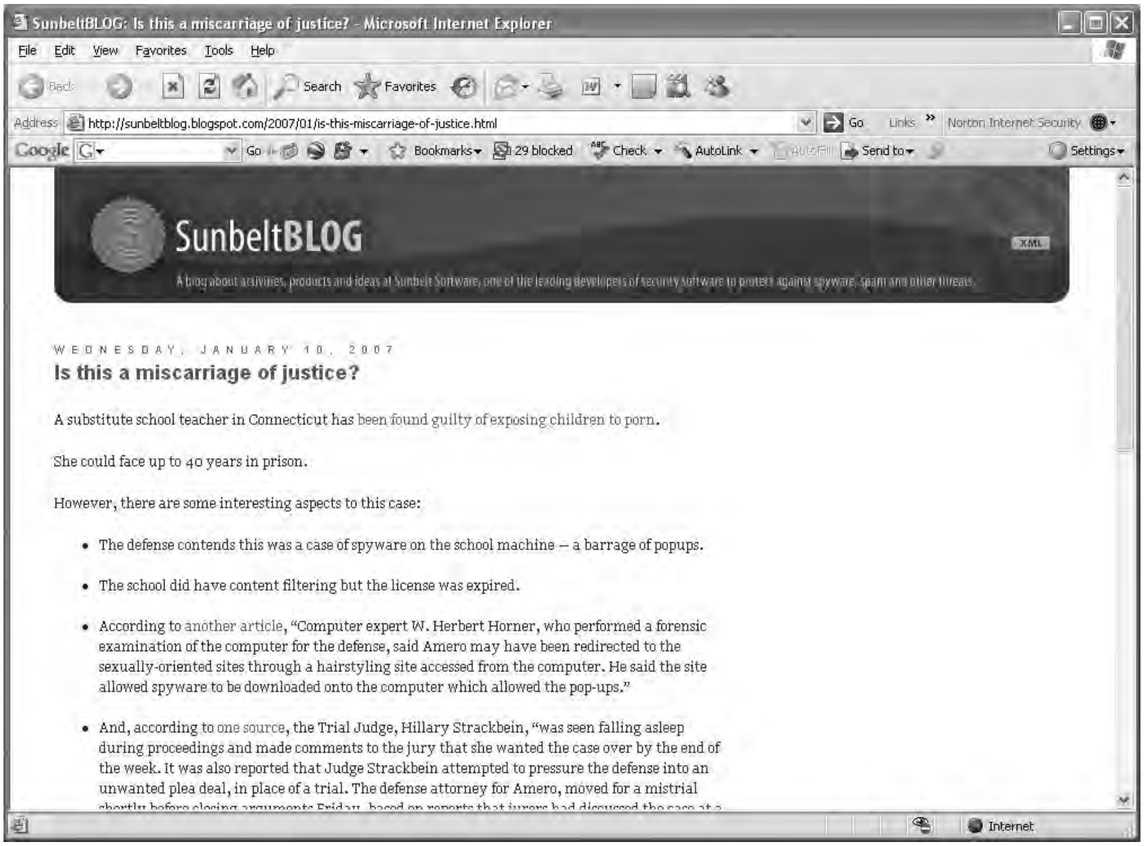
Even if they have generic training in how to deal with security problems in class, they may be reliant on whether the school's own procedures, protocols, and procedures are adequate and well documented, and even available to the substitute teacher. Such documentation should be supplementary to procedures defined in district documentation for substitute teachers. It doesn't seem unreasonable to expect a regular teacher to make sure that the substitute has suitable priming material available where practicable, in this as in other respects.

Furthermore, it has been reported that there were a number of technical issues that have a bearing on the issue of governance and responsibility:

- The school district's Web-filtering support contract had expired
- An obsolete antivirus program was installed, and there was no antispyware program
- Ms Amero tried to get help and support to get rid of the pop-up but wasn't able to get it, and her technical skills weren't up to dealing with it herself. In fact, it's been suggested that she was actually told not to switch off the PC.

Without going into the question of how much truth there is in these assertions, they do demonstrate clearly how reliant both pupils and staff are on competently implemented IT infrastructure, governance, and support.

Figure 8.12 Sunbelt Blog on the Amero Affair



SunbeltBLOG: Is this a miscarriage of justice? - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://sunbeltblog.blogspot.com/2007/01/is-this-miscarriage-of-justice.html Go Links Norton Internet Security
Google Go Bookmarks 29 blocked Check AutoLink AutoFill Send to Settings
SunbeltBLOG
XML
Internet

A blog about activities, products and ideas at Sunbelt Software, one of the leading developers of security software to protect against spyware, spam and other threats.

WEDNESDAY, JANUARY 10, 2007

Is this a miscarriage of justice?

A substitute school teacher in Connecticut has been found guilty of exposing children to porn.

She could face up to 40 years in prison.

However, there are some interesting aspects to this case:

- The defense contends this was a case of spyware on the school machine – a barrage of popups.
- The school did have content filtering but the license was expired.
- According to another article, "Computer expert W. Herbert Horner, who performed a forensic examination of the computer for the defense, said Amero may have been redirected to the sexually-oriented sites through a hairstyling site accessed from the computer. He said the site allowed spyware to be downloaded onto the computer which allowed the pop-ups."
- And, according to one source, the Trial Judge, Hillary Strackbein, "was seen falling asleep during proceedings and made comments to the jury that she wanted the case over by the end of the week. It was also reported that Judge Strackbein attempted to pressure the defense into an unwanted plea deal, in place of a trial. The defense attorney for Amero, moved for a mistrial shortly before closing arguments Friday, based on reports that jurors had discussed the case at a

Summary

It has long been held in some security circles that education doesn't work. In fact, this assertion is really based on a fundamental disagreement about what we can expect education to accomplish. It certainly hasn't solved all our security problems, but then there are many other problems it hasn't solved either. Criminal behavior, world poverty, armed conflict; they're all still out there. However, education and training, its less ambitious sibling, have certainly made a difference in many organizations, especially as part of a multi-layered protection strategy. Security awareness and good practice through a more rounded educational approach, rather than focusing entirely on operational training, is likely to make user education far more successful.

You might think that there'd be no difficulties in implementing an educational approach in education, but it's not always so. Fragmented responsibilities and uncertainty about which informational resources are reliable can create intense difficulties. However, community-oriented information and resource sharing initiatives like the WARP movement can make quite a difference.

Solutions Fast Track

User Education from an Educationalist's Perspective

☑ Security professionals take the tripartite or "CIA" (Confidentiality, Integrity, Availability) model as a starting point because it summarizes the essentials of security in a form that budget holders and even everyday users can understand (and sometimes even remember!) However, there is a widespread conviction in security circles that "user education doesn't work."

☑ There is a significant difference between training and education. Training is focused on operational issues. In the security arena, it tends to be concentrated on achieving "correct" behavior. Often, it's aimed mainly at clearing the decks so that technical and policy solutions have a reasonable chance of working properly. It doesn't necessarily require the trainee to fully understand the rationale and mechanisms that drive the instruction they receive. Education, in security as in life, is less deterministic. Done properly, it's less about trying to prescribe the correct action for every adverse circumstance, and more about developing the right "thinking" habits. When this happens, it's much likelier that when an adverse circumstance arises that isn't "in the manual," the individual will be able to extrapolate the most appropriate strategy for dealing with it.

- ☑ Education is not about turning out security experts, any more than art classes are about turning out great artists or religious studies are about turning out clerics. If people find a vocation in those areas, that's fine, but the primary benefit to society is that the general level of awareness is raised, and more people are able to recognize and practice good security, recognize and enjoy good art, and understand spiritual values.
- ☑ If educators can set their topic in a context that's recognizable and comparable to their personal circumstances, they will retain educational material better. If they learn to apply those lessons in their home environment, they're likelier to extrapolate to their work environment.

Security in Education

- ☑ All educationalists (not just specialists in computer-related technologies) need some understanding of security issues, not just to protect themselves, but to protect those for whom they have a duty of pastoral care. Specialists need deeper understanding, even when required to follow curricula that don't explicitly address security and safety issues.
- ☑ Security advice isn't hard to find. What is hard to do is to discriminate between unsound, anecdotal advice and "good" data. Community-oriented information sharing initiatives like WARPs can help to raise the general level of understanding, and have particular advantages in public and semi-public sector environments where resources as well as information can be shared between sites and organizations.
- ☑ Educational sites can have particular problems with students bypassing proxies for filtering undesirable sites by redirecting Web queries to other proxies. This is certainly a problem for many UK sites, and one that is neither well understood nor susceptible to a single simple solution.
- ☑ There is a tendency among adults to overestimate the technical sophistication of younger people in the context of ICT. There is certainly room for doubt as to whether previous generations have fully understood moral and ethical issues in anonline context, or learned to evaluate truth from deception in the absence of physical cues and clues, let alone managed to communicate a grasp of those issues to young people. In short, young people are probably still more vulnerable than their elders, if only because they tend to spend more time immersed in an online culture where, even if their technical smarts extend beyond Google and MSN, they lack the experience to make sound ethical judgments or assess the good will and good intentions of people with whom they communicate electronically.

- ☑ Effective and secure use of ICT across the curriculum requires the sharing of information and responsibility between many groups and at many organizational levels, from government and legislature, by way of the school district and the infrastructure providers, down to the teaching and support staff, parents, and the students themselves.

Not Exactly a Case Study: the Julie Amero Affair

- ☑ Irrespective of whether Julie Amero was rightfully convicted, the case does point to the need to share responsibility between all stakeholders, rather than risk scapegoating.
- ☑ Staff who are less likely to receive direct training and adequate proactive direction should at least be able to expect some protection against exposure to unnecessary personal and professional risk. It's not unreasonable to expect technical measures to be properly implemented, or support to be available when dealing with an unanticipated problem.

Frequently Asked Questions

Q: Isn't training part of education?

A: Of course. And while quasi-Victorian rote learning is out of fashion with educationalists and legislators in the UK and elsewhere at the moment, there has to be an element of mechanical drilling and memorization in many areas, as a means of teaching the basic skills that are the foundation for better understanding. Military organizations often understand this very well, expending considerable resources on teaching members on how to decide when reflexive obedience should give way to initiative and individual responsibility.

Q: Why hasn't user education worked better in security, then?

A: User education is a bit like socialism: we don't exactly know if it works, because no one's ever done it properly. For some people, fairly basic training works very well. After all, much security is just common sense dressed up with jargon. It's quite possible that many more people would benefit just as well, if the training they received was better resourced and/or better targeted. It's even possible that if people were better educated in the areas of ethics and social responsibility, that the online world would be a lot less like the Wild West.

Q: What are the essential components of a security policy?

A: We'll talk about that more in the governance chapter. However, commonly used definitions suggest that it should be a brief and high-level document that includes mission statement, goals, and objectives. It should be supported by standards, guidelines, baselines, and procedures.

Q: How do policies and standards fit in with training and education?

A: It makes sense for policies and related documentation to be part of the core training material. However, education, technological solutions, and policy are complementary components of a multi-layered strategy, not "Either...Or" alternatives.

Q: Shouldn't users be punished for infringing security rules?

A: By all means, if it's deliberate. But it makes sense not to discourage people from reporting problems, especially if they think they may have made a mistake.

Q: What are the advantages of registering as a WARP?

A: Initially, you benefit from the documented experience of previously registered organizations, and from the Filtered Warning Application made exclusively available to registered WARPS. In a wider sense, you benefit by affiliating with the wider CERT and WARP community, and can thus draw on a far wider range of experience and expertise.

Q: Wouldn't it have been better to include a summary of dependable resources?

A: There is such a summary in the resources section, later in the book. However, a printed resources section is always going to be out-of-date by the time it hits the bookshelves. There will therefore be regularly updated resources page at the Small Blue-Green World Web site. See www.smallblue-greenworld.co.uk/pages/avienguide.html for further details.

Q: How different are the teaching environments of the USA and the UK?

A: Services like healthcare and education are closely regulated in the UK by the government, although the exact degree of regulation and supervision can vary widely according to which party is currently in power. At present, the trend in state schools is towards very prescriptive curricula and performance evaluation criteria. Overall, state schools probably have less autonomy than their US equivalents.

Q: If young people are as technologically unsophisticated as you suggest, how come so many hackers and virus writers are kids?

A: Certainly there are technically accomplished teenage (and even younger) programmers, and not all of them are pure in heart and whiter than white. There are a great many others who know far less than they think they do, but then the same can be said of many adults! Stereotypical teenage virus writers are much less of a problem nowadays than the authors of malware written with criminal intent, but there's no doubt that they still exist, and there are many "script kiddy" attacks that show no particular sophistication or originality.

Q: Why shouldn't the IT department run a school's network?

A: Oh, they should and do (although sometimes they oversee an infrastructure configured and largely administered by a third-party provider.) However, they aren't necessarily the best-qualified group to specify the details of how ICT is used by teaching staff and pupils. That needs copious input from teaching staff, senior management, and so on. More to the point as regards the content of this chapter, they aren't necessarily fully conversant with security issues, let alone the professional, legal, and administrative pressures under which other staff have to work.

Chapter 9

DIY Malware Analysis

Solutions in this chapter:

- Anti-Malware Tools of the Trade 101
- The Basics – Identifying a Malicious File
- Process and Network Service Detection Tools
- Web-based Inspection and Virus Analysis Tools
- Using Packet Analyzers to Gather Information
- Examining Your Malware Sample with Executable Inspection Tools
- Advanced Analysis and Forensics
- Advanced Malware Analysis
- Static (Code) Analysis
- Dynamic (Behavior) Analysis

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

It's 3:00 on a Friday afternoon and your phone rings; it's your Security Operations Center (SOC). They tell you that they see an extraordinary amount of traffic going out to the Internet on port 443 from one particular Internet Protocol (IP) address on the network, which belongs to a machine in Santa Clara, California. Your job is to check out this machine to see what is causing this high traffic volume. You don't have physical access to this PC as you're in Boston, Massachusetts. How do you analyze this machine remotely? What tools do you use to accomplish this task? After reading Michael Blanchard's introductory section on "Anti-malware Tools of the Trade 101," you will know the basics of how to scan a machine for the presence of malicious software (malware), what to do once it's identified, and how to perform basic analysis on the malicious program itself, using a variety of tools.

Anti-malware software is at a pitch of sophistication that would never have been envisaged a few years ago. Unfortunately, it's never faced the volume and complexity of threats that it does now, and it isn't practical to rely purely on "signature"-based detection. Even where more proactive, generic measures are in place, system administrators in many organizations need to have an understanding of forensics and analysis in order to do their jobs as well as possible.

The sheer volume of new variants of bots and other malicious software associated with today's explosion of spam puts a severe strain on the ability of antivirus (AV) vendors to deploy timely reactive and proactive countermeasures. This in turn puts a strain on in-house security administrators, who often have to analyze code themselves, in order to maintain services during the "window of opportunity" exploited by malware variants that antimalware programs don't yet detect. In the second section of the chapter, Bojan Zdrnja shares some of his considerable forensic experience, looking at tools and techniques for advanced analysis, both static and dynamic.

Anti-Malware Tools of the Trade 101

It is not our intention in this chapter to explore every feature of the tools listed. Primarily, we'll focus on the use of the tools mentioned to help with the identification of malware on a host machine. We will also discuss how to determine exactly what the malware does, in order to take the necessary steps to isolate infected machines. This chapter will outline the methods that we've found to be the easiest way to start looking for malware on machines. In many cases the tools we'll be discussing can be used in many different ways. The examples that we use are by no means the only possible usage, just the ways that we've found to work very well both for the beginner and for the advanced analyst.

First steps should always be to have the SOC shut off Internet access for the suspect machine using firewall rules or router Access Control Lists (ACL's) and allow it to only communicate with the IP address from which you will be performing the analysis.

This lessens any risk to the company's network and other systems, while maintaining your access to the suspect machine for analysis. If this type of a lock down were not possible in a particular environment, physical access to the suspect machine would be required. Leaving the machine connected to a live network until the analysis is complete may also be an option, but a very risky one.

The Basics: Identifying a Malicious File

There are a number of tools that can be run on the remote system to aid you in determining what process or file is causing trouble on the machine. PsExec, part of the PSTOOLS package, is available from www.sysinternals.com (now owned by Microsoft), and is an almost essential tool for enabling remote analysis. Most remote tools available today provide full and transparent Graphical User Interface (GUI) access. PsExec allows you a slightly stealthy connection to a remote machine, however. Provided that you can offer the proper credentials to access the remote machine, using PsExec to open a remote command shell, the user of the remote machine may not even realize that you're connected to it, unless he or she is actively checking connections.

PsExec is the single most useful tool that can be added to any analyst's jump kit (for Europeans and others who may not be familiar with this term, it usually denotes a comprehensive first-aid kit of the type carried by paramedics and others). It will allow you to execute programs on a remote computer to which you have access. Most importantly, it allows you to launch a remote console and execute the needed tools on the remote machine, without needing to use a full graphical controller such as XP Remote Desktop (www.microsoft.com/windowsxp/using/mobility/getstarted/remoteintro.mspx), Virtual Network Computing (VNC), and so on.

Tools & Traps

VNC

This is a platform-independent desktop sharing system, originally developed at AT&T, for controlling a remote machine over a network. The original source code is open source, as are many recent implementations. It works on a client/server model, where the server is the program on the controlled machine, and the controlling machine connects to it via the client or viewer. The fact that the viewer and server don't have to use the same operating system makes it potentially useful in mixed environments, and its display capabilities make it more versatile than ye olde worlde Telnet (and not quite as insecure).

Continued

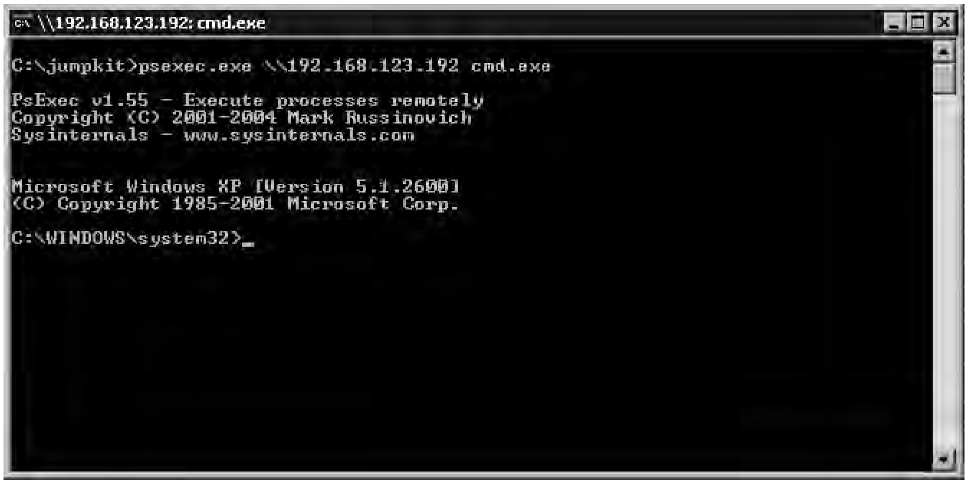
However, in today's insecure world, it needs to be boosted security-wise. Tunneling over Secure Shell (SSH) or Virtual Private Network (VPN) is an option, and there is an open-source plug-in for UltraVNC. RealVNC offer a commercial implementation that includes "Integrated Session Security," though we haven't tested this.

- http://ultravnc.sourceforge.net/
- www.realvnc.com/
- www.realvnc.com/products/enterprise/

PsExec.exe has many useful features, but for this example we will only use it to open up a remote command shell on the system in question. This is very easy to do by using a quick command line string on your system. You will only need to know the remote machine's IP address or Fully Qualified Domain Name (FQDN). Figure 9.1 shows an example of opening up a remote command shell.

The example shown in Figure 9.1 assumes that your current logged-in session has command line-level access to the remote machine. This can be either via native local permissions on the remote machine, Active Directory, other NT domain permissions, or via a mapped resource using other credentials that can be passed through using PsExec.

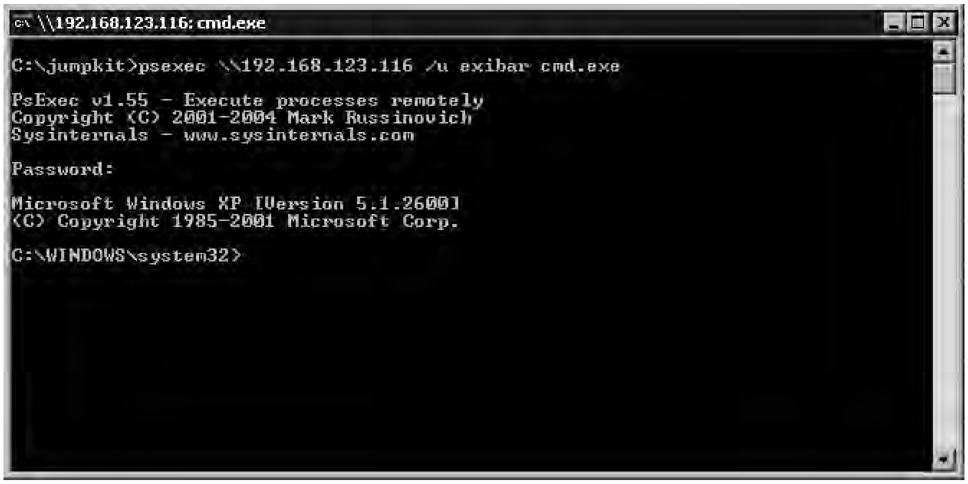
Figure 9.1 Opening a Remote Command Shell Using *psexec.exe*



```
\\192.168.123.192: cmd.exe

C:\jumpkit>psexec.exe \\192.168.123.192 cmd.exe

PsExec v1.55 - Execute processes remotely
Copyright (C) 2001-2004 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>_
```

If your account doesn't already enable you to get access to the remote machine, but you are aware of an account that does, you can use the example shown in Figure 9.2. Figure 9.2 shows the use of one of the many command line switches available with PsExec. The username "*/u*" switch and the password "*/p*" switch, (which is optional – if it isn't used you'll be prompted for the password), show how to pass user-level credentials along with the shell request all within the same command line.

Figure 9.2 Using the "/u" switch to Connect Using Different Credentials



```
C:\jumpkit>psexec \\192.168.123.116 /u exibar cmd.exe

PsExec v1.55 - Execute processes remotely
Copyright (C) 2001-2004 Mark Russinovich
Sysinternals - www.sysinternals.com

Password:

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

Here is an example command line showing how to open a remote shell using PsExec:

```
PSEXEC \\192.168.1.100 cmd.exe
```

Here is a summary of the syntax and command line switches for PsExec:

```
psexec [\\computer[,computer[,..] | @file ][-u user [-p psswd]]
[-n s][-l] [-s|-e][-i][-x][-c [-f|-v]][-d][-w directory]
[-<priority>][-a n,n,...] cmd [arguments]
```

If you omit the computer name, PsExec runs the application on the local system, and if you enter a computer name of "\\*" PsExec runs the applications on all computers in the current domain.

- ***@file*** Run the command on each computer listed in the text file specified.
- ***-a*** Run the application on the processor's specified, separated commas where 1 is the lowest numbered Central Processing Unit (CPU). Thus, to run the application on CPU 2 and CPU 4, enter: "*-a 2,4*"
- ***-c*** Copy a specified program to the remote system to execute. If this option is omitted, the application has to be in the execution path on the remote system.
- ***-d*** Don't wait for application to terminate; for use with non-interactive applications.
- ***-e*** Loads the profile for the specified account.

- ***-f*** Copy program to the remote system even if the file already exists there.
- ***-I*** Run the program to interact with the desktop on the remote system.
- ***-l*** Run process as an unprivileged user.
- ***-n*** Specify timeout in seconds for connection to remote computers.
- ***-p*** Specify an optional password for this user name. If this is omitted, you are prompted to enter a hidden password.
- ***-s*** Run remote process as the System account.
- ***-u*** Specify an optional user name for a remote login.
- ***-v*** Copy the specified file to the remote system only if it has a higher version number or is dated more recently.
- ***-w*** Set working directory of the specified process on the remote computer.
- ***-x*** Display user interface on the Winlogon desktop on the local system only.
- ***-priority*** Specifies priority level as -low, -belownormal, -abovenormal, -high or -realtime
- ***Program*** Specify name of program to execute here.
- ***Arguments*** Arguments to pass (file paths must be absolute paths on the target system).

NOTE

For more information on PsExec see the help file and www.microsoft.com/technet/sysinternals/Security/PsExec.mspx. You may also find Mark Russinovich's article on advanced usage at http://www.windowsitpro.com/Windows/Article/ArticleID/42919/42919.html interesting and useful.

Once you've successfully opened up a remote command shell, you'll have full access to that machine and will be able to run other tools that will enable you to determine exactly what else is running on that machine. More importantly, you'll be able to identify malicious files. Not only does this enable you to take remedial actions, but also indicates which files to copy for further sample analysis, forwarding to vendors, and so on.

Tools & Traps

The PSTools Utility Suite

We strongly urge you to download the entire PSTOOLS suite of tools rather than just PsExec. Even though we've only discussed the usage of one of the tools available in the suite, there are many other tools contained in the download package. You are sure to find that many of them come in handy from time to time. The tools are compact and free to use.

The suite was put together by Mark Russinovich, and the tools it currently includes are:

- **PsExec** A tool for executing remote processes
- **PsFile** A tool for showing files opened remotely
- **PsGetSid** Displays a user's or computer's Security Identifier (SID)
- **PsInfo** A tool for displaying system information, including how long the system has been up since reboot
- **PsKill** A tool that kills processes, identified either by name or by process ID
- **PsList** A tool to display detailed information about processes
- **PsLoggedOn** A tool to show who is logged on locally and via resource shares
- **PsLogList** A tool for dumping event log records
- **PsPasswd** A tool for account password management
- **PsService** A tool for remote services management
- **PsShutdown** A tool for shutting down and rebooting a remote system
- **PsSuspend** A tool for suspending processes

While the PsTools don't, of course, contain malware, their code and functionality has been misused by malicious programs, which may result in their being flagged by AV software as malware. You therefore need to be sure that you can distinguish a false positive in this scenario from the presence of real malware.

Now you have a remote shell open on a machine where you suspect that malware is running. Where do you go from here? The first thing that we usually do is create a directory on the root of the C:\ drive. This directory will serve as your repository for all the data that you gather using our emergency toolset, or "jumpkit tools." You need to keep in mind that any tool that you wish to run on the remote system must exist on the remote machine before it can be executed. Even though some of the tools can be run over the network, we prefer to keep the network

Continued

connections to a minimum while analyzing a machine for malware. However, in scenarios where forensics may be required with a view to future judicial review, this may not be an option. The section on advanced analysis and forensics later in this chapter will clarify this issue.

Once the repository directory is created, you can then copy all of your tools to this directory, and run them out of this directory when the time comes. The results can be saved in this directory as well, and when you've collected all the data you can, the directory can be moved to your main machine as a whole, without having to find the results files from all the tools.

One valuable tool that is native to every Windows installation is Netstat, which is an application that will display NETwork STATistics for the current machine. In other words, it will display all network connections to and from the machine. Let's look at how to use Netstat to help us determine what network ports are open on the system.

Here's a summary of the command line syntax for Netstat:

```
NETSTAT [-a] [-b] [-e] [-n] [-o] [-p proto] [-r] [-s] [-v] [interval]
```

And here's a summary of what those switches do.

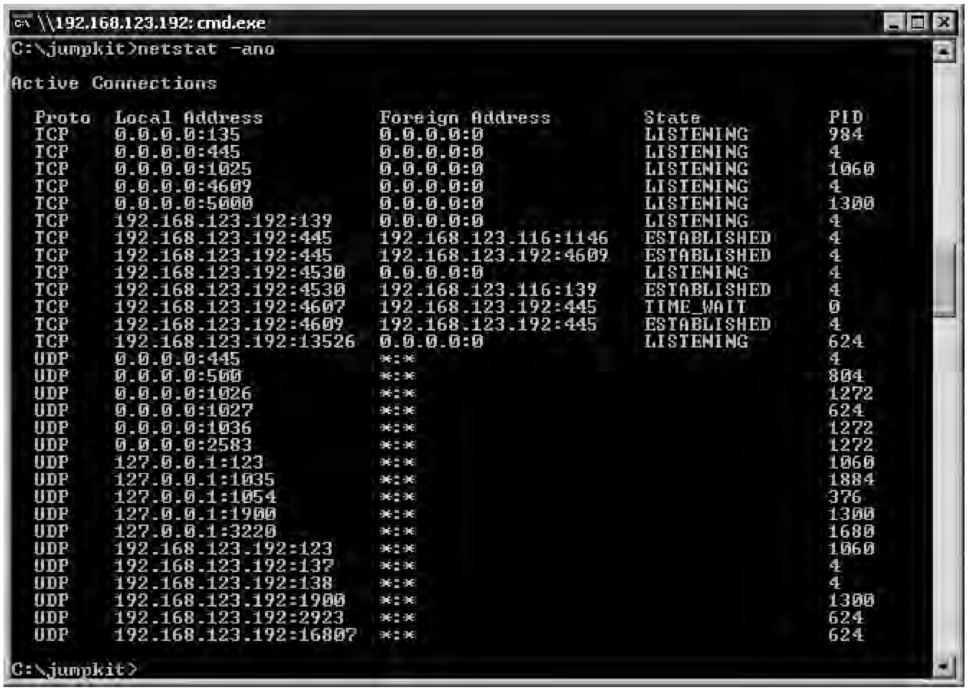
- ***–a*** Displays all active Transmission Control Protocol (TCP) connections, the TCP and User Datagram Protocol (UDP) ports on which the computer is listening.
- ***–b*** Displays the name of the program's name responsible for each connection or listening port (SP2 or higher).
- ***–e*** Displays Ethernet statistics (e.g., number of bytes and packets sent/received). This parameter can be combined with -s.
- ***–n*** Displays active TCP connections expressed numerically.
- ***–o*** Displays active TCP connections with the process ID (PID) for each connection. The Processes tab in Windows Task Manager shows the application associated with the PID. You can combine this parameter with *–a*, *–n*, and *–p*. (Windows XP or later)
- ***–p*** [Protocol] Shows connections for the [Protocol], which can TCP, UDP, TCPV6, OR UDPV6. Use with *–s* to display statistics by protocol, which can then be TCP, UDP, Internet Control Message Protocol (ICMP), IP, TCPV6, UDPV6, ICMPV6, OR IPV6.
- ***–r*** Display the [IP routing table]. (Equivalent to the route print command.)
- ***–s*** Displays statistics by protocol. (See "*–p*" parameter.) Statistics are shown for the TCP, UDP, ICMP, and IP protocols, by default. If the IPv6 protocol is installed (Windows XP), statistics for TCP over IPv6 and UDP over IPv6, ICMPv6, and IPv6 are shown.
- ***–v*** In conjunction with *–b*, displays the sequence of components that create a connection or the listening port for all executables.

- Interval Refresh the selected information display every [Interval] seconds. Ctrl+C stops the redisplay.
- `/?` Displays help text.

TIP

You'll find that if you simply run most of these tools, their output is larger than the command window buffer. For this reason, when running any tool from the command line, append a > *filename.ext* to the command (example: *netstat –a >report.txt*). This will send all of the output to a text file instead of to the screen. In the example shown, the output will be placed in a file named *report.txt*. This also has the huge benefit of saving the tool's output for review at a later time. This can also be a fully qualified path so it isn't limited to the current directory. We would recommend putting the full path to your repository directory if you're not running the command from within your repository directory. In other words, use the full (absolute) pathname from the root directory downward.

Figure 9.3 Typical Output of the Netstat Command



```
\\192.168.123.192: cmd.exe
C:\jumpkit>netstat -ano

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       984
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:1025           0.0.0.0:0              LISTENING       1060
  TCP    0.0.0.0:4609           0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:5000           0.0.0.0:0              LISTENING       1300
  TCP    192.168.123.192:139    0.0.0.0:0              LISTENING       4
  TCP    192.168.123.192:445    192.168.123.116:1146   ESTABLISHED     4
  TCP    192.168.123.192:445    192.168.123.192:4609   ESTABLISHED     4
  TCP    192.168.123.192:4530   0.0.0.0:0              LISTENING       4
  TCP    192.168.123.192:4530   192.168.123.116:139    ESTABLISHED     4
  TCP    192.168.123.192:4607   192.168.123.192:445    TIME_WAIT       0
  TCP    192.168.123.192:4609   192.168.123.192:445    ESTABLISHED     4
  TCP    192.168.123.192:13526  0.0.0.0:0              LISTENING       624
  UDP    0.0.0.0:445            *:*                                    4
  UDP    0.0.0.0:500            *:*                                    804
  UDP    0.0.0.0:1026           *:*                                    1272
  UDP    0.0.0.0:1027           *:*                                    624
  UDP    0.0.0.0:1036           *:*                                    1272
  UDP    0.0.0.0:2583           *:*                                    1272
  UDP    127.0.0.1:123          *:*                                    1060
  UDP    127.0.0.1:1035         *:*                                    1884
  UDP    127.0.0.1:1054         *:*                                    376
  UDP    127.0.0.1:1900         *:*                                    1300
  UDP    127.0.0.1:3220         *:*                                    1680
  UDP    192.168.123.192:123    *:*                                    1060
  UDP    192.168.123.192:137    *:*                                    4
  UDP    192.168.123.192:138    *:*                                    4
  UDP    192.168.123.192:1900   *:*                                    1300
  UDP    192.168.123.192:2923   *:*                                    624
  UDP    192.168.123.192:16807  *:*                                    624

C:\jumpkit>
```

Figure 9.3 shows typical output from the Netstat command. By using the *–a* switch as shown , you can easily identify what ports are open and what machine is making use of them. By column, the screenshot shows the following data.

- **Proto** This identifies the protocol that is in use, either TCP or UDP.
- **Local Address** This identifies the local hostname and port number/common name.
- **Foreign Address** This identifies the remote machine and port that is connected.
- **State** This identifies whether or not the port is listening for connections, has an established connection, or perhaps neither.
- **PID** This identifies the process identifier of the process that is using that connection.

The Netstat command does provide some good, if cursory, information that is easily and natively available. Quickly, without adding any applications to the suspect machine, you have a listing of all the network connections that are in use on that machine. Netstat does fall short of providing all the information you'll need, but there are other tools that will be covered that will fill in the gaps.

NOTE

Only Windows XP and higher versions of Netstat have the "-*o*" option available. Only Windows XP SP2 and higher have the "-*b*" command line option available.

Analyzing the output from Netstat is pretty straightforward. You're basically looking for any line item that shows any network port open that shouldn't be. However, it takes some experience to know what ports should or shouldn't be open. In order to narrow the field, read the output from the right to the left. Go down the "state" column, and stop on any line item where it says "established." Any connection that is in this state is actively communicating on that port. Once you've identified an established connection, go over to the "foreign address" column. This column will tell you to what address or system the connection is established. If that column has "localhost" or "127.0.0.1" in it, you can rule that one out at this time, as the machine is connecting back into itself. When the foreign address column shows an external hostname or IP address, and the state column shows that it's established, that suggests a likely malware candidate. I would suggest that you go ahead and write down the information on that row, to compare it to the results of running the more advanced tools that we'll be going through later on in this chapter.

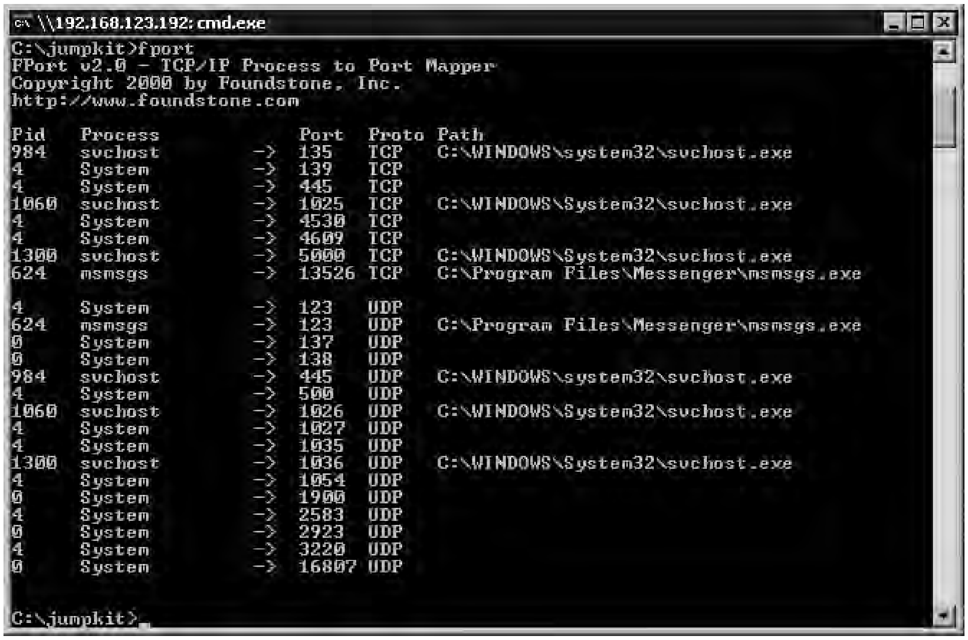
Once you review the output from Netstat, you may have an idea of what port the malicious code is using, but you're still pretty much in the dark as to exactly where that program is running from, or exactly what it is. More importantly, you still don't have a sample of the malware to analyze.

Process and Network Service Detection Tools

In this section, we'll be discussing three core tools for process and network service detection. No analyst's toolbox should be without FPort, tcpvcon, and Handle.

FPort is available free of charge from Foundstone's free resources page at www.foundstone.com/resources/freetools.html. FPort is a tool that provides some of the same information that Netstat does, but with one very significant difference: it gives you the complete path to the process that is making use of the port, along with the PID. Figure 9.4 shows what you can expect to see in a typical report from FPort, run with default options.

Figure 9.4 Typical FPort Output



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

Pid   Process            Port  Proto Path
984   svchost       ->  135   TCP   C:\WINDOWS\system32\svchost.exe
4     System        ->  139   TCP
4     System        ->  445   TCP
1060  svchost       ->  1025  TCP   C:\WINDOWS\System32\svchost.exe
4     System        ->  4530  TCP
4     System        ->  4609  TCP
1300  svchost       ->  5000  TCP   C:\WINDOWS\System32\svchost.exe
624   msmsgs        ->  13526 TCP   C:\Program Files\Messenger\msmsgs.exe

4     System        ->  123   UDP
624   msmsgs        ->  123   UDP   C:\Program Files\Messenger\msmsgs.exe
0     System        ->  137   UDP
0     System        ->  138   UDP
984   svchost       ->  445   UDP   C:\WINDOWS\system32\svchost.exe
4     System        ->  500   UDP
1060  svchost       ->  1026  UDP   C:\WINDOWS\System32\svchost.exe
4     System        ->  1027  UDP
4     System        ->  1035  UDP
1300  svchost       ->  1036  UDP   C:\WINDOWS\System32\svchost.exe
4     System        ->  1054  UDP
0     System        ->  1900  UDP
4     System        ->  2583  UDP
0     System        ->  2923  UDP
4     System        ->  3220  UDP
0     System        ->  16807 UDP

C:\jumpkit>_
```

You can see from the example in Figure 9.4, that the information is very similar to that of the NETSTAT command.

- **PID** This is the PID of the running process.
- **Process** This is the name of the process running.
- **Port** This is the network port that the process is using.
- **Proto** This is the protocol that the process is using to connect on the port.
- **Path** This is the fully qualified path, showing exactly where the executable is located.

Here are the command line switches available for FPORT:

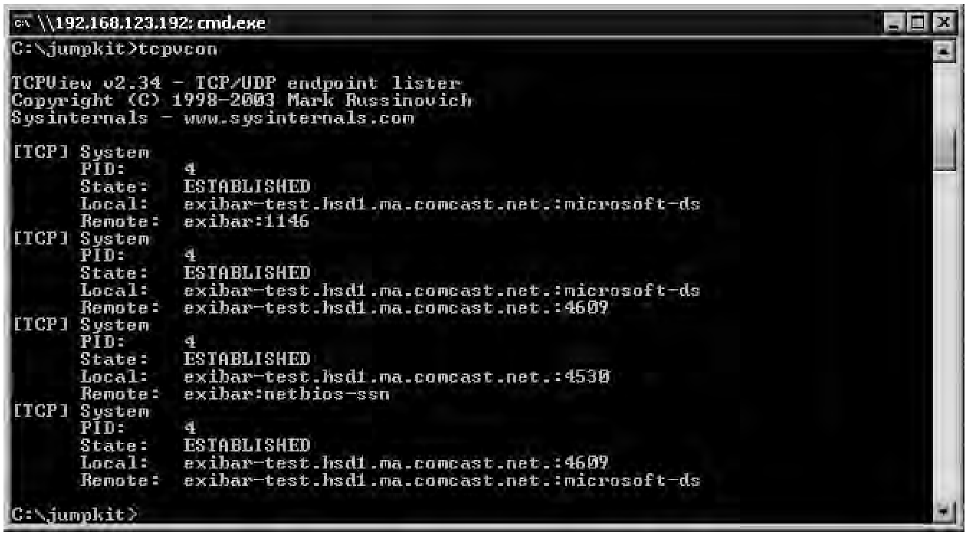
- ***/p*** Sort by port
- ***/a*** Sort by application
- ***/i*** Sort by PID
- ***/ap*** Sort by application path

By reviewing FPort's output, you can gather some very useful information. Within the output of FPort, you have the PID, port, and path items. These are extremely useful for finding the process that is causing problems, and retrieving that all-important malware sample.

For instance, if you happen to see a process called *svchost.exe* running, you might not think twice about it if you're using the Windows task manager's process view, or if you're looking at the output from Netstat. But with the output from FPort, you can see where *svchost.exe* is running from. *Svchost.exe* will normally run from the *c:\windows\system32* (*<systemroot>\system32*) directory. If you see it running from any other directory, that would normally be a piece of malware trying to trick you into thinking it's the real SVCHost process, one of a number of very common SVCHost impersonation tricks.

TCPVCON is another tool that should be in your malware jumpkit, and is available freely from www.sysinternals.com as part of the TCPView application download. tcpvcon provides another view on network connections. Figure 9.5 shows an example of tcpvcon's default output. If there aren't any command line switches applied, tcpvcon will show you only the established connections.

Figure 9.5 Example of Default Output from Running *tcpvcon*



```
\\192.168.123.192: cmd.exe
C:\jumpkit>tcpvcon

TCPView v2.34 - TCP/UDP endpoint lister
Copyright (C) 1998-2003 Mark Russinovich
Sysinternals - www.sysinternals.com

[TCP] System
      PID:     4
      State:   ESTABLISHED
      Local:   exibar-test.hsd1.ma.comcast.net.:microsoft-ds
      Remote:  exibar:1146
[TCP] System
      PID:     4
      State:   ESTABLISHED
      Local:   exibar-test.hsd1.ma.comcast.net.:microsoft-ds
      Remote:  exibar-test.hsd1.ma.comcast.net.:4609
[TCP] System
      PID:     4
      State:   ESTABLISHED
      Local:   exibar-test.hsd1.ma.comcast.net.:4530
      Remote:  exibar:netbios-ssn
[TCP] System
      PID:     4
      State:   ESTABLISHED
      Local:   exibar-test.hsd1.ma.comcast.net.:4609
      Remote:  exibar-test.hsd1.ma.comcast.net.:microsoft-ds

C:\jumpkit>
```

tcpvcon has another very useful feature. You can provide the PID of a process and tell tcpvcon to show you only the network connections that the supplied PID has created. You can do this by using the "*netstat -ao*" command that was discussed earlier in this chapter.

NOTE

TCPVCON is downloaded as part of the GUI application TCPView. TCPView performs the same checks and displays the same information as tcpvcon, but does so courtesy of a nice GUI version that has some added features such as real-time auto refresh, sorting, nice graphical layout, and so forth. If you're looking for malware and have full GUI control of the target machine, or are at the keyboard of the target machine itself, TCPVIEW is certainly a tool worth using.

These are tcpvcon's syntax and command line switches:

- ***tcpvcon*** *[-a] [-c] [-n]* [process name or PID]
- ***-a*** Show all endpoints.
- ***-c*** Print output as CSV.
- ***-n*** Don't resolve addressed process. Only show endpoints owned by the process specified.

Tcpvcon gives you a nice quick look at what network processes are running, and the path that they are running from. This enables you to establish quickly if a process with a familiar name is running from a directory that it shouldn't be running from, and allows you to take further action on that process.

After running the aforementioned tools you'll usually have a pretty good idea of which process is your item of malware. But, in some cases, you might be looking at malware that isn't readily apparent, or perhaps it has some rootkit-like qualities and is attempting to hide itself. Also, some malware will not open up a network connection on its own, but will communicate over the Internet using whatever Windows utilities or installed software applications are available. Perhaps the malware you're looking for is attempting to gather personal information and send it via e-mail or Instant Messaging (IM) to a location on the Internet, or even saving it to your local hard drive for later retrieval?

This is where *Handle.exe* comes into its own (available from www.microsoft.com/technet/sysinternals/Processesandthreadsutilities.mspx). Handle is the command-line version of Sysinternals' Process Explorer. When it is run on a system, by default it displays all the file objects or "handles" that are open or in use by a particular process. Any program that has a file or directory open will show up on the default listing. This includes Data Link Libraries (DLLs), files, directories, and so on. Running Handle with the "*-a*" command-line switch will cause it to display everything that is in use, not just objects that refer to files. Reviewing the output when the "*-a*" switch is used, can be a little bit overwhelming at first. Anything else that may be in use in memory, such as ports, registry keys, synchronization primitives, threads, and individual processes, will also be shown.

NOTE

The output from running Handle can be quite large on most systems, especially if the "*-a*" switch is used. For this reason, I recommend redirecting the output from the screen to a file as with other tools by appending a redirection instruction like *>handle_output.txt* to the end of the *handle.exe* command line in order to more easily review the output.

Here's the syntax for the Handle command line:

```
handle [[-a] [-u] | [-c <handle> [-y]] | [-s]] [-p <processname>|<pid>> [name]
```

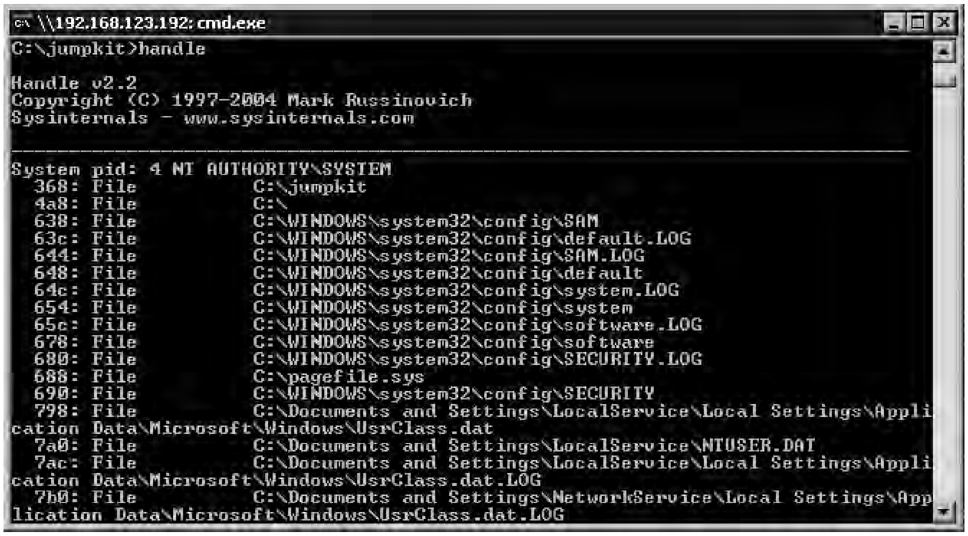
- ***-a*** Show information about all types of handles, including ports, Registry keys, synchronization primitives, threads, and processes as well as files.
- ***-c*** Closes the specified handle, identified by its PID.
- ***-y*** Close handle without prompting for confirmation.

- ***-s*** Show count of each type of open handle.
- ***-u*** Show the user name that owns an open handles.
- ***-p*** Don't examine all the handles: restrict scan to those processes that begin with "*<processname>*." For example, "*handle -p svc*" dumps open files for all processes whose names begin with *svc*.
- ***Name*** Search for references to an object with a particular name. For example, "*handle windows\system*" would find processes that have opened. Search is case-insensitive.

You can get additional information on Handle's syntax and parameters at www.microsoft.com/technet/sysinternals/ProcessesAndThreads/Handle.mspx.

Figure 9.6 shows a small example of what can be expected from Handle's output, which is broken down into sections separated by a row of dashes. The first line in each section shows the System PID, and the name of the process. In this case, the system PID is "*4*" and the name of the process is "NT AUTHORITY\SYSTEM." The data that follows after that first line in the section contains all the handles that are associated with that process. The first column is the specific handle identifier represented as a hexadecimal number, and is unique to that PID. The next column identifies the type of handle. Some of the most common identifiers that will appear in that column represent a process, an individual thread, a registry key, a port, an event, a section of memory, a token, a file, and a directory. The next column gives you the fully qualified path of what is in use by that particular PID, whether it's a register key, file, or directory.

Figure 9.6 Small Portion of the Default Output from Running Handle



```
C:\jumpkit>handle

Handle v2.2
Copyright (C) 1997-2004 Mark Russinovich
Sysinternals - www.sysinternals.com

------------------------------------------------------------------------------
System pid: 4 NT AUTHORITY\SYSTEM
   368: File          C:\jumpkit
   4a8: File          C:\
   638: File          C:\WINDOWS\system32\config\SAM
   63c: File          C:\WINDOWS\system32\config\default.LOG
   644: File          C:\WINDOWS\system32\config\SAM.LOG
   648: File          C:\WINDOWS\system32\config\default
   64c: File          C:\WINDOWS\system32\config\system.LOG
   654: File          C:\WINDOWS\system32\config\system
   65c: File          C:\WINDOWS\system32\config\software.LOG
   678: File          C:\WINDOWS\system32\config\software
   680: File          C:\WINDOWS\system32\config\SECURITY.LOG
   688: File          C:\pagefile.sys
   690: File          C:\WINDOWS\system32\config\SECURITY
   798: File          C:\Documents and Settings\LocalService\Local Settings\Appli
cation Data\Microsoft\Windows\UsrClass.dat
   7a0: File          C:\Documents and Settings\LocalService\NTUSER.DAT
   7ac: File          C:\Documents and Settings\LocalService\Local Settings\Appli
cation Data\Microsoft\Windows\UsrClass.dat.LOG
   7b0: File          C:\Documents and Settings\NetworkService\Local Settings\App
lication Data\Microsoft\Windows\UsrClass.dat.LOG
```

Tools & Traps

Avoiding the EULA trap

In mid 2006, Microsoft purchased Sysinternals, the makers of PSTools, Handle, and tcpvcon, which are described in detail above. There are some changes occurring to the Sysinternals tools. Most of these changes involve the addition of features to the tools. One change that is being made is the addition of an End User License Agreement (EULA) that must be agreed to before each tool is run for the first time on a new system. This usually wouldn't be a problem, but with the command-line tools that we're running on a remote system, there is a potential issue. When the tool is run the first time, it will attempt to display an EULA on the remote machine, which will never actually be displayed. The tool will appear to hang and do nothing as it's waiting for the EULA to be clicked on and agreed upon. (It's also worth remembering that when you redirect output to a file using the > or >> modifiers, you need to test for unexpected input that will result in a hang like this, usually requiring a Ctrl-C to get out of.)

There are three workarounds for this situation that will allow us to run our tools in the non-interactive manner that suits our needs. Any one of the following will prevent the displaying of the EULA, according to the Sysinternals and Microsoft Web sites:

- Microsoft has added a new command-line switch to all of the utilities in the PSTOOLS suite, that will prevent the tool from waiting for you to confirm agreement to the EULA. The command-line switch that is needed is "*-accepteula*." Microsoft has also said that this switch will be added to all the Sysinternals command-line tools. However, at the time of this writing, only the PSTOOLS recognize this command-line switch.
- If you run the following commands from within the remote command shell, as explained earlier, they will insert the registry keys necessary to allow the tools to run without displaying the EULA and short-circuiting your investigative processes by counterfeiting an application hang.
    - **PsExec** reg add *HKCU\Software\Sysinternals\PsExec /v EulaAccepted /d 1*
    - **TCPVCON** reg add *HKCU\Software\Sysinternals\Tcpvcon /v EulaAccepted /d 1*
    - **Handle** reg add *HKCU\Software\Sysinternals\Handle /v EulaAccepted /d 1*
- Use a version of the above tools that was released prior to November 1, 2006, which was when the EULA code was added to almost all Sysinternals products.

With the use of the aforementioned tools, you should be able to identify the particular malware that is causing the problems on your organization's machine or network. Once you've identified the specific file, the first thing that you need to do is to retrieve a sample of that file and secure it.

Gathering and securing the sample file is very easily done. What you *must* do is archive it and add a password to that archive. Archiving addresses a couple of critical issues. First and foremost, it secures the sample in a manner that ensures that it isn't easily executed by accident. You certainly wouldn't want to infect your own machine accidentally! Secondly, it prevents the sample from being accidentally removed by one of the AV or anti-spyware programs that should be running on your own machine. In many scenarios, you'll only have access to an infected machine for a short time, and you'll often only have access to a single live sample of the malware. It is very important to keep that item of malware secure, until you're sure you don't need it any longer and you've been able to put the proper countermeasures in place on your network. You also need to be able to distribute the proper cleaning procedures to those that will be performing the actual disinfection of the malware from your networked machines.

TIP

Floppy disks, though gradually disappearing from computer store shelves, are very convenient to use to store live samples on. You can enable write protection very easily, thus ensuring that your hard-earned live sample isn't accidentally deleted.

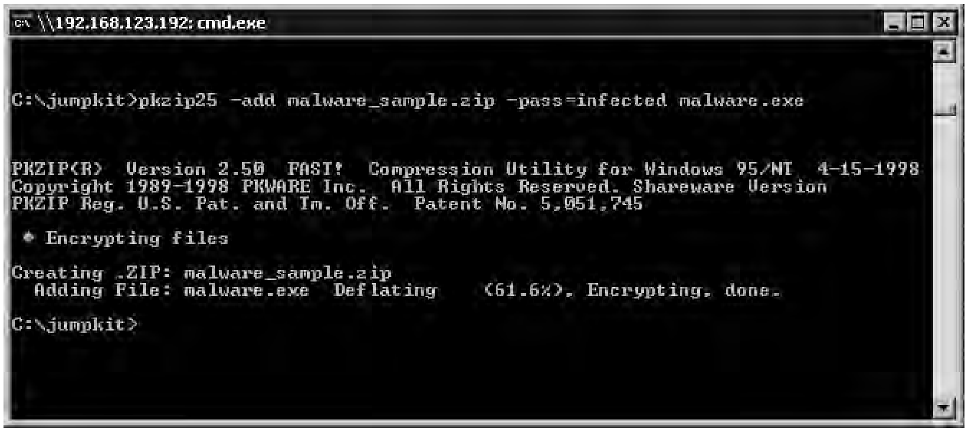
This author always has a supply of bright red floppies on hand that are used solely for storing live samples on. The bright red color makes them stand out, and along with the words "LIVE VIRUS," or similar, written on the label certainly flags them as something out of the ordinary, and stresses the need to handle the contents with care.

The archiving method that some like to use is the old tried and true command line version PKZip, from pkware (www.pkware.com). You may use any archival program that you prefer, but I like to stick with the standards whenever possible. All of the AV vendors and all of the online malware inspection tools accept *.ZIP* files for analysis, which makes our jobs a bit easier if we archive using the *.ZIP* format right from the start of sample collection. You might also prefer *pkzip* due to the fact that it's a stand-alone command line tool. You can copy that tool over to the machine that you are investigating without much hassle, and it's available to use right away. Figure 9.7 shows the typical use of *pkzip* in sample collection.

An example of the typical usage of *pkzip*, versions 2.5, is given here. However, you probably won't get much support for DOS and old Windows versions from PKWare nowadays. On the other hand, the format for the *.zip* archiving standard is freely available, and many other archiving programs fully support it.

```
pkzip25 -add <name of archive>.zip -pass=<password> <filename to be archived>
```

Figure 9.7 A Typical Usage of *pkzip25*



```
C:\jumpkit>pkzip25 -add malware_sample.zip -pass=infected malware.exe


PKZIP(R)  Version 2.50  FAST!  Compression Utility for Windows 95/NT  4-15-1998
Copyright 1989-1998 PKWARE Inc.  All Rights Reserved. Shareware Version
PKZIP Reg. U.S. Pat. and Tm. Off.  Patent No. 5,051,745

 • Encrypting files

Creating .ZIP: malware_sample.zip
  Adding File: malware.exe  Deflating   (61.6%), Encrypting, done.

C:\jumpkit>
```

Command line switches used in the example in Figure 9.7 are as follows:

- ***PKZIP25*** Executable command
- ***-add*** Instructs *pkzip* to add a file to an archive
- ***<name of archive>.zip*** This is the archive file (*.ZIP* file) that you wish to add your malware sample to; if the file doesn't exist, it will be created for you.
- ***-pass=<password>*** This switch allows you to add a password to the *.ZIP* file that you are creating
- ***<filename to be archived>*** This is the file that you wish to add to the *.ZIP* file, your malware sample itself.

pkzip25 has many features and switches aside from the ones described above. We will not go into all of them here, as it's out of scope for this book. If you are interested in using *pkzip* to perform your command-line archiving, we strongly suggest you check out all the features available in current versions by going to www.pkware.com, where there are a number of 30-day evaluation versions.

TIP

Regardless of what tool you wish to use to create your *.ZIP* file, you'll always want to apply a password to the *.ZIP* file when archiving or submitting malicious files. The AV industry has unofficially adopted as a *de facto* standard the password "infected" for compressed files. That password is almost universally used throughout the industry. Most online scanners, online analysis tools, or submissions to an AV vendor will assume that a .ZIP file has that password when it's submitted for analysis. However, you should always check the terms and conditions and other requirements on such Web sites before submission.

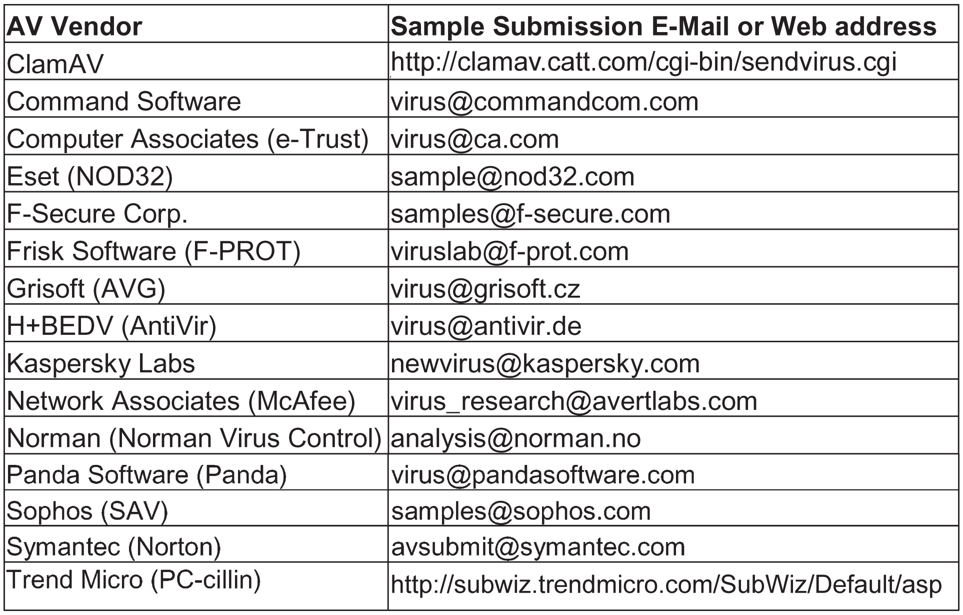
Web-based Inspection and Virus Analysis Tools

Now that you have a sample of the malicious software that is causing trouble, what do you do with it? How can you tell what it's doing, what communication channels it's set up, and what it's doing to the machine it's running on? This section will describe the next actions to take once you have secured your sample. It's worth noting here that you should always have a backup of your sample on a floppy disk or other removable or write-protected media. Often, the original file gets corrupted, deleted, or rendered unusable during analytical processing.

AV Vendors Accept Submissions

All AV vendors accept live samples either through a Web form, or through e-mail. Figure 9.8 contains a list of the major AV vendors and their malware submission e-mail or Web address for easy reference. The first place to which you should send your live malware sample is the AV vendor whose product is used in your organization. If your AV vendor isn't in the list in Figure 9.8, it is usually very easy to find the address by going to your vendor's Web site and searching for something like *sample submission* on their site.

Figure 9.8 Common AV Vendors and Their Sample Submission Addresses



AV Vendor	Sample Submission E-Mail or Web address
ClamAV	http://clamav.catt.com/cgi-bin/sendvirus.cgi
Command Software	virus@commandcom.com
Computer Associates (e-Trust)	virus@ca.com
Eset (NOD32)	sample@nod32.com
F-Secure Corp.	samples@f-secure.com
Frisk Software (F-PROT)	viruslab@f-prot.com
Grisoft (AVG)	virus@grisoft.cz
H+BEDV (AntiVir)	virus@antivir.de
Kaspersky Labs	newvirus@kaspersky.com
Network Associates (McAfee)	virus_research@avertlabs.com
Norman (Norman Virus Control)	analysis@norman.no
Panda Software (Panda)	virus@pandasoftware.com
Sophos (SAV)	samples@sophos.com
Symantec (Norton)	avsubmit@symantec.com
Trend Micro (PC-cillin)	http://subwiz.trendmicro.com/SubWiz/Default/asp

By getting the sample off to your AV vendor right away, you will ensure that they'll be able to start working on the appropriate definition (signature) files that are needed to detect and remove the malware that you sent them. Once they have completed their analysis of the sample and they've sent you the necessary definition update files, you'll want to test and deploy them throughout your enterprise right away.

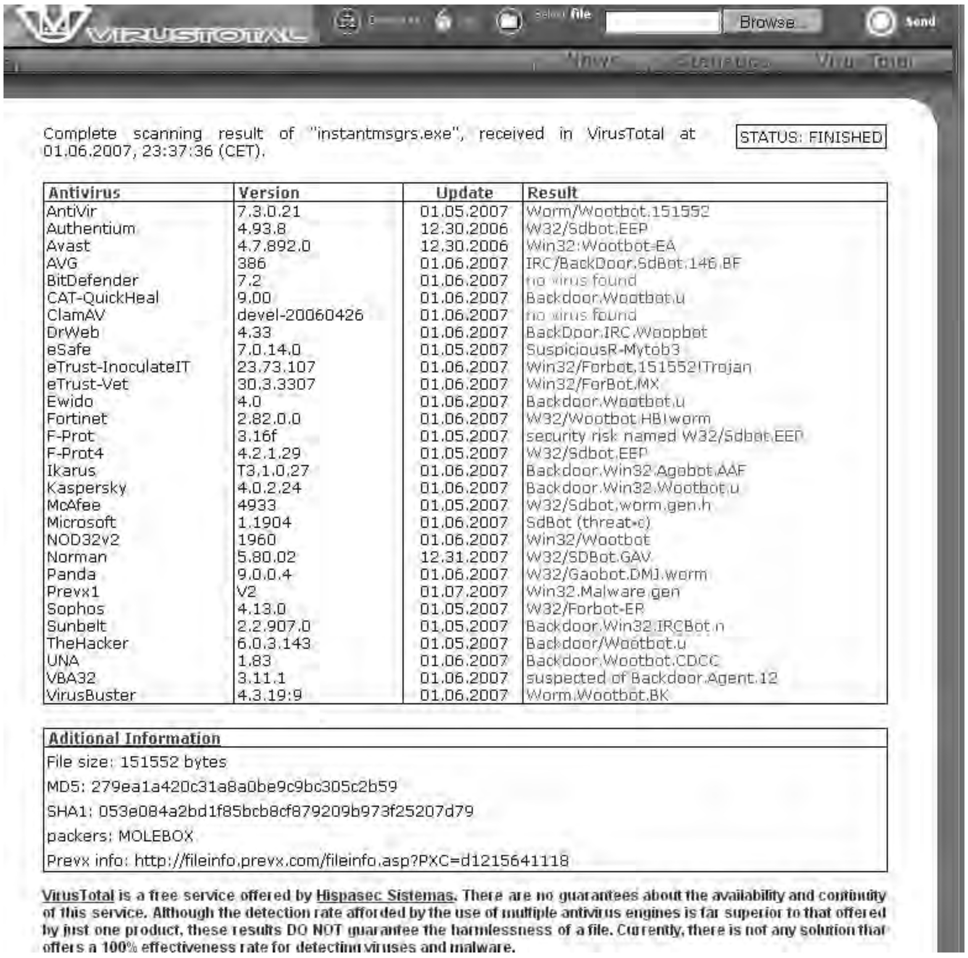
After you've sent off your sample, you'll want more information about what the program does to a machine and/or network, and whether it's already a known threat. There are a few Web-based sites that accept malware sample submissions, which will analyze them for you, up to a point.

The first two Web sites that we'd like to discuss, allow anyone to submit samples for scanning, and will report their findings. They both use multiple AV scan engines and will report back to you if any of those engines detect your sample as a known threat, and will tell you what that particular vendor has named it. This is very valuable as a front line check. The results of having VirusTotal (www.virustotal.com) and Jotti Malware Scanner (http://virusscan.jotti.org) scan a submitted item of malware are shown in Figure 9.9 and Figure 9.10, respectively. (You might also want to try Virus.Org (http://scanner.virus.org, which offers a similar service.)

The file that I submitted for this example is a variant of W32/WOOTBOT, which was discovered in March of 2005. Something that you may notice right away is that not all the products identify the sample as a known threat, and that where a threat is identified, the name used may vary widely from product to product. This is often the case. There are so many new malicious programs and variants being created, that it's likely that not all AV vendors will be able to detect 100 percent of them, especially when they first appear. This reinforces the reasoning behind sending your sample to the vendor of your enterprise AV solution as quickly as possible. The second item is that there isn't a standard naming convention within the industry for malware threats. Or to be precise, there is more than one, and it's not practically possible to enforce the universal use of a single standard. (This issue is addressed at greater length in Chapter 1.) Each scanning engine may detect different pieces of a blended or multipolar threat, and the same code can turn up in more than one threat family. This complicates the issue, when different vendors ascribe the same malware to different threat families. We would recommend using the same naming convention within your enterprise that your AV vendor uses, for internal documentation and consistency. However, when discussing malicious code with other vendors, users of other software, mailing lists like those available to AVIEN and AVIEWS members, and so on, you will probably find that it's useful to cross-reference this name with other names, such as that used by the WildList Organization (www.wildlist.org), the Computer AV Research Organization (CARO - www.caro.org), or even the Common Malware Enumeration (CME list) (http://cme.mitre.org/)

Using VirusTotal also has a benefit in that they will submit your sample to the AV vendors that they use for their Web-hosted scans. I would still submit it yourself to your AV vendor, even knowing that VirusTotal will submit as well. Many AV vendors rank the prevalence of each piece of malware according to how many unique submissions they receive from their customers, so it's important to have your sample counted toward their total.

Figure 9.9 Results of Submitting "*instantmsgrs.exe*" Malware Sample to www.virustotal.com



Complete scanning result of "instantmsgrs.exe", received in VirusTotal at 01.06.2007, 23:37:36 (CET).

STATUS: FINISHED

Antivirus	Version	Update	Result
AntiVir	7.3.0.21	01.05.2007	Worm/Wootbot.151552
Authentium	4.93.8	12.30.2006	W32/Sdbot.EEP
Avast	4.7.892.0	12.30.2006	Win32:Wootbot-EA
AVG	386	01.06.2007	IRC/BackDoor.SdBot.146.BF
BitDefender	7.2	01.06.2007	no virus found
CAT-QuickHeal	9.00	01.06.2007	Backdoor.Wootbot.u
ClamAV	devel-20060426	01.06.2007	no virus found
DrWeb	4.33	01.06.2007	BackDoor.IRC.Woopbot
eSafe	7.0.14.0	01.05.2007	SuspiciousR-Mytob3
eTrust-InoculateIT	23.73.107	01.06.2007	Win32/Forbot.151552!Trojan
eTrust-Vet	30.3.3307	01.06.2007	Win32/ForBot.MX
Ewido	4.0	01.06.2007	Backdoor.Wootbot.u
Fortinet	2.82.0.0	01.06.2007	W32/Wootbot.HB!worm
F-Prot	3.16f	01.05.2007	security risk named W32/Sdbot.EEP
F-Prot4	4.2.1.29	01.05.2007	W32/Sdbot.EEP
Ikarus	T3.1.0.27	01.06.2007	Backdoor.Win32.Agobot.AAF
Kaspersky	4.0.2.24	01.06.2007	Backdoor.Win32.Wootbot.u
McAfee	4933	01.05.2007	W32/Sdbot.worm.gen.h
Microsoft	1.1904	01.06.2007	SdBot (threat-c)
NOD32v2	1960	01.06.2007	Win32/Wootbot
Norman	5.80.02	12.31.2007	W32/SDBot.GAV
Panda	9.0.0.4	01.06.2007	W32/Gaobot.DMJ.worm
Prevx1	V2	01.07.2007	Win32.Malware.gen
Sophos	4.13.0	01.05.2007	W32/Forbot-ER
Sunbelt	2.2.907.0	01.05.2007	Backdoor.Win32.IRCBot.n
TheHacker	6.0.3.143	01.05.2007	Backdoor/Wootbot.u
UNA	1.83	01.06.2007	Backdoor.Wootbot.CDCC
VBA32	3.11.1	01.06.2007	suspected of Backdoor.Agent.12
VirusBuster	4.3.19:9	01.06.2007	Worm.Wootbot.BK

Aditional Information

File size: 151552 bytes

MD5: 279ea1a420c31a8a0be9c9bc305c2b59

SHA1: 053e084a2bd1f85bcb8cf879209b973f25207d79

packers: MOLEBOX

Prevx info: http://fileinfo.prevx.com/fileinfo.asp?PXC=d1215641118

VirusTotal is a free service offered by Hispasec Sistemas. There are no guarantees about the availability and continuity of this service. Although the detection rate afforded by the use of multiple antivirus engines is far superior to that offered by just one product, these results DO NOT guarantee the harmlessness of a file. Currently, there is not any solution that offers a 100% effectiveness rate for detecting viruses and malware.

Tools & Traps

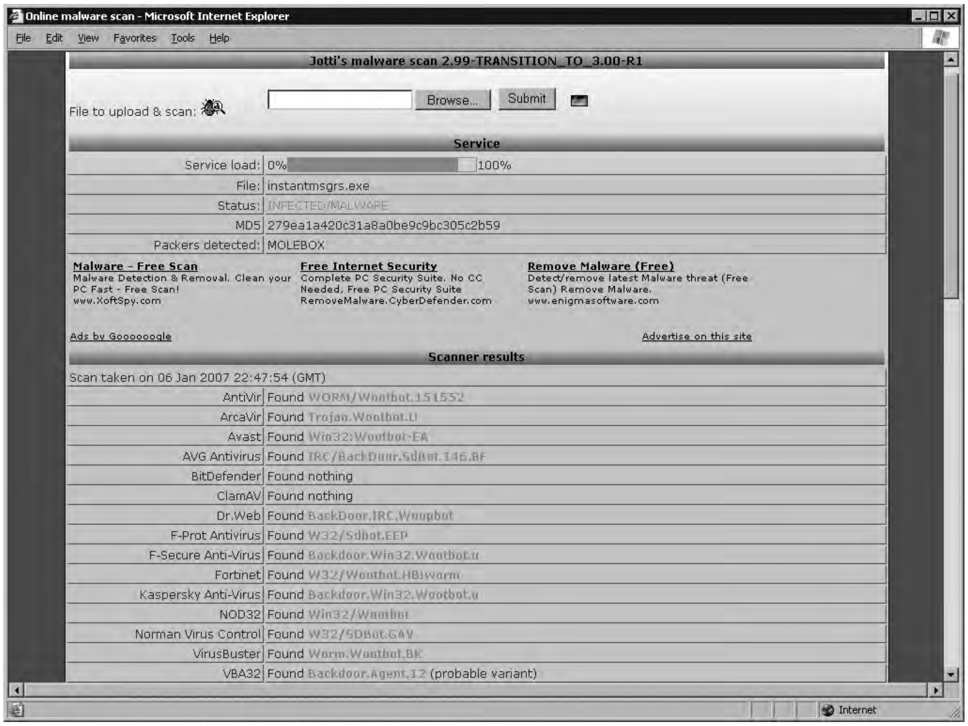
Malware Submission Sites – The Downside

Sites like Jotti and VirusTotal can be useful in the early stages of an outbreak, as part of the process of identifying the precise nature of a possible threat. However, there are dangers in putting too much trust in these sites. Most obviously, there is no way of guaranteeing that a file not identified by any of the engines used as being malicious might not turn out to be malicious, nonetheless. Not proven is not the same as innocent.

However, you're not just putting your trust in the vendors whose scanning engines are used. You're also placing your trust in the site to update the scanner promptly, configure it appropriately, and use an appropriate methodology. For example, such sites are usually reticent on precisely how they implement their scanning in terms of the level of heuristics employed. This is not altogether a bad thing. It makes it (a little) harder for the bad guys to misuse a site to test their malware fully against their scanners for free.

It does, however, make it harder to evaluate their accuracy and fitness for particular purposes. A false positive, false negative, or misdiagnosis can be hard to interpret correctly. In particular, it's not a good idea to use these sites as a basis for comparative evaluation of scanners.

Figure 9.10 Results of Scanning Sample "*instantmsgrs.exe*" on http://virusscan.jotti.org


Online malware scan - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Browse...
Submit
Internet

Jotti's malware scan 2.99-TRANSITION_TO_3.00-R1

File to upload & scan:

Service

Service load:	0% 100%
File:	instantmsgrs.exe
Status:	INFECTED/MALWARE
MD5	279ea1a420c31a8a0be9c9bc305c2b59
Packers detected:	MOLEBOX

Malware - Free Scan
Malware Detection & Removal. Clean your PC Fast - Free Scan!
www.XoftSpy.com

Free Internet Security
Complete PC Security Suite. No CC Needed, Free PC Security Suite
RemoveMalware.CyberDefender.com

Remove Malware (Free)
Detect/remove latest Malware threat (Free Scan) Remove Malware.
www.enigmasoftware.com

Ads by Goooooogle

Advertise on this site

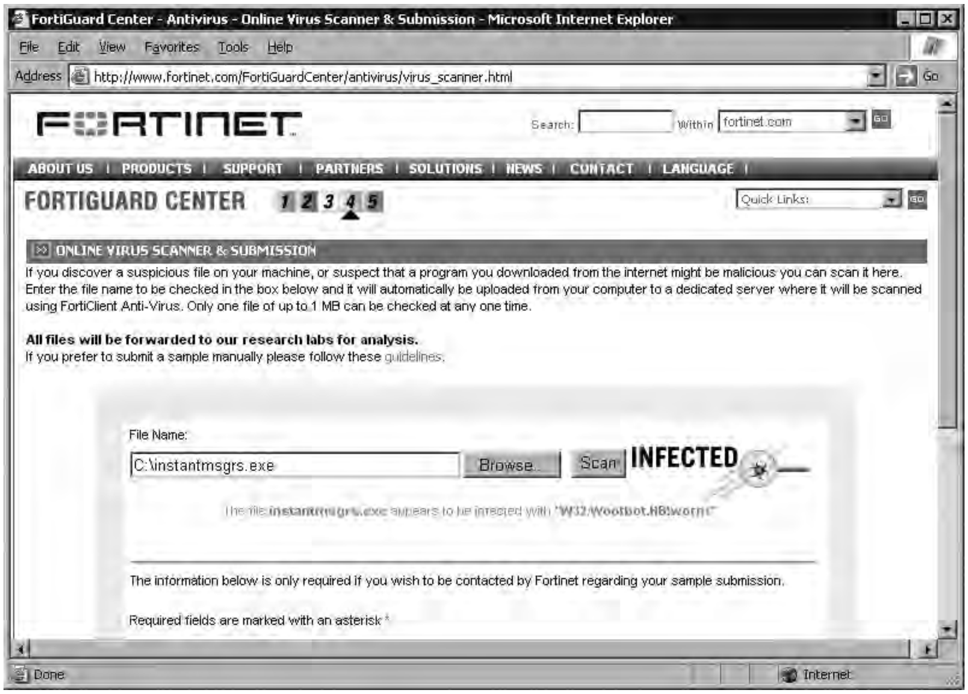
Scanner results

Scan taken on 06 Jan 2007 22:47:54 (GMT)

AntiVir	Found WORM/Wootbot.151552
ArcaVir	Found Trojan.Wootbot.U
Avast	Found Win32:Wootbot-EA
AVG Antivirus	Found IRC/BackDoor.SdBot.146.BF
BitDefender	Found nothing
ClamAV	Found nothing
Dr.Web	Found BackDoor.IRC.Wuupbot
F-Prot Antivirus	Found W32/Sdbot.EEP
F-Secure Anti-Virus	Found Backdoor.Win32.Wootbot.u
Fortinet	Found W32/Wootbot.HB!worm
Kaspersky Anti-Virus	Found Backdoor.Win32.Wootbot.u
NOD32	Found Win32/Wootbot
Norman Virus Control	Found W32/SDBot.GAV
VirusBuster	Found Worm.Wootbot.BK
VBA32	Found Backdoor.Agent.12 (probable variant)

At the time of this writing, we could only find three AV vendors that would accept a single sample online, perform a check on that individual file, and display the results, though several vendors (Trend, ESET et al.) have online scanners that will check a whole system. Granted, the results are very basic and limited, usually to the display of no more than "possibly not-infected, further analysis needed" or "infected with <*malware name*>." In McAfee's case, if it's a new malicious program that isn't within their daily definition files, they will send you a special definition file (an "*extra.dat*" file) that you can use to detect and clean the submitted malware, which is very useful if your organization is a McAfee customer. Figures 9.11, 9.12, and 9.13 are screenshots showing typical results of submitting a known malicious program to each of these three vendors for instant online analysis. If you're a customer of one of the three, there is a particular benefit to using their online analysis. The first benefit is that you're sending them a suspicious file that their virus engineers will be able to analyze for incorporation into their next update, if appropriate. The second, if it's a known threat to them, is that you'll find out right away what they are calling this threat, and what update is needed to detect and clean it. If you're not a customer of any of the three, then it may be enough simply to use VirusTotal or Jotti Malware Scanner and let those Web sites check your sample against many different AV products. There is, however, the possibility that a more accurate result will be obtained from the configuration on the vendor's site.

Figure 9.11 Fortinet.com Results After Scanning "*instantmsgrs.exe*"


FortiGuard Center - Antivirus - Online Virus Scanner & Submission - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://www.fortinet.com/FortiGuardCenter/antivirus/virus_scanner.html
FORTINET
Search: Within fortinet.com GO
ABOUT US | PRODUCTS | SUPPORT | PARTNERS | SOLUTIONS | NEWS | CONTACT | LANGUAGE |
FORTIGUARD CENTER 1 2 3 4 5
Quick Links:
ONLINE VIRUS SCANNER & SUBMISSION
Browse...
Scan
INFECTED
Done
Internet

If you discover a suspicious file on your machine, or suspect that a program you downloaded from the internet might be malicious you can scan it here. Enter the file name to be checked in the box below and it will automatically be uploaded from your computer to a dedicated server where it will be scanned using FortiClient Anti-Virus. Only one file of up to 1 MB can be checked at any one time.

All files will be forwarded to our research labs for analysis.
If you prefer to submit a sample manually please follow these guidelines.

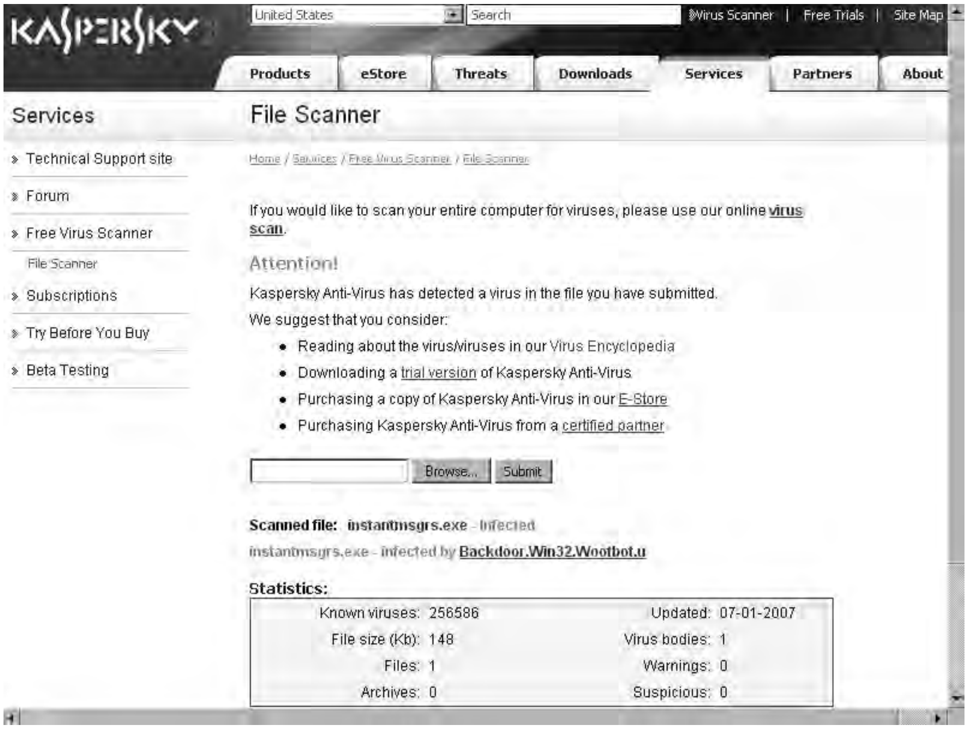
File Name: C:\instantmsgrs.exe

The file **instantmsgrs.exe** appears to be infected with **"W32/Wootbot.HB!worm"**

The information below is only required if you wish to be contacted by Fortinet regarding your sample submission.

Required fields are marked with an asterisk *

Figure 9.12 Kaspersky.com Results After Submitting "*instantmsgrs.exe*"



United States | Search | Virus Scanner | Free Trials | Site Map

Products | eStore | Threats | Downloads | Services | Partners | About

Services

- Technical Support site
- Forum
- Free Virus Scanner
  - File Scanner
- Subscriptions
- Try Before You Buy
- Beta Testing

File Scanner

Home / Services / Free Virus Scanner / File Scanner

If you would like to scan your entire computer for viruses, please use our online virus scan.

Attention!

Kaspersky Anti-Virus has detected a virus in the file you have submitted.

We suggest that you consider:

- Reading about the virus/viruses in our Virus Encyclopedia
- Downloading a trial version of Kaspersky Anti-Virus
- Purchasing a copy of Kaspersky Anti-Virus in our E-Store
- Purchasing Kaspersky Anti-Virus from a certified partner

Browse... Submit

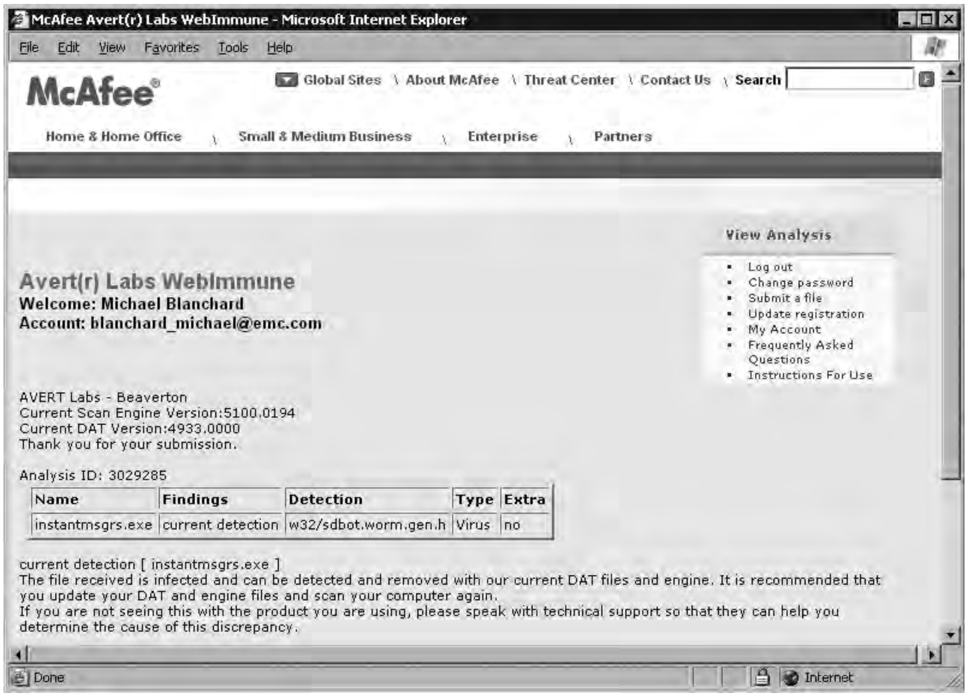
Scanned file: instantmsgrs.exe - Infected

instantmsgrs.exe - infected by **Backdoor.Win32.Wootbot.u**

Statistics:

Known viruses:	256586	Updated:	07-01-2007
File size (Kb):	148	Virus bodies:	1
Files:	1	Warnings:	0
Archives:	0	Suspicious:	0

Figure 9.13 Results after Submitting *instantmsgrs.exe* to McAfee AVERT Labs Submission Site, www.webimmune.net



McAfee Avert(r) Labs WebImmune - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Global Sites \ About McAfee \ Threat Center \ Contact Us \ Search

Home & Home Office \ Small & Medium Business \ Enterprise \ Partners

Avert(r) Labs WebImmune
Welcome: Michael Blanchard
Account: blanchard_michael@emc.com

View Analysis

- Log out
- Change password
- Submit a file
- Update registration
- My Account
- Frequently Asked Questions
- Instructions For Use

```
AVERT Labs - Beaverton
Current Scan Engine Version:5100.0194
Current DAT Version:4933.0000
Thank you for your submission.

Analysis ID: 3029285
```

Name	Findings	Detection	Type	Extra
instantmsgrs.exe	current detection	w32/sdbot.worm.gen.h	Virus	no

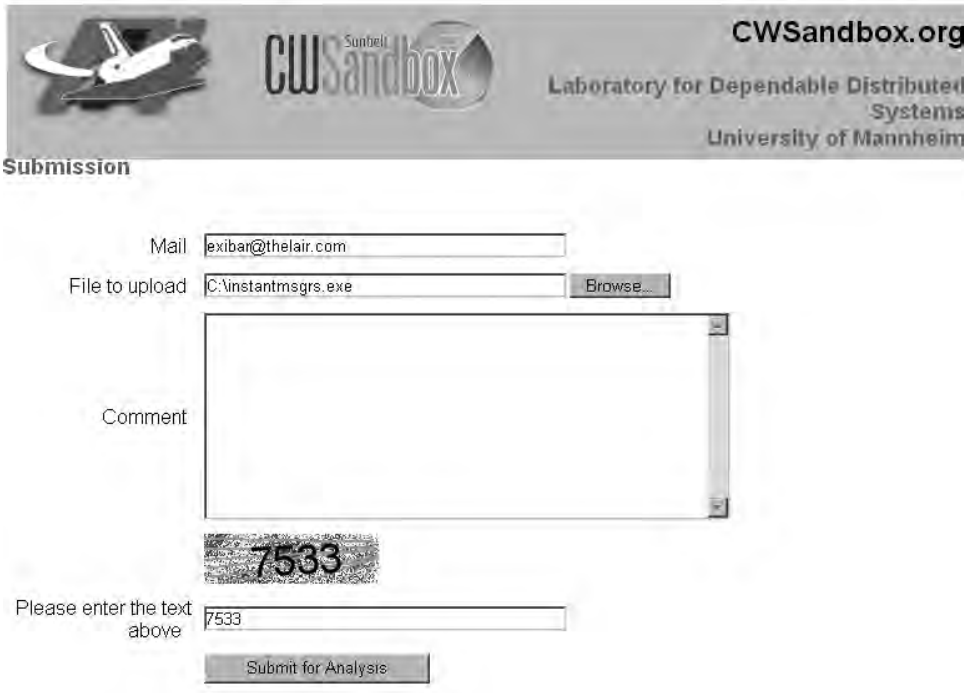
current detection [instantmsgrs.exe]
The file received is infected and can be detected and removed with our current DAT files and engine. It is recommended that you update your DAT and engine files and scan your computer again.
If you are not seeing this with the product you are using, please speak with technical support so that they can help you determine the cause of this discrepancy.

Done | Internet

Using an Online Malware Inspection Sandbox

There are a few sites at the time of this writing that accept malware samples. These sites will do a fairly comprehensive "sandbox" analysis for you, and will e-mail you the results. These sites are extremely useful when you find that you have a brand new malicious program or variant, one that none of the AV vendors can detect and clean at the time of discovery. These sandbox analysis reports enable you to put quick countermeasures in place, even before you're able to run your sample on your own test machine (known as a "goat" or "sacrificial goat" machine). (More on using a goat machine later in this chapter.)

Figure 9.14 CWSandbox's Submission Page Found at www.cwsandbox.org/

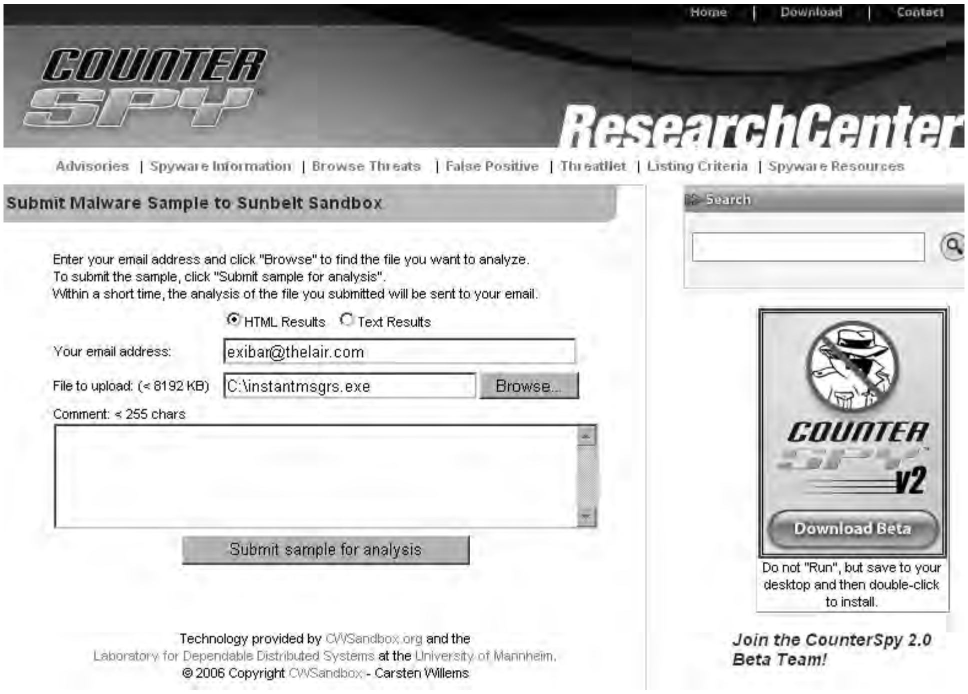


Mail: exibar@thelair.com
File to upload: C:\instantmsgrs.exe
Comment:
7533
Please enter the text above: 7533

There are three such Sandbox sites available to the public, free of charge, at the time of this writing. Figure 9.14 and Figure 9.15 show two of these sites. They use the same back-end engine and produce the same results. The first site, CWSandbox (www.cwsandbox.org/), will only send you the malware analysis report in eXtensible

Markup Language (XML) format, which needs to be fed into an XML processor in order to be easily readable. The second, CounterSpy's Sunbelt sandbox (http://research.sunbelt-software.com/submit.aspx), will send you the analysis report in a choice of Hypertext Markup Language (HTML) or plain text.

Figure 9.15 Counter Spy's Research Submission Page Found at http://research.sunbelt-software.com/submit.aspx


Home | Download | Contact
COUNTER SPY
ResearchCenter
Advisories | Spyware Information | Browse Threats | False Positive | Threatnet | Listing Criteria | Spyware Resources
Search
COUNTER SPY v2
Download Beta
Join the CounterSpy 2.0 Beta Team!

Submit Malware Sample to Sunbelt Sandbox

Enter your email address and click "Browse" to find the file you want to analyze.
To submit the sample, click "Submit sample for analysis".
Within a short time, the analysis of the file you submitted will be sent to your email.

[✓ HTML Results] Text Results

Your email address: exibar@thelair.com

File to upload: (< 8192 KB) C:\instantmsgrs.exe [Browse...]

Comment: < 255 chars

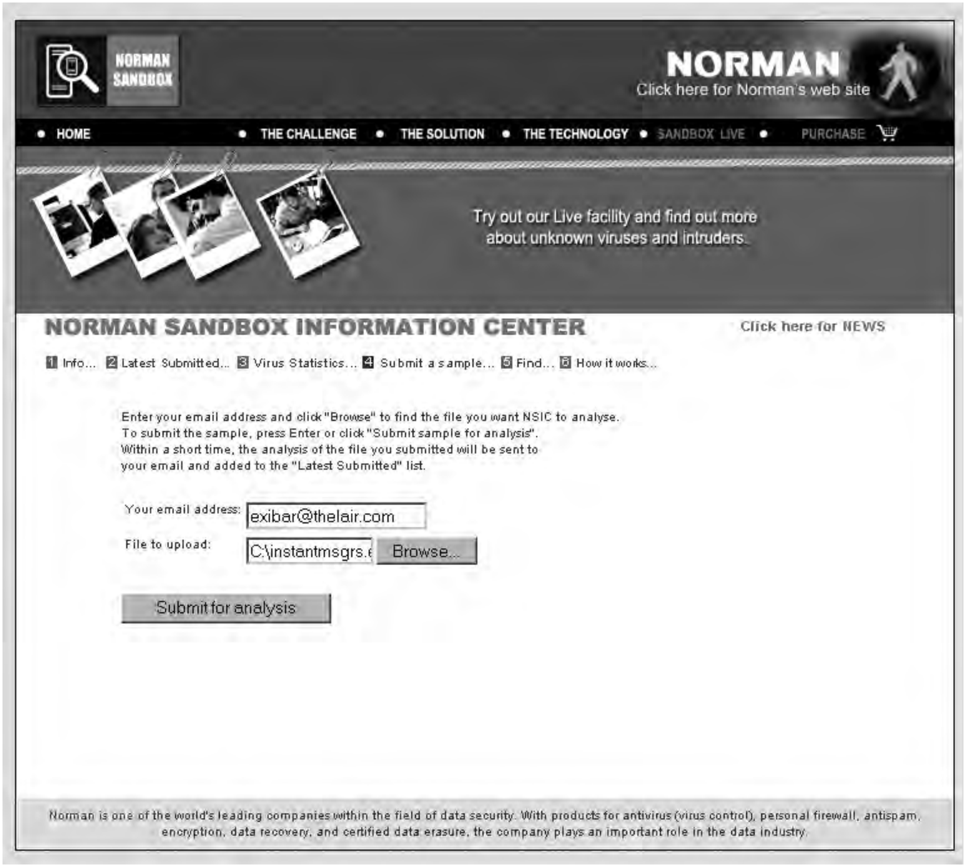
[Submit sample for analysis]

Do not "Run", but save to your desktop and then double-click to install.

Technology provided by CWSandbox.org and the Laboratory for Dependable Distributed Systems at the University of Mannheim.
© 2006 Copyright CWSandbox - Carsten Willems

The third Sandbox, shown in Figure 9.16, is Norman AV' sandbox (http://sandbox.norman.no/live.html). Norman Antivirus' sandbox will send you the results in text format only.

Figure 9.16 Norman Sandbox Submission Page, Found Here: http://sandbox.norman.no/live.html


NORMAN SANDBOX
NORMAN
Click here for Norman's web site
• HOME • THE CHALLENGE • THE SOLUTION • THE TECHNOLOGY • SANDBOX LIVE • PURCHASE
Try out our Live facility and find out more about unknown viruses and intruders.
NORMAN SANDBOX INFORMATION CENTER
Click here for NEWS
1 Info... 2 Latest Submitted... 3 Virus Statistics... 4 Submit a sample... 5 Find... 6 How it works...
Browse...
Submit for analysis

Enter your email address and click "Browse" to find the file you want NSIC to analyse.
To submit the sample, press Enter or click "Submit sample for analysis".
Within a short time, the analysis of the file you submitted will be sent to
your email and added to the "Latest Submitted" list.

Your email address: exibar@thelair.com

File to upload: C:\instantmsgrs.e

Norman is one of the world's leading companies within the field of data security. With products for antivirus (virus control), personal firewall, antispam, encryption, data recovery, and certified data erasure, the company plays an important role in the data industry.

Out of the three sandboxes, I prefer the results from CounterSpy's Sunbelt Sandbox. I feel it is the most usable and provides the most information without having to process an XML file. CounterSpy's Sunbelt Sandbox and CWSandbox both provide a very in-depth analysis on the malware that is submitted to those sites. Although Norman's sandbox provides very good analysis on the initial file that is submitted, it does not appear to process any files that are created by that initial file. So let's say that the initial file you submit creates two files and executes them on your system. CounterSpy and CWSandbox will process those two other files as well as your original sample. This can provide a huge benefit when you're trying to control an outbreak situation.

I've included samples of a CounterSpy's Sunbelt Sandbox report and a Norman Sandbox report. Within these samples you can clearly see that there is more information provided by the Counterspy report over the Norman report. I have not included a sample report from CWSandbox, as it is essentially the same report as the one from Counterspy, but in XML format. It's also worth noting that the sample report from CounterSpy is only an extract from the entire report, which is nearly seven full pages long! Within the CounterSpy report, you can clearly see the registry keys that are created, any mutexes that are created, any network connections that are created or used, and many other useful pieces of information. Everything provided in the reports from the Sandbox Web sites can be of considerable value when you're trying to prevent a massive outbreak within your enterprise.

Here is a partial sample report from CounterSpy Sunbelt Sandbox (**http://www.cwsandbox.org/**) received after submitting "instantmsgrs.exe," an SDBOT variant.

Analysis Summary:

Analysis Date	1/7/2007 4:57:43 AM
Sandbox Version	1.106
Filename	279ea1a420c31a8a0be9c9bc305c2b59.exe

Technical Details:

The following process was started by process: 1

Analysis Number	2
Parent ID	1
Process ID	1424
Filename	C:\WINDOWS\system32\instantmsgrs.exe -bai c:\279ea1a420c31a8a0be9c9bc305c2b59.exe
Filesize	151552 bytes
MD5	279ea1a420c31a8a0be9c9bc305c2b59
Start Reason	CreateProcess
Termination Reason	Timeout
Start Time	00:04.844
Stop Time	01:00.344
Detection	Trojan - W32/Sdbot.EEP (Authentium Command Antivirus - EngVer: 4.92.123.35 - SigVer: 20061222 35)
	Infected - Generic.Sdbot.B2963C9B (BitDefender Antivirus - EngVer: 7.0.0.2311 - SigVer: 7.10647)
	Backdoor - Backdoor.Win32.IRCBot.n (CounterSpy - EngVer: 2.1.628.0 - SigVer: 469)
	Infected -Backdoor:Win32/Sdbot!2111 (Microsoft Malware Protection -EngVer: 1.1.1904.0 - SigVer: Tue Dec 26 01:26:26 2006)
	Virus - W32.Spybot.Worm (Norton AntiVirus - EngVer: 20061.3.0.12 - SigVer: 20061226 13:19:02)

	Loaded DLLs
DLL-Handling	C:\WINDOWS\system32\instantmsgrs.exe
	C:\WINDOWS\system32\ntdll.dll
	C:\WINDOWS\system32\kernel32.dll
	C:\WINDOWS\system32\user32.dll
	C:\WINDOWS\system32\GDI32.dll
	C:\WINDOWS\system32\advapi32.dll
	C:\WINDOWS\system32\RPCRT4.dll
	C:\WINDOWS\system32\oleaut32.dll
	C:\WINDOWS\system32\msvcrt.dll
	C:\WINDOWS\system32\ole32.dll
	C:\WINDOWS\system32\comctl32.dll
	C:\WINDOWS\system32\wsock32.dll
	C:\WINDOWS\system32\WS2_32.dll
	C:\WINDOWS\system32\WS2HELP.dll
	C:\WINDOWS\system32\pstorec.dll
	C:\WINDOWS\system32\ATL.DLL
	C:\WINDOWS\system32\Wship6.dll
	C:\WINDOWS\system32\Secur32.dll
	user32.dll
	gdi32.dll
	shlwapi.dll
	MSVCRT.dll
	MSVCP60.dll
	iphlpapi.dll
	KERNEL32.dll
	USER32.dll
	ADVAPI32.dll
	SHELL32.dll
	WS2_32.dll
	MPR.dll
	PSAPI.DLL
	DNSAPI.dll
	KERNEL32.DLL
	kernel32.dll

Filesystem	C:\WINDOWS\system32\mswsock.dll hnetcfg.dll C:\WINDOWS\System32\wshtcpip.dll C:\WINDOWS\System32\mswsock.dll C:\WINDOWS\System32\winrnr.dll rasadhlp.dll **New Files** \Device\Tcp - Stored: \Device\Ip - Stored: \Device\Ip - Stored: \Device\RasAcd - Stored: **Opened Files** \\.\Ip - Stored: \\.\pipe\net\NtControlPipe8 - Stored: **Deleted Files** c:\279ea1a420c31a8a0be9c9bc305c2b59.exe **Chronological order** Create/Open File: \Device\Tcp (OPEN_ALWAYS) Create/Open File: \Device\Ip (OPEN_ALWAYS) Create/Open File: \Device\Ip (OPEN_ALWAYS) Open File: \\.\Ip (OPEN_EXISTING) Delete File: c:\279ea1a420c31a8a0be9c9bc305c2b59.exe Create/Open File: \Device\RasAcd (OPEN_ALWAYS) Open File: \\.\pipe\net\NtControlPipe8 (OPEN_EXISTING)
Registry	**Changes** HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run "mousedrive.exe" = instantmsgrs.exe HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce "mousedrive.exe" = instantmsgrs.exe HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run "mousedrive.exe" = instantmsgrs.exe HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce "mousedrive.exe" = instantmsgrs.exe HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices "mousedrive.exe" = instantmsgrs.exe

	Reads
	HKEY_LOCAL_MACHINE\Software\Microsoft\Rpc\SecurityService “DefaultAuthLevel”
	HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess “Start”
Process Management	Enum Processes
	Enum Modules - Target PID: (892)
	Enum Modules - Target PID: (1688)
	Enum Modules - Target PID: (1796)
	Enum Modules - Target PID: (1100)
	Enum Modules - Target PID: (1108)
	Enum Modules - Target PID: (1148)
	Enum Modules - Target PID: (1424)
	Open Process - Filename () Target PID: (4)
	Open Process - Filename () Target PID: (596)
	Open Process - Filename () Target PID: (644)
	Open Process - Filename () Target PID: (668)
	Open Process - Filename () Target PID: (712)
	Open Process - Filename () Target PID: (728)
	Open Process - Filename () Target PID: (740)
	Open Process - Filename (C:\Program Files\Faronics\Deep Freeze\Install C-0\DF5Serv.exe) Target PID: (892)
	Open Process - Filename () Target PID: (936)
	Open Process - Filename () Target PID: (1036)
	Open Process - Filename () Target PID: (1124)
	Open Process - Filename () Target PID: (1176)
	Open Process - Filename () Target PID: (1336)
	Open Process - Filename () Target PID: (1524)
	Open Process - Filename (C:\WINDOWS\Explorer.EXE) Target PID: (1688)
	Open Process - Filename (C:\WINDOWS\system32\VTTimer.exe) Target PID: (1796)
	Open Process - Filename (C:\Program Files\Faronics\Deep Freeze\Install C-0_$Df\FrzState2k.exe) Target PID: (1100)
	Open Process - Filename (C:\WINDOWS\system32\rundll32.exe) Target PID: (1108)

	Open Process - Filename (C:\WINDOWS\system32\WgaTray.exe) Target PID: (1148)	
	Open Process - Filename () Target PID: (1236)	
Service Management	Open Service Manager - Name: "SCM"	
System Info	Get System Directory	
	DNS Lookup	
	Host Name	**IP Address**
Network	donna.teh1.com	
Activity	donna.teh1.com	192.168.1.105
	Opened listening TCP connection on port: 3626	
	Outgoing connection to remote server: 255.255.255.255 TCP port 6667	
	Outgoing connection to remote server: donna.teh1.com TCP port 6667	

Next, here is a report received from Norman Sandbox (http://sandbox.norman.no/live.html) after submitting "*instantmsgrs.exe*," a W32/WootBot (AKA: SdBot) variant:

```
Your message ID (for later reference): 20070114-565

Hello,

Thanks for taking the time to submit your samples to the Norman Sandbox
Information Center. Customer delight is our top priority at Norman. With that
in mind we have developed Sandbox Solutions for organizations that are committed
to speedy analysis and debugging.

Norman Sandbox Solutions give your organization the opportunity to analyze files
immediately in your own environment.

To find out how to bring the power of Norman Sandbox into your test environments
follow the links below.
Norman Sandbox Solutions

http://www.norman.com/Product/Sandbox-products/

Norman Sandbox Analyzer

http://www.norman.com/Product/Sandbox-products/Analyzer/

Norman Sandbox Analyzer Pro

http://www.norman.com/Product/Sandbox-products/Analyzer-pro/

Norman SandBox Reporter

http://www.norman.com/Product/Sandbox-products/Reporter/

instantmsgrs.exe : W32/SDBot.gen2 (Signature: W32/SDBot.GAV)
```

```
[ General information ]
   *File length: 151552 bytes.
   *MD5 hash: 279ea1a420c31a8a0be9c9bc305c2b59.

[ Security issues ]
   *Modified OS kernel function code.

(C) 2004-2006 Norman ASA. All Rights Reserved.

The material presented is distributed by Norman ASA as an information source only.

Sent by exibar@thelair.com. Received 14.Jan 2007 at 03.16 -processed 14.Jan 2007
at 03.18.
```

Tools and Traps

Signing Up for Norman Sandbox's Daily Digest

Even though the report from my submission to Norman Sandbox of the SDBOT variant, as shown above, is a little sparse, Norman has a wealth of information to share with qualified individuals. If you head out to http://sandbox.norman.no/live_2.html, you'll see that the Norman Sandbox shows you the last 30 or so files that have been submitted. Within those reports, all the Uniform Resource Locators (URLs) have been removed to prevent malicious use. The URLs have all been replaced with the word "REMOVED," which itself is a link to http://sandbox.norman.no/form.html.

If you follow that link, you will be presented with a sign-up form that will enable you to receive a daily digest of all the submissions that occurred during that day. You will have to be approved by them before you start receiving the daily digests, however.

The reports contained within the digest contain all the URLs, Internet IP's, and so forth to which the submissions attempt to open a connection. The reason that this is so valuable is that you can create some parsing scripts that can compare the daily digest URLs and IP addresses to your firewall logs. Any machine within your enterprise that is attempting to go out to any of those addresses or URLs, is surely controlled by malware!

Using Packet Analyzers to Gather Information

All of the tools previously mentioned in this section, enable you to get to a point where someone else can analyze the malware you've found in a sanitized environment. In the next section, we'll look at tools that you can use in your own enterprise environment. Ideally, you'll use them on a typical enterprise system; not an actual production machine but a "goat" machine.

A "sacrificial goat" machine or simply a "goat" machine is nothing more than a victim machine whose sole purpose is to be the victim machine on which you'll be actually running your sample. The purpose of this machine is simply to sacrifice its own integrity so you can gather as much threat data as possible in a safe environment, without fear that the sample might escape onto your production network.

This machine should not normally be connected to your production network; its only network connection should be directly to the Internet or to a 'faked' server providing DNS, for instance. This machine should very closely resemble your production machine setup, perhaps created using the same image process. It should contain all of the tools that you require to perform your testing, along with some way to copy your sample over to it, either via floppy disk or via a secure Universal Serial Bus (USB) flash drive.

Your goat machine doesn't even have to be an actual physical machine. Using software such as VMware's VMware Workstation (www.vmware.com) or Microsoft's Virtual PC (www.microsoft.com), you can have multiple, fully functional operating systems running on one host operating system simultaneously.

It's important to note that some malware writers have placed special checks in their code that will check whether it's being run in a virtual environment. If it is, it will stop running. So far, this behavior has not been much observed, but it is happening more often.

The use of a packet analyzer is one of the fundamentals techniques of malware analysis. A packet analyzer is a program that will enable you to look directly at information passing across the network to and from the machine on which it's running. Three of the most popular packet analyzers available today are tcpdump for UNIX-type machines, Windump for Windows platforms, and Wireshark (formerly known as Ethereal and available for both windows and UNIX.)

Before you can run any packet capture programs (and many of the more advanced tools coming up), you must have the correct drivers installed. In this case, you must have "packet capture libraries" installed. For Linux you must install Libcap. Windows platforms require the installation of WinPcap. Both installations are quick and simple, and the programs are available at the same location as tcpdump and windump (see below).

Tcpdump and Windump are basically the same application. Windump is a port of tcpdump from UNIX to Windows. The output from both programs and their command line switches are identical. TCPdump can be found at www.tcpdump.org, and WinPcap can be found at www.winpcap.org.

Typical tcpdump and windump command line options are discussed here. For the complete manual on tcpdump, please refer to the *man* pages found at www.rt.com/man/tcpdump.1.html. We've presented the most common command-line options and expressions below.

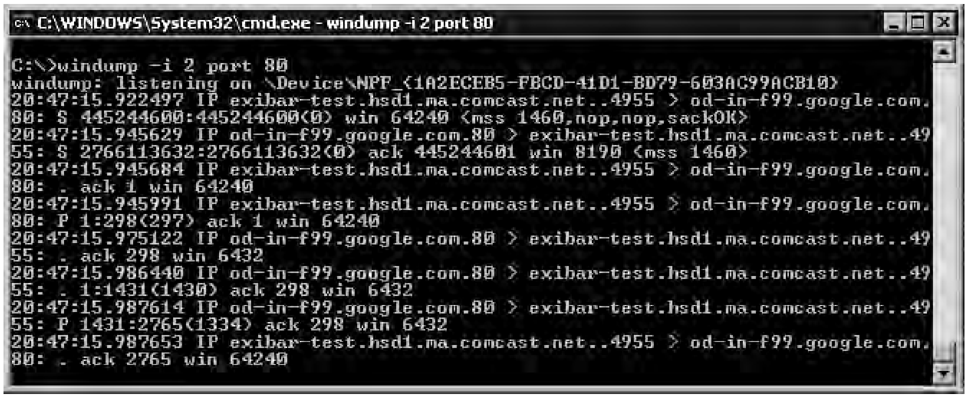
Results of Running *windump* at the Command Line to Show Proper Syntax Formatting

```
windump version 3.9.5, based on tcpdump version 3.9.5
WinPcap version 3.1 (packet.dll version 3, 1, 0, 27), based on libpcap
version 0.9[.x]
Usage: windump [-aAdDeflLnNOpqRStuUvxX] [ -B size ] [-c count]
[ -C file_size ][ -E algo:secret ] [ -F file ] [ -i interface ]
[ -M secret ] [ -r file ] [ -s snaplen ] [ -T type ] [ -w file]
[ -W filecount ] [ -y datalinktype ] [ -Z user ] [ expression ]
```

Here is a listing of tcpdump and windump common options and expressions

- ***-?*** List command line syntax options (see Figure 9.17).
- ***-c*** End capture after capturing *[count]* number of packets.
- ***-i*** Listen for packets on *[interface]*. Default is to use the lowest numbered interface. On Windows boxes, this is usually the dial-up adapter.
- ***-n*** Don't convert numbers to common names. IP addresses, ports, and so on, will remain in numeric format, and not be converted to common names. This speeds up some captures due to the lack of Domain Name Domain name system (DNS) lookups.
- ***-r*** Read from a packet capture file instead of from the network interface. Use the "*-w*" option to create a file.
- ***-v*** Verbose output.
- ***-vv*** More verbose output.
- ***-vvv*** Extremely verbose output.
- ***-w*** Write the capture to a file instead of displaying the packets on the screen.
- ***host*** Limit capture to this *[host]* only. *[Host]* can be either an IP address or a qualified domain name, and can be preceded with DST or SRC for Destination or Source host, respectively.
- ***port*** Limit capture to *[port]* only. Can also be preceded with a protocol type of UDP or TCP.

Figure 9.17 Windump Output Showing a Connection to www.google.com Using IE 6.0



```
C:\>windump -i 2 port 80
windump: listening on \Device\NPF_{1A2ECEB5-FBCD-41D1-BD79-603AC99ACB10}
20:47:15.922497 IP exibar-test.hsd1.ma.comcast.net..4955 > od-in-f99.google.com.
80: S 445244600:445244600(0) win 64240 <mss 1460,nop,nop,sackOK>
20:47:15.945629 IP od-in-f99.google.com.80 > exibar-test.hsd1.ma.comcast.net..49
55: S 2766113632:2766113632(0) ack 445244601 win 8190 <mss 1460>
20:47:15.945684 IP exibar-test.hsd1.ma.comcast.net..4955 > od-in-f99.google.com.
80: . ack 1 win 64240
20:47:15.945991 IP exibar-test.hsd1.ma.comcast.net..4955 > od-in-f99.google.com.
80: P 1:298(297) ack 1 win 64240
20:47:15.975122 IP od-in-f99.google.com.80 > exibar-test.hsd1.ma.comcast.net..49
55: . ack 298 win 6432
20:47:15.986440 IP od-in-f99.google.com.80 > exibar-test.hsd1.ma.comcast.net..49
55: . 1:1431(1430) ack 298 win 6432
20:47:15.987614 IP od-in-f99.google.com.80 > exibar-test.hsd1.ma.comcast.net..49
55: P 1431:2765(1334) ack 298 win 6432
20:47:15.987653 IP exibar-test.hsd1.ma.comcast.net..4955 > od-in-f99.google.com.
80: . ack 2765 win 64240
```

Don't let the plethora of options available with tcpdump and windump scare you. Using these two tools for malware analysis is really quite simple and basic. A typical use would be for capturing every packet to a file to be analyzed later, while gathering information of a piece of malware that you're analyzing.

In order to use tcpdump or windump, you'll first want to find out which interface you want to capture packets from. On a UNIX system, the command *ifconfig* will give you the information about the interfaces needed. On a Windows platform, you can run *windump –D* to will display results similar to this:

```
c:\>windump -D
1.\Device\NPF_GenericDialupAdapter (Generic dialup adapter)
2.\Device\NPF_{2F11EF65-4EDD-4FA6-B9A1-567C4AA8CD1C}
(VMware Virtual Ethernet Adapter)
3.\Device\NPF_{B5410B0A-C3B7-4A5A-82C2-B4EEFFBD5350}
(Marvell Gigabit Ethernet Controller (Microsoft's
Packet Scheduler) )
4.\Device\NPF_{8307E6E0-6B30-4387-94B8-45F212CFA29C}
(VMware Virtual Ethernet Adapter)
```

From the above sample output you can see that the "Generic Dialup Adapter" is listed first, then one of two VMware virtual adapters, while number three is the main network interface adapter from which we want to capture packets. The command line that you would use in this case is shown below. This would capture as much data as possible from that network interface adapter, and write it to a file for later analysis.

```
c:\>windump -i 3 -w c:\test.pcap
windump: listening on \Device\NPF_{B5410B0A-C3B7-4A5A-82C2-B4EEFFBD5350}

99 packets captured
99 packets received by filter
0 packets dropped by kernel
```

The above command is broken down as follows:

- ***windump*** Core command to start the packet capture
- ***-i 3*** Capture packets on Interface 3
- ***-w c:\test.pcap*** Don't display the packets on the screen. Instead, write them to file *"c:\test.pcap"* for later analysis.

As you can see from Figure 9.17, the usage of tcpdump and its output can be quite overwhelming. If you don't use the correct command-line options and expressions, you could miss key data. On the other hand, if you don't use any options or expressions and capture *all* the packets from an interface card, there is simply too much data to be able to go through in a timely manner. This leads me to consider the usage of Wireshark (formally known as Ethereal and freely available at http://www.wireshark.org). Wireshark will help organize the output of the capture in a fashion that makes it much easier to use.

Tools & Traps

Wireshark-infested Custard

Wireshark is freely available at www.wireshark.org. There is also a wealth of documentation at www.wireshark.org/docs/

Wireshark development was started back in 1998 by a man named Gerald Combs, originally under the name of Ethereal and under the terms of the GNU General Public License. Since 1998, hundreds of people have contributed to the project to make it the industry standard in packet capture applications. In 2006, the name Ethereal changed to Wireshark due to copyright and trademark issues (www.wireshark.org/faq.html -q1.2; trends.newsforge.com/article.pl?sid=06/06/09/1349255&from=rss), but improvements continue to be made. Wireshark also remains one of the most widely used packet capture applications available and is still free for use under the GNU General Public License (GPL).

We don't have space to do more than hint at the capabilities and functionality of Wireshark here. However, Syngress have published a book by Angela Orebaugh *et al* called "Wireshark and Ethereal: Network Protocol Analyzer Toolkit," which goes into considerable detail.

More details at www.syngress.com/catalog/?pid=3770.

As well as performing packet captures on its own, Wireshark has the ability to read all PCAP-formatted output files including tcpdump, windump, the Snort Intrusion Detection System (IDS), and many other applications that produce output in PCAP format. Wireshark has a nice GUI front-end and includes many built-in capabilities that make it easy and intuitive to use for both producing and analyzing packet captures. Figure 9.18 shows a connection to www.google.com in action, using Internet Explorer 6.0. You can easily see the Synchronous (SYN), Synchronous/Acknowledge (SYN-ACK), Acknowledge (ACK) of the three-way TCP handshake on lines 4, 5, and 6. Then the connection out to www.google.com via Hypertext Transfer Protocol (HTTP) on line 7 is completed and the Google homepage is displayed.

Tools and Traps

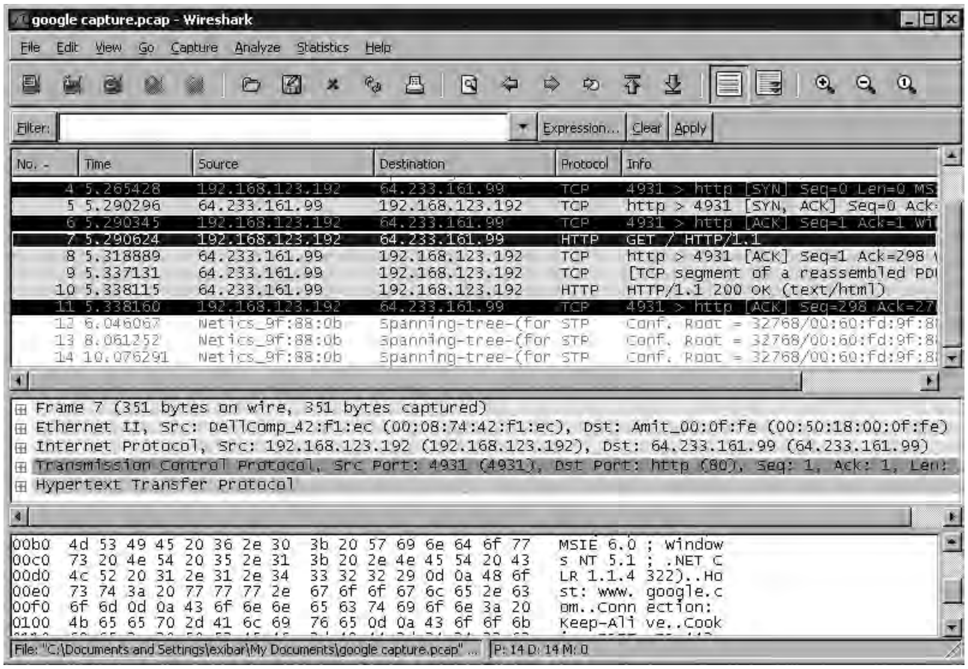
Understanding the Three-way Handshake

The three-way handshake describes the method TCP uses to establish a connection. A server opens a port up for connections (a "passive open"). When the server is "listening," a client can initiate an "active open" by sending a SYN to the server. The active open is performed by sending a SYN to the server.

Now the server acknowledges the SYN request with a SYN-ACK.

Finally, the client responds to the server's acknowledgement with an answering acknowledgement simply called an ACK. It could be said that once the client and the server have received an acknowledgement, the connection is formally open.

Figure 9.18 Wireshark Captures the Complete Connection to www.google.com


google capture.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. -  Time       Source              Destination          Protocol  Info
    4 5.265428   192.168.123.192     64.233.161.99        TCP       4931 > http [SYN] Seq=0 Len=0 MS
    5 5.290296   64.233.161.99       192.168.123.192      TCP       http > 4931 [SYN, ACK] Seq=0 Ack
    6 5.290345   192.168.123.192     64.233.161.99        TCP       4931 > http [ACK] Seq=1 Ack=1 Wi
    7 5.290624   192.168.123.192     64.233.161.99        HTTP      GET / HTTP/1.1
    8 5.318889   64.233.161.99       192.168.123.192      TCP       http > 4931 [ACK] Seq=1 Ack=298
    9 5.337131   64.233.161.99       192.168.123.192      TCP       [TCP segment of a reassembled PD
   10 5.338115   64.233.161.99       192.168.123.192      HTTP      HTTP/1.1 200 OK (text/html)
   11 5.338160   192.168.123.192     64.233.161.99        TCP       4931 > http [ACK] Seq=298 Ack=27
   12 6.046067   Netics_9f:88:0b     Spanning-tree-(for   STP       Conf. Root = 32768/00:60:fd:9f:8
   13 8.061252   Netics_9f:88:0b     Spanning-tree-(for   STP       Conf. Root = 32768/00:60:fd:9f:8
   14 10.076291  Netics_9f:88:0b     Spanning-tree-(for   STP       Conf. Root = 32768/00:60:fd:9f:8

⊞ Frame 7 (351 bytes on wire, 351 bytes captured)
⊞ Ethernet II, Src: DellComp_42:f1:ec (00:08:74:42:f1:ec), Dst: Amit_00:0f:fe (00:50:18:00:0f:fe)
⊞ Internet Protocol, Src: 192.168.123.192 (192.168.123.192), Dst: 64.233.161.99 (64.233.161.99)
⊞ Transmission Control Protocol, Src Port: 4931 (4931), Dst Port: http (80), Seq: 1, Ack: 1, Len:
⊞ Hypertext Transfer Protocol

00b0  4d 53 49 45 20 36 2e 30  3b 20 57 69 6e 64 6f 77   MSIE 6.0 ; Window
00c0  73 20 4e 54 20 35 2e 31  3b 20 2e 4e 45 54 20 43   s NT 5.1 ; .NET C
00d0  4c 52 20 31 2e 31 2e 34  33 32 32 29 0d 0a 48 6f   LR 1.1.4 322)..Ho
00e0  73 74 3a 20 77 77 77 2e  67 6f 6f 67 6c 65 2e 63   st: www. google.c
00f0  6f 6d 0d 0a 43 6f 6e 6e  65 63 74 69 6f 6e 3a 20   om..Conn ection:
0100  4b 65 65 70 2d 41 6c 69  76 65 0d 0a 43 6f 6f 6b   Keep-Ali ve..Cook

File: "C:\Documents and Settings\exibar\My Documents\google capture.pcap" ...   P: 14 D: 14 M: 0
```

Examining Your Malware Sample with Executable Inspection Tools

One of my standard procedures when I get my malware sample for analysis is to run it against another www.sysinternals.com program, called *Strings.exe*. This program will go through the file and (by default) look for three or more consecutive Unicode and/or American Standard Code for Information Interchange (ASCII) characters in a row. This isn't an analysis procedure that I focus too much time on (and Strings is considered later in more detail in the section on advanced forensics), but may be well worth the time spent in a lot of cases. Sometimes Internet Relay Chat (IRC) server names, IP addresses, and other useful information can be gathered by looking through the malware executable itself.

Available Command-line Options for STRINGS.EXE from www.sysinternals.com

```
Strings v2.1
Copyright (C) 1999-2003 Mark Russinovich
Systems Internals -www.sysinternals.com
usage: strings [-s] [-n length] [-a] [-u] [-q] <file or directory>
-s Recurse subdirectories
-n Minimum string length (default is 3)
-a Ascii-only search (Unicode and Ascii is default)
-u Unicode-only search (Unicode and Ascii is default)
-q Quiet (no banner)
```

Up to this point, all the tools that we've discussed have been either open source or freeware tools. There are a few commercial tools that we use on a regular basis, and one such tool is from FreeSpace Internet Security (http://www.freespaceinternetsecurity.com), and is called VIP Utility. VIP Utility was written back in 2003, by Lixin Lu, who then went ahead and founded Freespace Internet Security the same year. VIP Utility produces results that are very similar to those given by the previously mentioned Web inspection sites. The benefit of using this utility is that it is run in your own environment, and you get immediate results, whereas there is some delay waiting for results e-mailed to you from Web inspection sites.

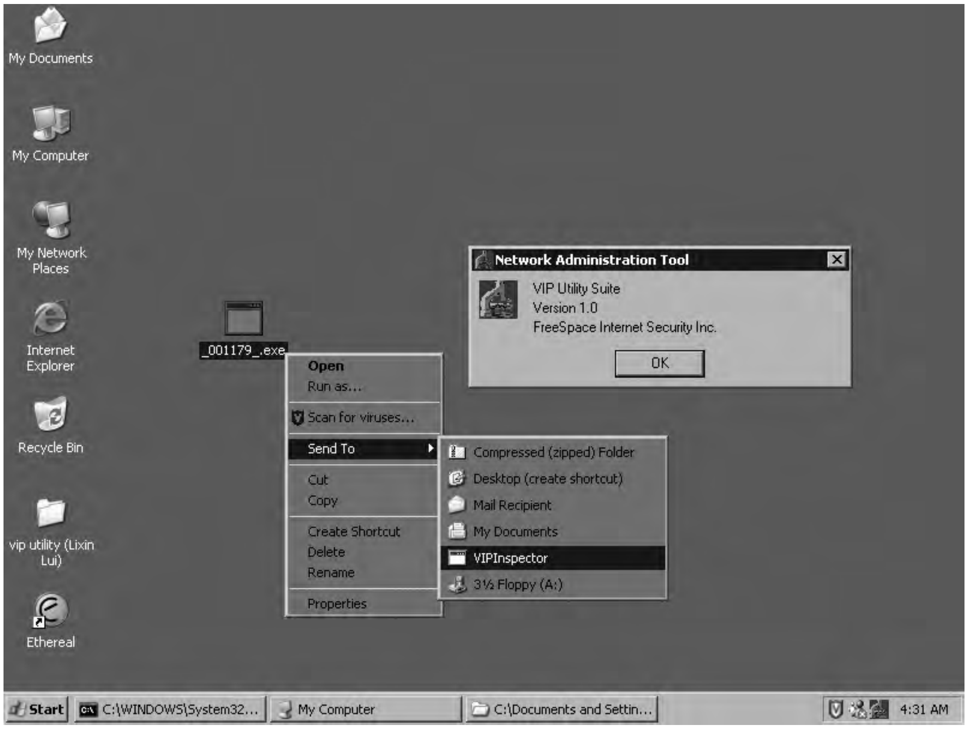
VIP Utility is basically an open sandbox that monitors the executable that is passed through it, using a combination of heuristics and behavior monitoring.

The reason that I say it's an "open" sandbox is that in some cases, VIP Utility will allow the executable to create actual files or folders on the host system. VIP Utility does a remarkable job in terms of "rolling back" anything that the executable does to the host system in this regard. In the two years or so that this author has been using VIP Utility to analyze malware code, I have never once had any problems on the host. Nonetheless, we do not recommend running VIP Utility, or any malware analysis tools, on a production machine. Always use a goat machine or a goat virtual machine for analysis purposes.

Using VIP Utility is very simple. After installation and registration of the product, you'll notice if you right-click on an executable, as shown in Figure 9.19, that there is a new "send-to" menu option. This is the command that you'll use to send an executable through to the VIP Inspector, which is the core inspection module of VIP Utility. Once you do this, you'll see a DOS box open up while VIP Inspector is analyzing the executable. When this DOS box closes, your analysis is complete.

Once the VIP Inspector has completed its analysis, it will produce a report and store that report in the "reports" directory, which resides inside the installation directory for VIP Utility. The reports are very straightforward and easy to read. A sample report is shown in Figure 9.20.

Figure 9.19 Sending a File Over to VIP Utility Using the "Send To" Right-click Menu



VIP Utility Suite
Version 1.0
FreeSpace Internet Security Inc.

Figure 9.20 Sample Output From VIP Utility

```
File name: scpr32b.exe
Warning Level: 10
Viral Type: Infected by a new variant of Unknown virus.
Activities:

1. Tries to set unhandled exception filter.
2. Tries to start up a Winsock to communicate outside.
3. Tries to delete file autorun.inf.
4. Tries to create a mutex named as 3676C64A-W454-122E-BFC6-
083C2BF4S551.
5. Tries to query Windows folder.
6. Tries to copy itself onto the system as
C:\WINDOWS\System\CSRSS.EXE.
7. Tries to open registry
Software\Microsoft\Windows\CurrentVersion\Run.
8. Tries to set auto run for the newly dropped executable file:
.svchost.
9. Tries to set unhandled exception filter.
```

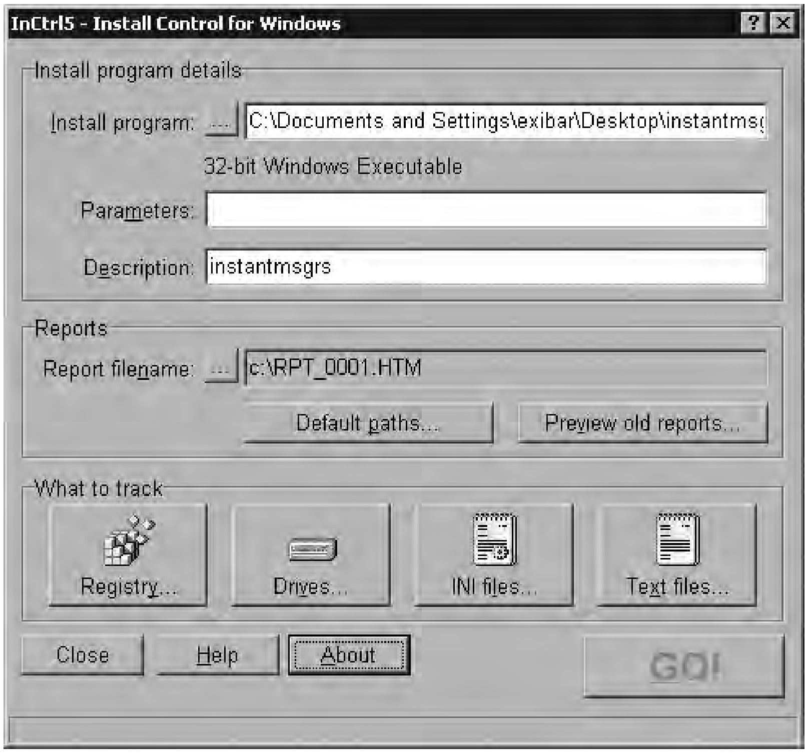
Another extremely useful commercial tool is InCtrl5, from PC Magazine, and this can be found at http://www.pcmag.com/article2/0,4149,9882,00.asp. The cost for this product is negligible, and is well worth paying, given everything it does. InCtrl5 basically takes a snapshot of all files, registry keys, text files, and INI files. It takes these snapshots before and after the file is executed, compares the differences, and generates a report based upon those differences.

WARNING

It's very important to note that when using InCtrl5 to track changes when a file executes, the file is actually executed on that machine, live. It is not run in a sandbox environment, as is the case with VIP Utility. If you use InCtrl5 to analyze an actual virus or worm, that machine will be infected after the execution is complete. This is where using VMware's "VMware Workstation" to host your virtual goat machine comes in handy. VMware has a "snapshot" capability that allows you to revert back to a previous machine state before the infection took place, at the click of a button.

Figure 9.21 shows the GUI for InCtrl5. It is very intuitive to use, and the default settings work in 99 percent of all situations that you are likely to encounter. The only feature that we would add would be under the "Text Files" section. By default, the program will only track inside *autoexec.nt* and *config.nt*. We would recommend that you add the *c:\windows\system32\drivers\etc\hosts* file to this listing as well, as many malicious programs will attempt to modify this file.

Figure 9.21 The Main GUI Front End for InCtrl5 from PCMagazine


InCtrl5 - Install Control for Windows

Install program details

Install program: C:\Documents and Settings\exibar\Desktop\instantmsg

32-bit Windows Executable

Parameters:

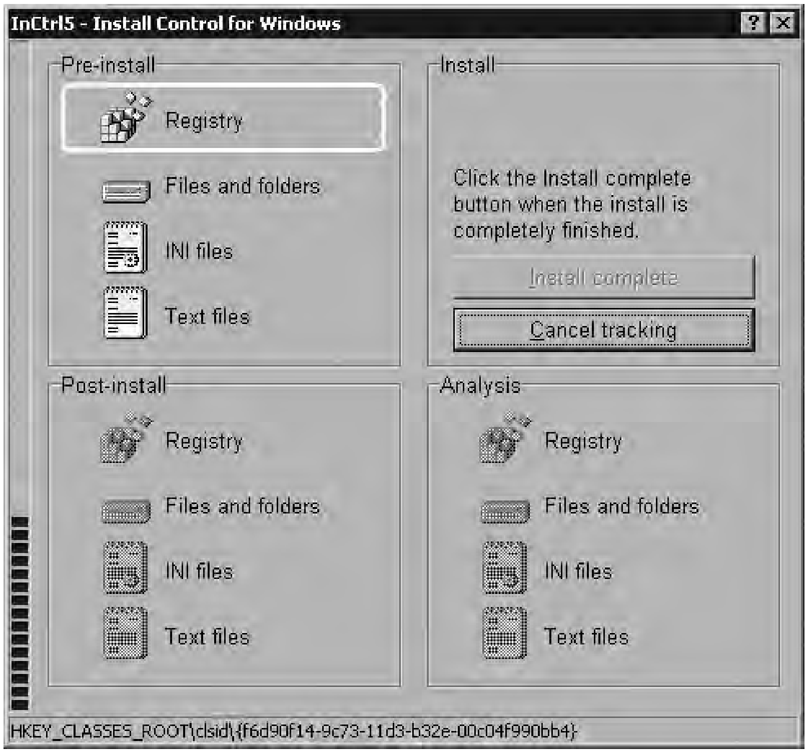
Description: instantmsgrs

Reports

Report filename: c:\RPT_0001.HTM

Figure 9.22 shows InCtrl5 in action as it goes through the machine, analyzing the state of the machine before the executable is run.

Figure 9.22 InCtrl5 Collects the Data Before Launching the Executable

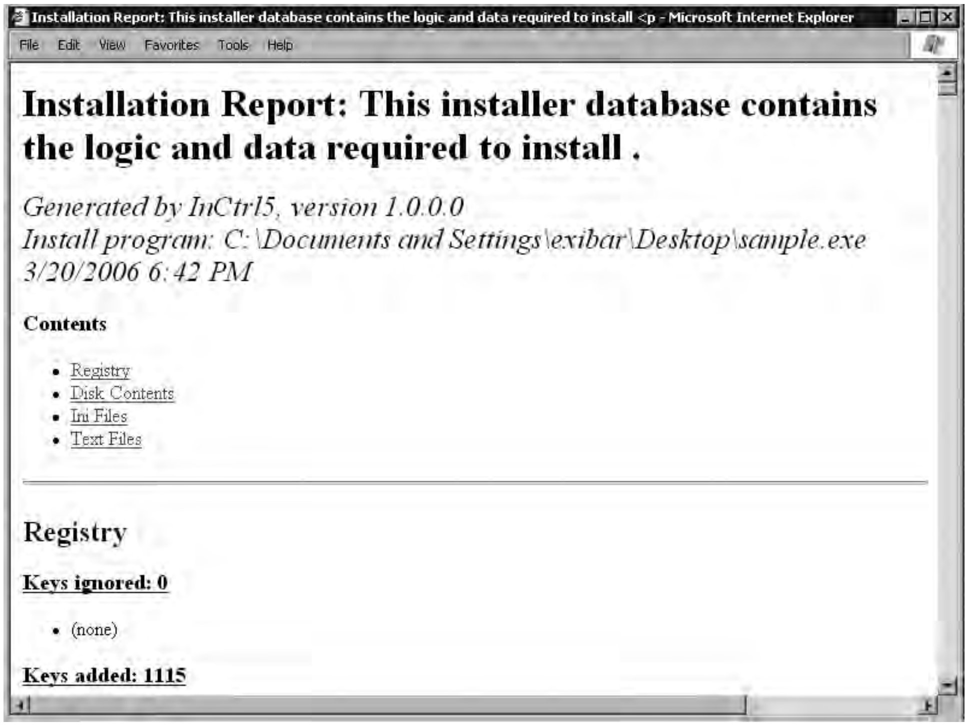


Click the Install complete button when the install is completely finished.

HKEY_CLASSES_ROOT\clsid\{f6d90f14-9c73-11d3-b32e-00c04f990bb4}

Figure 9.23 shows an extract from a sample report from InCtrl5. Notice that the executable *sample.exe* added 1,115 new registry keys!

Figure 9.23 Sample Report After Running InCtrl5


Installation Report: This installer database contains the logic and data required to install <p - Microsoft Internet Explorer
File Edit View Favorites Tools Help

Installation Report: This installer database contains the logic and data required to install .

Generated by InCtrl5, version 1.0.0.0
Install program: C:\Documents and Settings\exibar\Desktop\sample.exe
3/20/2006 6:42 PM

Contents

- Registry
- Disk Contents
- Ini Files
- Text Files

Registry

Keys ignored: 0

- (none)

Keys added: 1115

TIP

In order to minimize the number of times that a piece of malware code is executed on a live system, we'll always have WireShark up and running before we launch the executable with InCtrl5. This way, we'll be able to gather data on any network activity that the executable is generating, along with the machine changes that it's causing. By doing this, we only have to physically run our sample once, and we gather all the data that we need to analyze it properly.

Using Vulnerability Assessment and Port Scanning Tools

Many pieces of malware code today make use of certain vulnerabilities in order to propagate. Alternatively, some of the more notorious malicious programs will open up a port on your machine in order to allow others to connect to it. Part of the process of analysis requires you to be able to search out those vulnerable machines on your network. After all, it's not good enough just knowing that your sample malware opens up port TCP/1337 on your computer system if you can't actually hunt down vulnerable and infected systems and clean them up, using the data that you gained as a result of the analysis. This next section will give you a feeling for the most popular (and free!) vulnerability assessment and port scanning tools available today.

The first such tool is a port-scanning tool, probably the most well known of its type. Its presence in movies such as *The Listening,* and recently and much more notably in *The Matrix Reloaded* have elevated this tool way beyond cult status. This tool is *Nmap* from www.insecure.org. Fyodor, the creator of Nmap, set the standard in port scanners when its source code made its first appearance in Phrack Magazine, volume 7, Issue 51 back in September 1997. Since that fateful day in September 1997, there have been literally hundreds of improvements including an optional GUI front-end version called *Nmapfe*, also available from www.insecure.org.

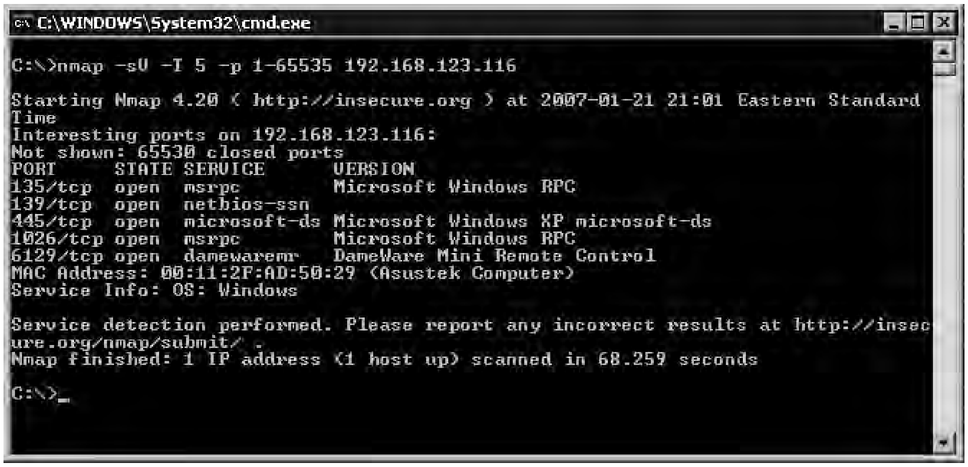
There are many command-line options that can be used with Nmap; I've listed the most useful for our purposes below. For a complete listing of command-line options for use with Nmap, please refer to www.insecure.org and the help pages found there.

Most Common Command Line Switches for Scanning with Nmap

```
Usage: nmap [Scan Type(s)] [Options] {target machine or range}

-P0: Do not ping the target machine first, simply check for open ports
-n: Don't resolve hostnames during scan. Can speed up performance
significantly
-sT: Perform a full TCP connect scan
-sU: Scan UDP ports only
-p <port ranges>: Scan these ports only. ("-p 1-65535" will scan
all TCP ports)
-sV: Attempt to determine what service and version is running on
the open ports found
-A: Attempt to determine remote host's Operating System and version of OS
-T[0-5]: Timing of scan, 0 is slower and stealthier, 5 is faster
and noisier
-v: verbose output
-vv: very verbose output
```

Figure 9.24 shows Nmap in action after scanning a machine for open ports. You can see how intuitive the output from Nmap is and how well laid out it is. No wonder Nmap has become the standard in port scanners!

Figure 9.24 Output After Running a Port Scan Using Nmap



```
C:\>nmap -sU -T 5 -p 1-65535 192.168.123.116

Starting Nmap 4.20 ( http://insecure.org ) at 2007-01-21 21:01 Eastern Standard
Time
Interesting ports on 192.168.123.116:
Not shown: 65530 closed ports
PORT     STATE SERVICE      VERSION
135/tcp  open  msrpc        Microsoft Windows RPC
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds Microsoft Windows XP microsoft-ds
1026/tcp open  msrpc        Microsoft Windows RPC
6129/tcp open  damewaremr   DameWare Mini Remote Control
MAC Address: 00:11:2F:AD:50:29 (Asustek Computer)
Service Info: OS: Windows

Service detection performed. Please report any incorrect results at http://insec
ure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 68.259 seconds

C:\>_
```

Nmap's usual command-line options, shown in Figure 9.24, will suffice for most of the port scanning activities you're likely to need in your enterprise. But there may be times when you require a more aggressive or a more passive scan. My advice is to download a copy from www.insecure.org, print out the *man* page for it, and go through the options that Nmap provides while testing on a spare machine. This way you'll be able to see what each option does, and how they all affect the scan output.

If you'd prefer a port scanner that offers a GUI, and don't want to use Nmapfe, then Superscan is the port scanner for you. Superscan was created by the fellows over at Foundstone, and is free to download from their Web site at http://www.foundstone.com. This is a port scanner that performs basically the same duties as Nmap, but in a nice, easy to read, graphical format. Superscan has an advantage over Nmap in that it has a GUI for all scan options. You simply go through each available tab, and check off the options that you want Superscan to look for while it scans. Although the default options will pick up most items of interest, there are a couple of items that I change when using Superscan to perform a port scan. These closely resemble the options we'll use when running Nmap from the command line.

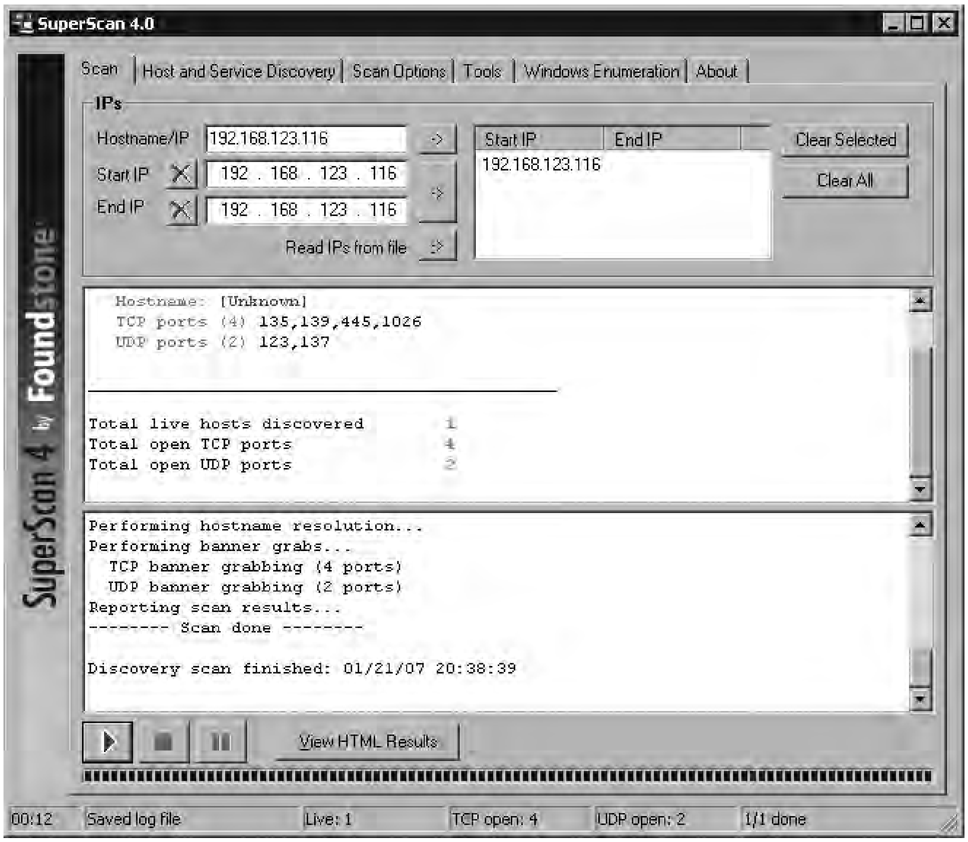
By default, Superscan doesn't scan all TCP and UDP ports (see www.skullbox.net/tcpudp.php for a succinct and mildly humorous explanation of the differences between TCP and UDP. When looking for unknown malware that has opened up a new port on the target machine, it's important to check all of these on the suspect machine, unless you were able to determine exactly what port was opened by analyzing a sample with the tools previously mentioned.

If you're scanning an entire subnet or subnets, you might also set all Host Discovery options to active, by placing a checkmark next to each one. By doing this, you'll stand a better chance of detecting that a remote machine is live and on the network, even if it's not returning ICMP pings.

The final option that you might change from the default can cause the scan to take a bit longer to complete if you're scanning many class C or larger network subnets. Nevertheless, consider turning on a full TCP connect scan. With this option turned on, Superscan will perform a full three-way handshake with each port before it attempts to gather information about that port. You can certainly see why this would increase the scan time compared to a simple SYN TCP scan. To compensate for the time increase waiting for the full three-way handshake, it is possible to decrease the TCP port scan timeout from the default of 4000 ms, down to 1000 ms. The danger with decreasing the timeout is that on slower networks, or networks where there are many hops between the scanning machine and the subnets being scanned, you could exceed that timeout and Superscan would assume that the port is not active. In reality, the port could be active in this scenario and simply taking longer than 1000 ms for the replies to reach your machine.

Figure 9.25 shows you a view of the main GUI page. Here there is only one item that has to be entered, the hostname or the IP address of the machine that you wish to scan. If you wish to scan a range or multiple ranges, this is where you'll enter that information as well. To start the scan, click on the little blue arrow button. During the scan you can watch scanning progress in the two large windows. Once the scan is complete, click on the large **View HTML Results** button and a report similar to the one shown in Figure 9.26 will open up in your default Internet browser.

Figure 9.25 Main GUI Interface Page of Superscan 4.0



Hostname/IP 192.168.123.116
Start IP 192 . 168 . 123 . 116
End IP 192 . 168 . 123 . 116

Start IP	End IP
192.168.123.116	

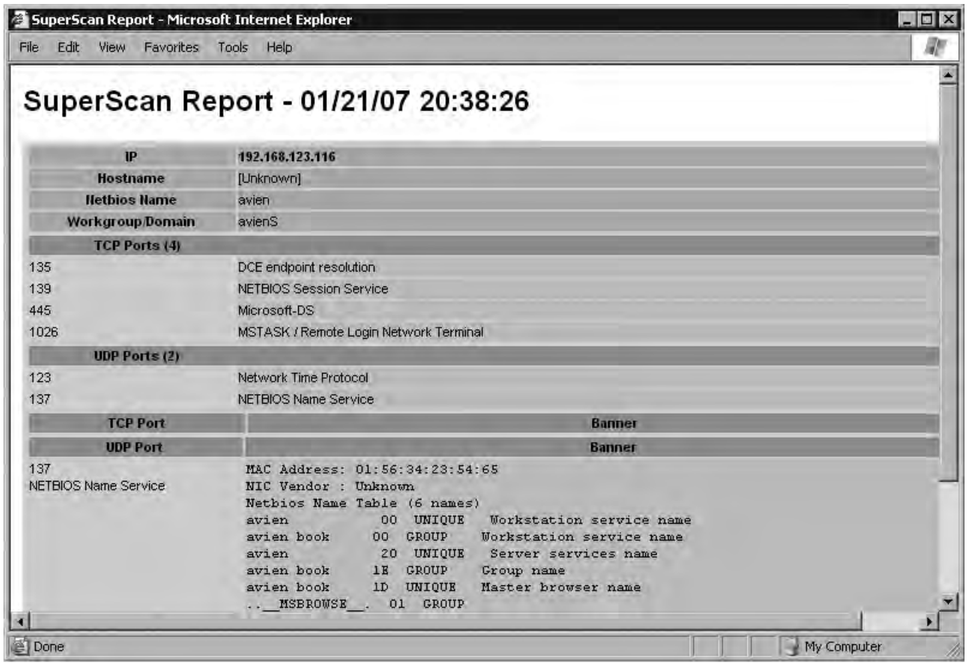
```
  Hostname: [Unknown]
  TCP ports (4) 135,139,445,1026
  UDP ports (2) 123,137

Total live hosts discovered       1
Total open TCP ports              4
Total open UDP ports              2
```

```
Performing hostname resolution...
Performing banner grabs...
  TCP banner grabbing (4 ports)
  UDP banner grabbing (2 ports)
Reporting scan results...
-------- Scan done --------

Discovery scan finished: 01/21/07 20:38:39
```

00:12 Saved log file Live: 1 TCP open: 4 UDP open: 2 1/1 done

In Figure 9.26, you can see how the report is laid out. The first item on the report is general information about the machine that was scanned, then follows a listing of TCP and UDP ports that Superscan found open. The section on the report is a listing of the individual ports that were found open, their common name, and the banner that is retrieved from the connection. The banner will usually contain identifying information about the application that is listening on that port, usually providing, as a minimum, the name and version information it has on that application.

Figure 9.26 HTML Report Generated After Running Superscan 4.0


SuperScan Report - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

SuperScan Report - 01/21/07 20:38:26

IP	**192.168.123.116**
Hostname	[Unknown]
Netbios Name	avien
Workgroup/Domain	avienS
TCP Ports (4)	
135	DCE endpoint resolution
139	NETBIOS Session Service
445	Microsoft-DS
1026	MSTASK / Remote Login Network Terminal
UDP Ports (2)	
123	Network Time Protocol
137	NETBIOS Name Service
TCP Port	**Banner**
UDP Port	**Banner**

137
NETBIOS Name Service

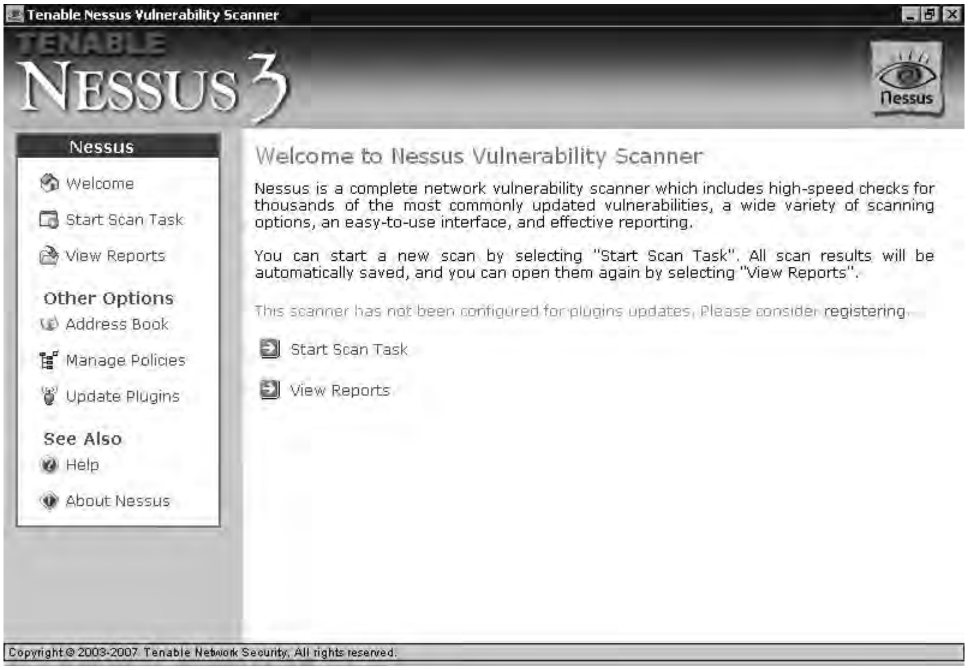
```
MAC Address: 01:56:34:23:54:65
NIC Vendor : Unknown
Netbios Name Table (6 names)
avien            00  UNIQUE   Workstation service name
avien book      00  GROUP    Workstation service name
avien            20  UNIQUE   Server services name
avien book      1E  GROUP    Group name
avien book      1D  UNIQUE   Master browser name
..__MSBROWSE__.  01  GROUP
```

There will come a time that you'll want to examine a remote machine for more than just open ports. You'll want to know if the machine has any known vulnerabilities, perhaps due to a Windows or Linux patch that either wasn't applied, or was not applied correctly. This is where you'll want to use a full vulnerability assessment tool. The most popular vulnerability assessment tool is, without a doubt, Nessus, from Tenable Network Security. Renaud Deraison started Nessus as the "Nessus Project" back in 1998. Four years later, he teamed up with Ron Gula and Jack Huffard to form Tenable Network Security, which offers Nessus free for download on their Website www.nessus.org.

Nessus is updated on a daily basis with the latest vulnerability checks. However, in order to get these same-day updates, Tenable Network Security charges a nominal subscription. However, Nessus can still be used without having to pay a fee. Users that who choose not to pay for daily updates can still use Nessus and update it as the paying customers do. However, your vulnerability database will be seven days old. In an enterprise setting, the fee for the subscription is well worth the cost, but Nessus is still quite usable without having the up-to-date subscription, if you don't mind the fact that your vulnerability database will be a week out of date.

Nessus is available for numerous different platforms, including Linux, FreeBSD, Solaris, Mac OS X, and Windows. It is extremely easy to use, and yet it has many advanced features that can be used to tailor scans according to the environment. Figure 9.27 shows the initial screen that you're presented with once you've installed Nessus. After clicking **Start Scan Task**, you enter the IP or machine name of the machine that you wish to scan. Then you're asked to choose the plug-in set to use. The default policies are very basic, yet very effective. Non-intrusive scans and intrusive scans make up the two basic policies. If you choose the non-intrusive scans, the machine that you are scanning should not be negatively affected in any way. With the non-intrusive scans, only passive checks are performed. If you choose the intrusive checks, you take the chance of performing a Denial of Service (DoS) on the machine that you're scanning, as you'll be attempting to brute-force passwords, possibly even attempting to overflow buffers. This could cause the machine being scanned to crash or re-boot, or trigger alerts from other security software, so care must be taken when performing intrusive scans.

Figure 9.27 The Initial GUI Screen Upon Starting Nessus



Welcome to Nessus Vulnerability Scanner

Nessus is a complete network vulnerability scanner which includes high-speed checks for thousands of the most commonly updated vulnerabilities, a wide variety of scanning options, an easy-to-use interface, and effective reporting.

You can start a new scan by selecting "Start Scan Task". All scan results will be automatically saved, and you can open them again by selecting "View Reports".

This scanner has not been configured for plugins updates. Please consider registering.

Start Scan Task

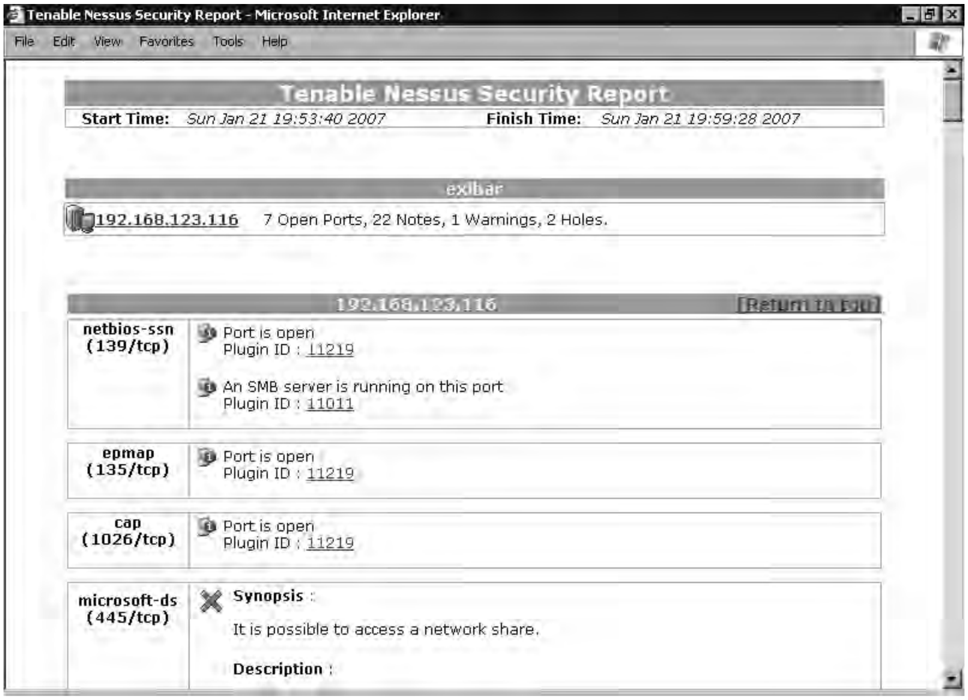
View Reports

Copyright © 2003-2007 Tenable Network Security. All rights reserved.

When the scan is complete, you'll be presented with an HTML report that contains the results of the scan. You may be surprised at what is found on your network! Figure 9.28 shows you a sample of one of the reports. The reports provided by Nessus are plain, basic, and chock-full of information about the machines that were scanned. The reports identify what ports are open, what vulnerabilities were found, where to go for more information about the vulnerability, and how to remedy the vulnerability found. The vulnerability remediation function is very useful. After all, what good is knowing there's a vulnerability if you don't know how to fix it?

Nessus also offers its own scripting language. This allows you to create your own vulnerability checks. Nessus Attack Scripting Language (NASL) is easy to learn to use for performing all the vulnerability checks you set up, and is the language used to create your own Nessus plug-ins. All of the existing NASL plug-ins are in plaintext. This means that you can open them up with Notepad or another text editor and see exactly how they work. This makes it easy to borrow code from existing NASL plug-ins for use in your own plug-ins.

Figure 9.28 Sample Report After Performing a Scan Using Nessus


Tenable Nessus Security Report - Microsoft Internet Explorer
File Edit View Favorites Tools Help

Tenable Nessus Security Report

Start Time: Sun Jan 21 19:53:40 2007 Finish Time: Sun Jan 21 19:59:28 2007

exibar

192.168.123.116 7 Open Ports, 22 Notes, 1 Warnings, 2 Holes.

192.168.123.116 [Return to top]

netbios-ssn (139/tcp)	Port is open Plugin ID : 11219 An SMB server is running on this port Plugin ID : 11011
epmap (135/tcp)	Port is open Plugin ID : 11219
cap (1026/tcp)	Port is open Plugin ID : 11219
microsoft-ds (445/tcp)	Synopsis : It is possible to access a network share. Description :

Advanced Tools: An Overview of Windows Code Debuggers

Sooner or later you will want to know absolutely everything about an executable file. You may want to know, for instance:

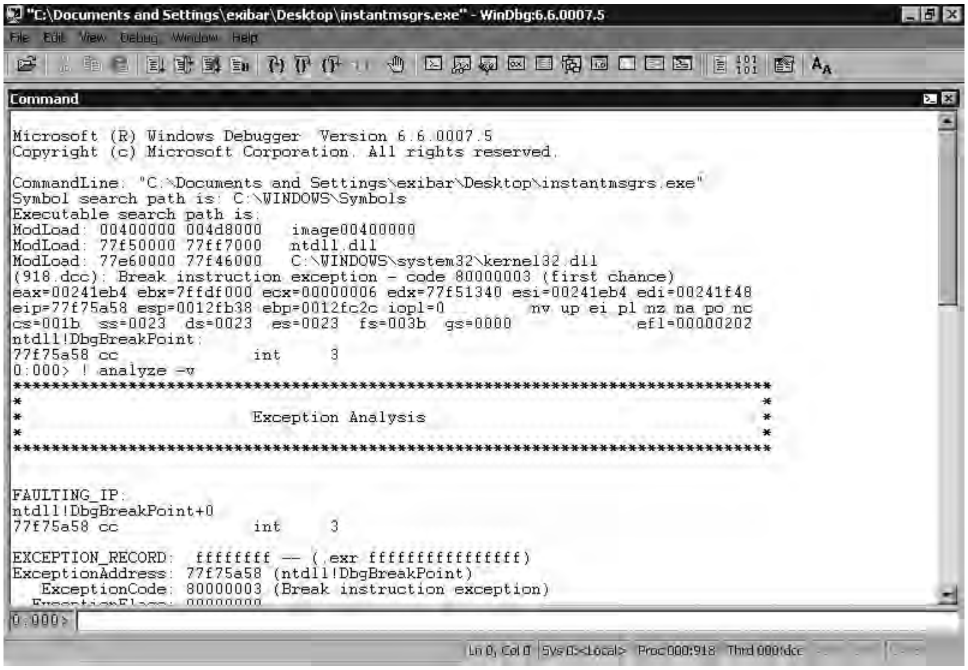
- The exact memory address that it is calling
- The exact region of memory that it is writing to
- What region it's reading from
- Which registers it's making use of

Debuggers will aid you in reverse-engineering a file for which you don't have the source code, by disassembling the file in question. (For more on the relationship between programming and debugging tools, see the later section on advanced forensic analysis.) This comes in handy when you're analyzing malware, as you almost never have access to the executable's original source code. The goal of this section is not to coach you in depth on how to use these debuggers, but simply to show you that they are out there and available for you to use. Debuggers are very powerful tools that take a long time to learn to use to their fullest extent.

There are three popular debuggers for the Windows platforms that we'll be discussing: two of them are free and one is a commercial product. The first debugger in this section is Windows Debugger (WinDbg), available free of charge from Microsoft. It's part of the Debugging Tools for Windows (www.microsoft.com/whdc/devtools/debugging/default.mspx). WinDbg, although basic in its functionality, will allow you to fully debug and analyze Windows applications and other user-mode items. Along with services, drivers, and other kernel mode items, it can be used to analyze crash dumps created by a Blue Screen of Death (BSoD). This is probably its most popular use. The first thing to keep in mind when using WinDbg is that you must download the proper symbol set for the operating system that you're working on. Symbol sets are available free from Microsoft at the link above. They are quite large; the average size is about 150Mb.

Figure 9.29 shows WinDbg starting to analyze a WootBot Variant with the file name "*instantmsgrs.exe*."

Figure 9.29 WinDbg Analysis of a WootBot Variant


"C:\Documents and Settings\exibar\Desktop\instantmsgrs.exe" - WinDbg:6.6.0007.5
File Edit View Debug Window Help
Command

```
Microsoft (R) Windows Debugger  Version 6.6.0007.5
Copyright (c) Microsoft Corporation. All rights reserved.

CommandLine: "C:\Documents and Settings\exibar\Desktop\instantmsgrs.exe"
Symbol search path is: C:\WINDOWS\Symbols
Executable search path is:
ModLoad: 00400000 004d8000   image00400000
ModLoad: 77f50000 77ff7000   ntdll.dll
ModLoad: 77e60000 77f46000   C:\WINDOWS\system32\kernel32.dll
(918.dcc): Break instruction exception - code 80000003 (first chance)
eax=00241eb4 ebx=7ffdf000 ecx=00000006 edx=77f51340 esi=00241eb4 edi=00241f48
eip=77f75a58 esp=0012fb38 ebp=0012fc2c iopl=0         nv up ei pl nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000202
ntdll!DbgBreakPoint:
77f75a58 cc               int     3
0:000> ! analyze -v
*******************************************************************************
*                                                                             *
*                        Exception Analysis                                   *
*                                                                             *
*******************************************************************************


FAULTING_IP:
ntdll!DbgBreakPoint+0
77f75a58 cc               int     3

EXCEPTION_RECORD:  ffffffff -- (.exr ffffffffffffffff)
ExceptionAddress: 77f75a58 (ntdll!DbgBreakPoint)
   ExceptionCode: 80000003 (Break instruction exception)
  ExceptionFlags: 00000000
0:000>
```

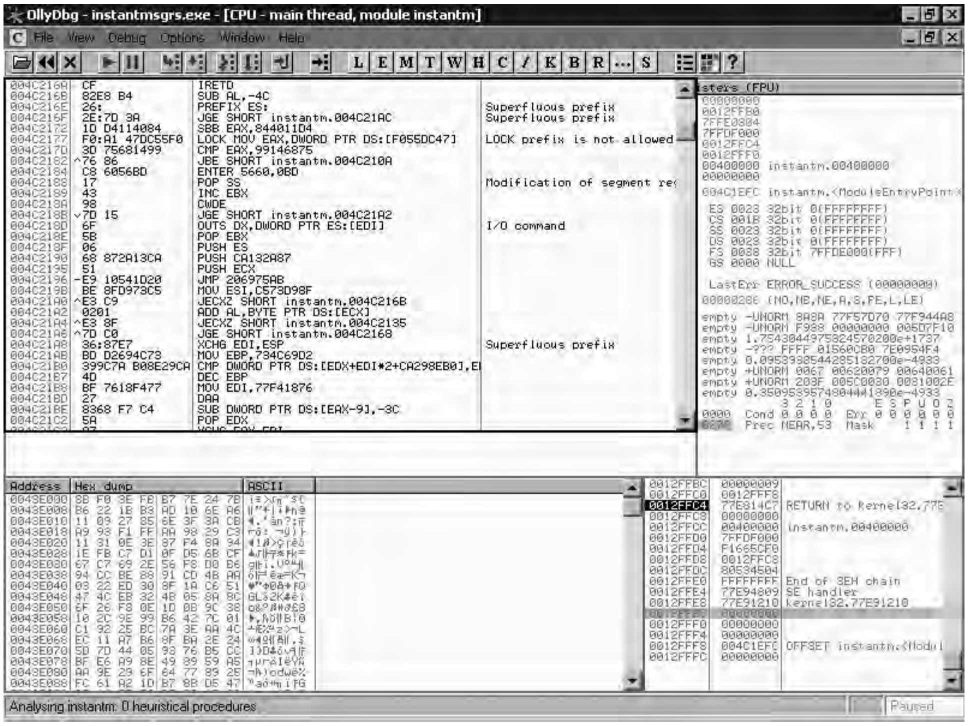
A very useful command for use with WinDbg is "*! Analyze –v*." If you load the proper symbol set and a mini-dump into WinDbg, you can retrieve information such as the exact instruction that was being executed at the time of the crash, the particular threads in use, loaded modules, and so on. If you were to load a full crash dump file, you have access to even more, including the contents of the process heap itself at the time of the crash.

NOTE

The same PE format is used by both file types. The only difference between the two is the use of a single distinguishing byte .exe and .dll, so that the operating system knows how to execute them. Note also that a number of other file types are also PE format files, for instance Control Panels (.*cpl*), and can be executed by the OS as if they were .*exes* or .*dlls*. Hence the frequent generic blocking of .*scrs*, .*ocxs* and so on by e-mail filters. For a fuller explanation of the PE format, try Matt Pietrek's Inside Windows article "An In-Depth Look into the Win32 Portable Executable File Format," at http://msdn.microsoft.com/msdnmag/issues/02/02/PE/

The next Debugger has gained in popularity tremendously over the past year since SoftICE was discontinued in April 2006. Oleh Yuschuk (Olly) released OllyDbg in November 2000, for free download. OllyDbg is a very powerful, full featured 32-bit assembler level-debugger. Figure 9.30 shows a WootBot variant being disassembled within OllyDbg. This utility basically picks up where WinDbg leaves off. Where WinDbg really shines at analyzing crash dumps and makes diagnosing a BSoD much easier, OllyDbg excels at binary code analysis, tracing memory registers and recognizing procedures, tables, strings, and more. OllyDbg is perfect for debugging or reverse engineering Windows malware code, as long as it's in the Windows Portable Executable (PE) format used in current generations of Windows executables (*.exe*) and Dynamic Link Libraries. (*.dll*).

Figure 9.30 OllyDbg Analyzing *instantmsgrs.exe*, a WootBot Variant


OllyDbg - instantmsgrs.exe - [CPU - main thread, module instantm]
C File View Debug Options Window Help
L E M T W H C / K B R ... S

```
004C216A   CF              IRETD
004C216B   82E8 B4         SUB AL,-4C
004C216E   26:             PREFIX ES:                               Superfluous prefix
004C216F   2E:7D 3A        JGE SHORT instantm.004C21AC              Superfluous prefix
004C2172   1D D4114084     SBB EAX,844011D4
004C2177   F0:A1 47DC55F0  LOCK MOV EAX,DWORD PTR DS:[F055DC47]     LOCK prefix is not allowed
004C217D   3D 75681499     CMP EAX,99146875
004C2182  ^76 86           JBE SHORT instantm.004C210A
004C2184   C8 6056BD       ENTER 5660,0BD
004C2188   17              POP SS                                   Modification of segment re
004C2189   43              INC EBX
004C218A   98              CWDE
004C218B  v7D 15           JGE SHORT instantm.004C21A2
004C218D   6F              OUTS DX,DWORD PTR ES:[EDI]               I/O command
004C218E   5B              POP EBX
004C218F   06              PUSH ES
004C2190   68 872A13CA     PUSH CA132A87
004C2195   51              PUSH ECX
004C2196  -E9 10541D20     JMP 206975AB
004C219B   BE 8FD973C5     MOV ESI,C573D98F
004C21A0  ^E3 C9           JECXZ SHORT instantm.004C216B
004C21A2   0201            ADD AL,BYTE PTR DS:[ECX]
004C21A4  ^E3 8F           JECXZ SHORT instantm.004C2135
004C21A6  ^7D C0           JGE SHORT instantm.004C2168
004C21A8   36:87E7         XCHG EDI,ESP                             Superfluous prefix
004C21AB   BD D2694C73     MOV EBP,734C69D2
004C21B0   399C7A B08E29CA CMP DWORD PTR DS:[EDX+EDI*2+CA298EB0],E
004C21B7   4D              DEC EBP
004C21B8   BF 7618F477     MOV EDI,77F41876
004C21BD   27              DAA
004C21BE   8368 F7 C4      SUB DWORD PTR DS:[EAX-9],-3C
004C21C2   5A              POP EDX
```

```
isters (FPU)
00000000
0012FFB0
7FFE0304
7FFDF000
0012FFC4
0012FFF0
00400000 instantm.00400000
00000000

004C1EFC instantm.<ModuleEntryPoint>

 ES 0023 32bit 0(FFFFFFFF)
 CS 001B 32bit 0(FFFFFFFF)
 SS 0023 32bit 0(FFFFFFFF)
 DS 0023 32bit 0(FFFFFFFF)
 FS 0038 32bit 7FFDE000(FFF)
 GS 0000 NULL

 LastErr ERROR_SUCCESS (00000000)

00000286 (NO,NB,NE,A,S,PE,L,LE)

empty -UNORM 8A8A 77F57D70 77F944A8
empty -UNORM F938 00000000 005D7F10
empty 1.7543044975824570200e+1737
empty -??? FFFF 01560CB0 7E0954F4
empty 0.0953938544285182700e-4933
empty +UNORM 0067 00620079 00640061
empty +UNORM 203F 005C0030 0081002E
empty 0.3509539574804441890e-4933
               3 2 1 0      E S P U O Z
0000  Cond 0 0 0 0  Err 0 0 0 0 0 0
027F  Prec NEAR,53  Mask    1 1 1 1
```

```
Address   Hex dump                  ASCII
0043E000  8B F0 3E FB B7 7E 24 7B
0043E008  B6 22 1B B3 AD 10 6E A6
0043E010  11 09 27 85 6E 3F 3A CB
0043E018  A9 93 F1 FF AA 98 29 C3
0043E020  11 31 0E 3E 87 F4 8A 94
0043E028  1E FB C7 D1 0F D5 6B CF
0043E030  67 C7 69 2E 56 F8 D0 B6
0043E038  94 CC BE 88 91 CD 4B AA
0043E040  03 22 ED 30 8F 1A C6 51
0043E048  47 4C EB 32 4B 05 8A 8C
0043E050  6F 26 F8 0E 1D 0B 9C 38
0043E058  10 2C 9E 99 B6 42 7C 01
0043E060  C1 92 25 BC 7A 3E AA 4C
0043E068  EC 11 A7 B6 8F BA 2E 24
0043E070  5D 7D 44 05 93 76 B5 CC
0043E078  BF E6 A9 8E 49 89 59 A5
0043E080  AA 9E 29 6F 64 77 89 25
0043E088  FC 61 A2 1D B7 8B D5 47
```

```
0012FFBC  00000009
0012FFC0  0012FFF8
0012FFC4  77E814C7  RETURN to kernel32.77E
0012FFC8  00000000
0012FFCC  00400000  instantm.00400000
0012FFD0  7FFDF000
0012FFD4  F1665CF0
0012FFD8  0012FFC8
0012FFDC  80534504
0012FFE0  FFFFFFFF  End of SEH chain
0012FFE4  77E94809  SE handler
0012FFE8  77E91210  kernel32.77E91210
0012FFEC  00000000
0012FFF0  00000000
0012FFF4  00000000
0012FFF8  004C1EFC  OFFSET instantm.<Modul
0012FFFC  00000000
```

```
Analysing instantm: 0 heuristical procedures                Paused
```

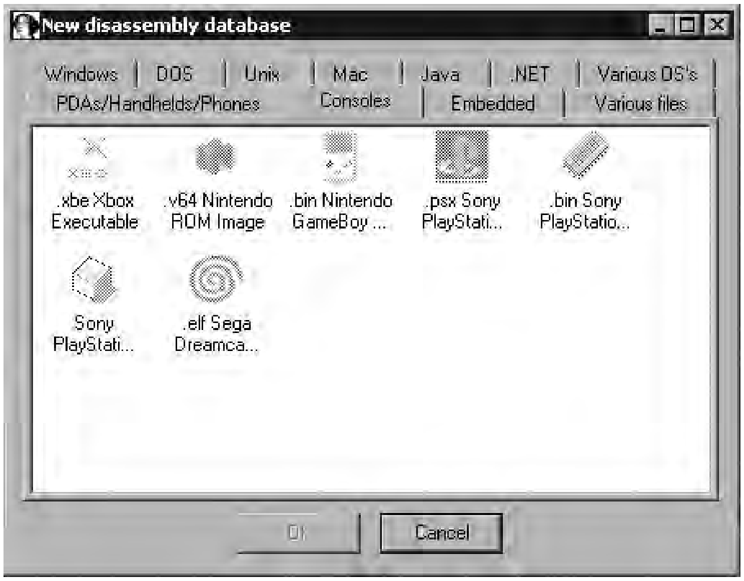
Let's say that you want to analyze a piece of malware that isn't an .exe or a .dll, or a Crash Dump. You can't use WinDbg, nor can you use OllyDbg, as neither of those debuggers can perform analysis on other types of files. This is where you need the "cream of the crop" in debuggers. To be precise, you need Interactive Disassembler Pro (IDA Pro), available from DataRescue. IDA Pro should be your first choice of debuggers for an enterprise environment. It isn't really expensive, and is well worth the nominal outlay for the features it offers.

TIP

DataRescue offers a demo version from their Web site at www.datarescue.com/idabase/index.htm. This version can only work with a limited range of file and processor types, is time limited, runs only as a Windows GUI application, and so on.

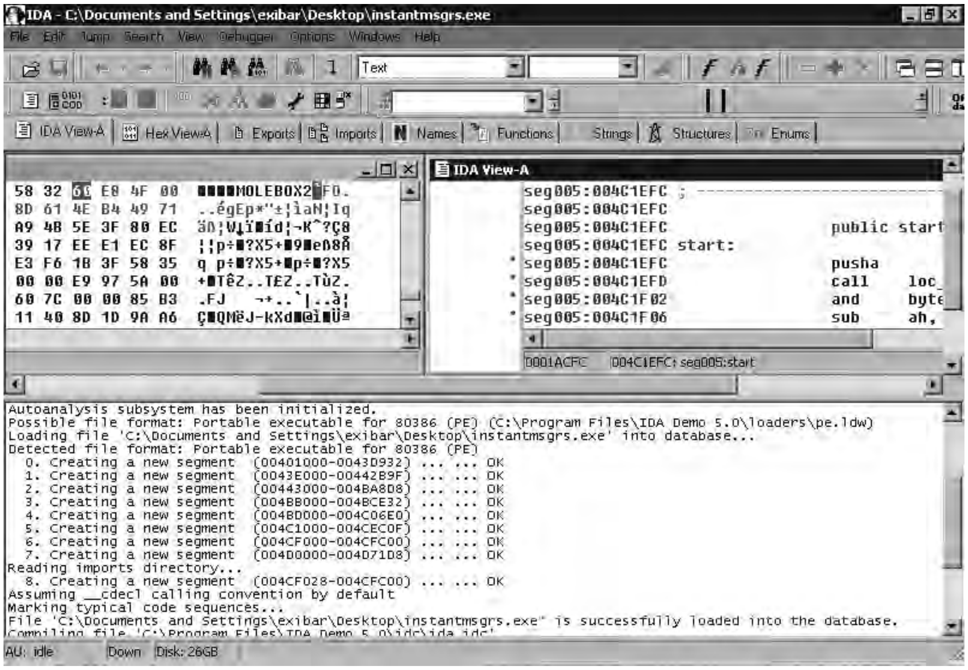
IDA Pro is much more than a simple debugger. It is a programmable, interactive disassembler and debugger. With IDA Pro you can reverse-engineer just about any type of executable or application file in existence. IDA Pro can handle files from console machines such as Xbox, Playstation, Nintendo, to Macintosh computer systems, to PDA platforms, Windows, UNIX, and a whole lot more. Figure 9.31 shows the initial load screen wizard when you first start IDA Pro. Notice all the file types and tabs that will help you select the proper analysis for the file type that you wish to disassemble.

Figure 9.31 IDA Pro's Disassembly Database Chooser Loads Upon Start



In Figure 9.32, IDA Pro has loaded and is disassembling a WootBot variant with file name *instantmsgrs.exe*. Part of what we can see from Figure 9.33 is that *instanmsgrs.exe* was packed using an executable packer called Molebox. You can also plainly see the memory calls that it's making, and the Windows DLLs that are being called. This type of information can be invaluable when it comes to fighting off a virus or malware outbreak, especially if you need to make a custom cleaner in order to repair your systems.

Figure 9.32 IDA Pro Disassembles *instantmsgrs.exe*, a WootBot Variant


IDA - C:\Documents and Settings\exibar\Desktop\instantmsgrs.exe
File Edit Jump Search View Debugger Options Windows Help
Text
IDA View-A
Hex View-A
Exports
Imports
Names
Functions
Strings
Structures
Enums
AU: idle
Down
Disk: 26GB

```
58 32 60 E8 4F 00   ■■■■MOLEBOX2■F0.
8D 61 4E B4 49 71   ..égEp*"±¦laN¦Iq
A9 4B 5E 3F 80 EC   ЗЛ¦Щ↓Ї■íd¦¬K^?Ç8
39 17 EE E1 EC 8F   ¦¦p÷■?X5+■9■eЛ8Å
E3 F6 1B 3F 58 35   q p÷■?X5+■p÷■?X5
00 00 E9 97 5A 00   +■TêZ..TEZ..TùZ.
60 7C 00 00 85 B3   .FJ   ¬+..`|..à¦
11 40 8D 1D 9A A6   Ç■QMëJ-kXd■@ì■Üª
```

```
seg005:004C1EFC ; ---------------------------------
seg005:004C1EFC
seg005:004C1EFC                 public start
seg005:004C1EFC start:
seg005:004C1EFC                 pusha
seg005:004C1EFD                 call    loc_
seg005:004C1F02                 and     byte
seg005:004C1F06                 sub     ah,
0001ACFC        004C1EFC: seg005:start
```

```
Autoanalysis subsystem has been initialized.
Possible file format: Portable executable for 80386 (PE) (C:\Program Files\IDA Demo 5.0\loaders\pe.ldw)
Loading file 'C:\Documents and Settings\exibar\Desktop\instantmsgrs.exe' into database...
Detected file format: Portable executable for 80386 (PE)
  0. Creating a new segment  (00401000-0043D932) ... ... OK
  1. Creating a new segment  (0043E000-00442B9F) ... ... OK
  2. Creating a new segment  (00443000-004BA8D8) ... ... OK
  3. Creating a new segment  (004BB000-004BCE32) ... ... OK
  4. Creating a new segment  (004BD000-004C06E0) ... ... OK
  5. Creating a new segment  (004C1000-004CEC0F) ... ... OK
  6. Creating a new segment  (004CF000-004CFC00) ... ... OK
  7. Creating a new segment  (004D0000-004D71D8) ... ... OK
Reading imports directory...
  8. Creating a new segment  (004CF028-004CFC00) ... ... OK
Assuming __cdecl calling convention by default
Marking typical code sequences...
File 'C:\Documents and Settings\exibar\Desktop\instantmsgrs.exe' is successfully loaded into the database.
Compiling file 'C:\Program Files\IDA Demo 5.0\idc\ida.idc'
```

Advanced Analysis and Forensics

The end of 2006 saw an explosion in spam e-mails. Security analysts agree that the vast majority of spam sent today is broadcast by huge zombie/bot armies. These are PCs (usually) that are infected with various forms of malware, allowing them to be accessed covertly and used remotely for spam distribution and other unwanted, unsuspected activities. The AV industry (and related antimalware companies) is barely coping with the amount of new malware variants being released daily, as both reactive and proactive defensive measures fail to stop the intrusions.

When an incident happens in an organization, the security administrator often has to assess the impact of the incident quickly and decide what should be done with the infected machine(s). While the correct answer, unfortunately, is often to reinstall the machine from scratch, this does not prevent future infections, unless the infection vector has been determined.

It usually takes hours or longer for AV companies to release definitions for new malware and days, if not weeks, to post technical documents about most common malware. And it is often the case that particular malware that has hit the organization is not widespread, in which case the technical analysis from the AV vendor may not be available at all.

In cases like this, the security administrator often has to analyze the malware himself and gather as much information as possible in order to ensure that his organization's core business functions remain stable and uninterrupted.

Tools & Traps

Lies, Damned Lies, and (Spam in 2006) Statistics

Spam statistics are notoriously variable according to source. In this instance, traffic on Anti Virus Information Exchange Network (AVIEN) and other lists clearly suggested a spike in spam volumes over the last part of 2006, and spam vendor reports such as those at www.marshal.com/trace/spam_statistics.asp and www.messagelabs.com/Threat_Watch/Threat_Statistics supported that supposition.

However, the media seemed to be more interested in a report by SoftScan that there was actually a 30 percent drop in spam volumes at the beginning of 2007 (www.softscan.co.uk/composite-491.html). Several theories were put forward to account for this, including the disruption of network services by an Asian earthquake and the possible unavailability of a major robot network (botnet). The media chose to focus on the alternative suggestion of the replacement of large quantities of compromised PCs by new machines received as Christmas presents.

There's nothing wrong with some healthy speculation; however, it may be misleading to build too many theories on data received from a single source.

Advanced Malware Analysis

Analysis of any code can be a very demanding and complex process. It becomes even more difficult when dealing with unknown, potentially malicious, and often obfuscated code. Malware can be analyzed statically when we review the code, in order to determine what it will do. Alternatively, it can be reviewed dynamically, when we observe the behavior of the malware when it is being executed. Depending on exactly how you plan to deal with what you find and how much time you are prepared to invest in analysis, you might want to use either method, a combination of both approaches, or elements of both approaches.

Static (Code) Analysis

The majority of systems and application programs today are developed in high-level programming languages such as C or C++. These programs are then compiled into machine code translating into a stand-alone executable program. An aspiring virus writer can choose to use any high-level programming language to create a virus. However, it has not been unusual to see code, even malicious code, written completely in assembly language. This has become less common in the last couple of years, since malicious writers are also looking at "return of investment (ROI)" when writing malware. It is much quicker and easier to code in high-level programming languages.

Static or code analysis is (usually) based on manual review of unknown code. When analyzing malicious programs, you will almost always encounter binary executables. This means that code analysis must be performed on disassembled instructions, a process that is not trivial and that requires a lot of knowledge about machine code, the lowest level of binary code. There are various utilities that can make this process easier.

Tools & Traps

The 10-Second Guide to Programming and Disassembly

Dedicated, high-level programming languages like C, C++, Delphi, and so on are often roughly divided into compiled languages and interpreted languages. In fact, this is an artificial distinction. There's rarely any absolute reason why a given language needs to be one or the other, though the specific characteristics of a given language may make it difficult to include all of its features in both forms. Many languages are, in fact, implemented as interpreters and as compilers, sometimes packaged together.

The term compiler is popularly applied in PC programming to a program that translates human read-writable source code into object code, usually with the intention of creating a standalone executable program. (Real computer scientists are probably already sniffing contemptuously at this gross oversimplification of an esoteric concept.)

An interpreter runs through a program translating one statement at a time into machine code, rather than translating the entire program into a single executable. The execution of the program could be said to take place within the application environment rather than in system space. Tools like Visual Basic for Applications (VBA) provide advanced programming functionality within an application like Microsoft Word that is not primarily seen as a programming tool in itself.

Machine code is the lowest level of code, the level at which the computer is able to read it directly. Assembly language (often referred to informally as "assembler") is the next level up from machine code. It allows the programmer to use mnemonics like "MOV BL, 34H" or "JMP MYLABEL" instead of pure human-hostile machine code ("001010011"), while retaining the advantages of working "close to the metal."

A disassembler translates machine code to assembly language, which is a little easier to read for humans, and is therefore useful for reverse engineering. A decompiler is somewhat similar in intent, but translates to something closer to a high-level language rather than assembly language.

Reverse engineering is the process of ascertaining the workings and technological principles of a system (in this case, a possibly malicious program) by examination and analysis.

Packers and Memory Dumping

The first obstacles that malicious authors put in our path are runtime packers. The idea of packing (compressing) an executable is not new. Back in the old days of Commodore 64, programs were packed with simple packers in order to decrease their size, so the transfer of files would take less time. This is still one of the reasons malware is packed today: malicious authors want their programs to be as small as possible to reduce the loading time.

The other and most important goal of a packer is to make analysis or reverse engineering of the code more difficult. All packed malware consists of the unpacker code and packed data, which typically look like random data. The benefit of packed malware, from a malicious author's point of view, is that it cannot be analyzed until it is properly unpacked. This presents a problem for AV software as well, and is the main protection that malicious authors employ today. You might ask why AV vendors do not just detect the unpacker code. The answer is simple: there are legitimate purposes for runtime compression, and it is often the case that malicious authors use legitimate packers on purpose, so as to hide their code. (Actually, there has been ongoing discussion for some years as to whether it's acceptable to filter all packed code on the grounds that it's probably malicious, but that is still sometimes viewed as controversial.)

Unpacking can therefore be the most difficult part of a static analysis. Malicious authors usually go through a very lengthy and complex process to obfuscate the unpacker and make it as difficult to analyze as possible. After all, if it is difficult for a human analyst to unpack the malware, it will be even more difficult for an automated process running inside an antimalware program.

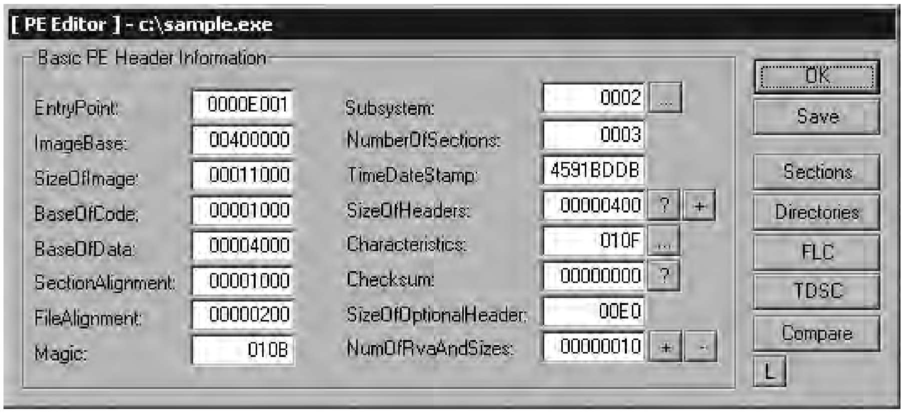
Before we start unpacking malware, you need to be aware of some basics about executable file format on Windows operating systems. All executables and DLLs, in order to be valid and interpreted correctly by the operating environment as executable code, have to be in a special format called Packed Executable (PE). This format defines information that the operating system loader needs in order to start the program.

NOTE

You can find detailed information about the PE format at www.microsoft.com/whdc/system/platform/firmware/PECOFF.mspx. The document "Visual Studio, Microsoft Portable Executable and Common Object File Format Specification," describes the structure of executable (image, PE) files and object files (Common Object File Format) under Windows®.

It is easy to identify the PE header. Every PE file has to start with the two bytes "MZ" (0x4D 0x5A), in order to be identified by the operating system correctly. PE headers contain crucial information about the file that includes the Original Entry Point (OEP), ImageBase, BaseOfCode, BaseOfData, NumberOfSections, and other data. When the loader loads data from the file into the memory, it uses the ImageBase address as the base, and all other addresses from the PE header are taken as an offset from ImageBase. In other words, the real entry address (the address that the program starts at) will be calculated as the sum of OEP and ImageBase. As previously mentioned, when an executable is packed, the OEP actually points to the unpacker code that will prepare (unpack) the file and then jump to the unpacked code. In the case of real malware, this will be the beginning of the actual malicious code. The unpacker will, in fact, create a new OEP address. This is an important part of the unpacking process. The other options in the PE header include BaseOfCode, which specifies the address in memory on which program code is stored, BaseOfData (which is the memory section holding program's data), and NumberOfSections, which defines the total number of sections in the PE file.

Figure 9.33 Section Table of a Malicious Program



EntryPoint:	0000E001	Subsystem:	0002
ImageBase:	00400000	NumberOfSections:	0003
SizeOfImage:	00011000	TimeDateStamp:	4591BDDB
BaseOfCode:	00001000	SizeOfHeaders:	00000400
BaseOfData:	00004000	Characteristics:	010F
SectionAlignment:	00001000	Checksum:	00000000
FileAlignment:	00000200	SizeOfOptionalHeader:	00E0
Magic:	010B	NumOfRvaAndSizes:	00000010

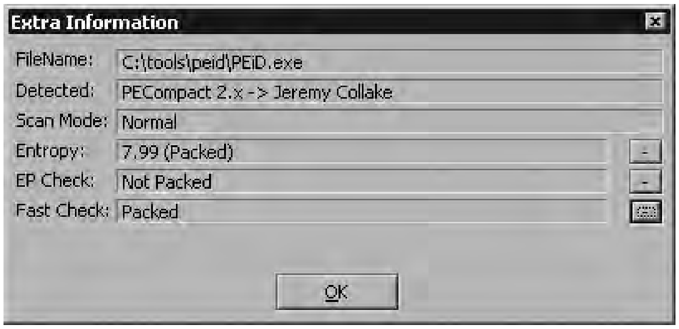
The section information here was listed with LordPE (available at http://scifi.pages.at/yoda9k/LordPE/LPE-DLX.ZIP), a program that allows you to review and edit PE header information.

The first step, when analyzing an unknown malicious program, is to determine if it is indeed packed and, if possible, to determine what packer has been used. At the time of writing this book there are around 600 packers and the most popular ones, such as UPX or ASPack are frequently used. Knowing what packer has been used will allow you to decide how to unpack the program, and whether it is worth going through the trouble of manually unpacking it. A utility that you should consider as an essential component of your reverse-engineering arsenal is PEiD (available at http://peid.has.it/. PEiD is a very powerful utility that has a database of fingerprints for identifying most popular packers. The database is also expanded regularly, so when a new packer is encountered, the PEiD community quickly generates a unique fingerprint that identifies it. In cases when PEiD does not identify the packer, you can still use it to see if the malware was packed or not. As previously mentioned, a packed program has some characteristics that can be easily identified, and the most important one is that no program code or data can be seen when it is packed or encrypted. PEiD can calculate program entropy and, depending on the result of the calculation, conclude whether or not a program is packed.

NOTE

Program entropy in the context of data compression is a measure of the amount of non-redundant, non-compressible data a file contains. Files that contain large quantities of redundant material (e.g. repetitive byte sequences) can be compressed very small, even with a fairly unsophisticated compression algorithm.

Figure 9.34 PEiD Calculating Entropy of a Packed Program



FileName: C:\tools\peid\PEiD.exe
Detected: PECompact 2.x -> Jeremy Collake
Scan Mode: Normal
Entropy: 7.99 (Packed)
EP Check: Not Packed
Fast Check: Packed

The process of unpacking can be very complex, depending on the packer,. Besides the code obfuscation that is typical of most packers, the authors of malicious software add various anti-debugging techniques. They will, for instance, attempt to detect when the program is being run inside a virtual machine, which usually means that someone is attempting to analyze/reverse-engineer the code. Again, depending on the time and effort you want to invest into the analysis of a malicious program, and whether an unpacker utility is readily available, you might want to perform dynamic analysis (see later section) or utilize dumps of process memory, which will in most cases circumvent packers.

Quick Assessment

In some cases, it is not necessary to perform a full blown analysis and (if appropriate) manual unpacking when you want to assess quickly what an unknown executable is doing. However, you do need to unpack program code and data, so as to see what it is doing. One easy way to retrieve unpacked program code is to dump its memory when it is running. The assumption and hope here is that the program will not just do something and exit immediately, as in this case it is difficult to "capture" it. The benefit of this approach is that, as you execute the malware, it will automatically unpack itself and you will be able to dump the unpacked code and data to a file for further analysis.

The utility that is most commonly used for this purpose is LordPE, but any competent process dump utility will do the same job. Of course, it must be stressed again that you should do this only on an isolated machine, not one which is connected to a network. Alternatively, you may choose to do it in an appropriate virtual environment (see the section on dynamic analysis for virtual machine requirements). The main reason for this is that you do not know what the malware you are analyzing does, and you do not, therefore, want to risk getting other machines in your network infected by an unanalyzed threat. The procedure is as follows: you should start LordPE first, then execute the unknown file, find its process in memory and dump it with LordPE to disk. This will create a memory image of the process on your disk. As the file has to unpack itself in order to run the real code, the image will contain raw program code and data. Unless, that is, further obfuscation has been used, as happens with a well-known packer called Armadillo. This program packs data in multiple sections and unpacks only what is needed for the current execution. All other parts of the program are still packed and/or encrypted. In cases like this, a memory dump will not contain the whole program, but can still be useful.

A dumped process image cannot be directly started, because it will not contain a proper OEP address. (Remember, the first OEP address points to the unpacker code. Once the program is unpacked, OEP needs to be changed to point to the real start address.)

Now that you have an image of the unpacked program code and data (or alternatively, once you unpacked it manually or with an unpacker utility, such as the one available for UPX), you can do a quick assessment of what the program actually does. Most program files under Windows operating systems make great use of DLLs. A lot of DLLs come preinstalled with Windows, and offer basic operating system and network functionality. In order to use functions that are available in DLLs (functions are *exported* by DLLs), every program has to *import* them. All DLLs that need to be imported by a program are listed in the Import Table, which is a part of the PE header. One problem springs to mind here: if you just dumped the process' memory with a utility such as LordPE, you will not have a proper Import Table, so you cannot just refer to this part of the header.

A very brief and quick assessment of an unknown file can be carried out with a utility called "*strings*" (www.microsoft.com/technet/sysinternals/Miscellaneous/Strings.mspx).

The *strings* utility lists all textual (ASCII) strings that were found in a file and are, by default, longer than three characters.

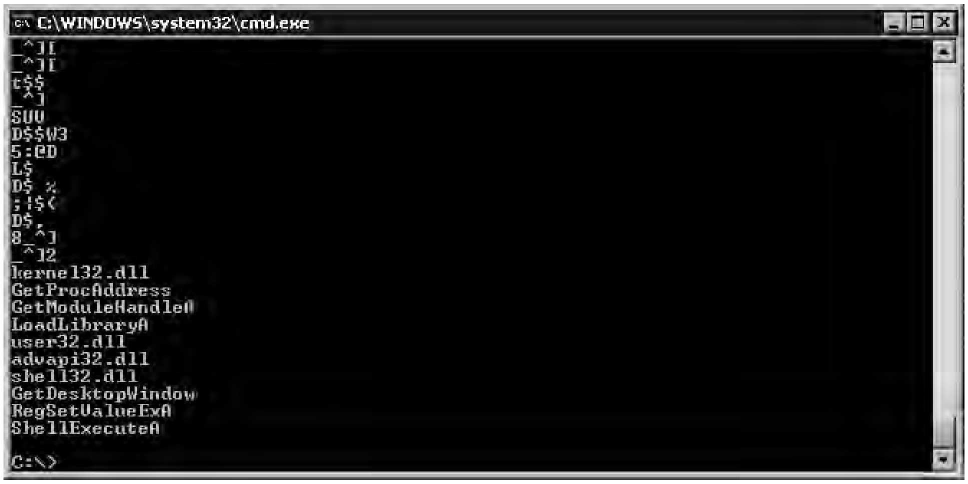
Obviously, if the program has been unpacked, running *strings* on it will result in a wealth of information. It is not unusual to find the DNS address of a command and control (C&C) IRC server, or the URL that the malware visits, shown as clear text in the program data. The *strings* utility can also list DLLs that the program imports, as well as the functions. In some cases, malware authors try to obfuscate text strings such as URLs so that they can't be viewed with simple utilities (such as *strings*): however, all of these text strings can still be spotted, with a little bit of practice and experience. Figure 9.35 shows the *strings* utility run on a malicious, packed file; you can see imported DLLs in clear text. String obfuscation, if present, is typically done through simple Exclusive Or (XOR) operations, or just by swapping bytes.

NOTE

The Windows *strings* utility has been ported from UNIX. However, its functionality and options are a little different. The UNIX utility, by default, looks for strings of four or more printable characters ending with a newline or null, and uses the options –*a* (search entire file), -*o* (display offset position before string), and –*n l* (find strings of minimum length *l*)

Mark Russinovich's port to Windows includes the options:
-*s* (recurse subdirectories)
-*a* (search for ASCII only)
-*u* (search for Unicode only)

Figure 9.35 Running the *strings* Utility on a Packed File



```
_^][
_^][
t$$
_^]
SUV
D$$W3
5:@D
L$
D$ %
;!$(
D$,
8_^]
_^]2
kernel32.dll
GetProcAddress
GetModuleHandleA
LoadLibraryA
user32.dll
advapi32.dll
shell32.dll
GetDesktopWindow
RegSetValueExA
ShellExecuteA

C:\>
```

At this stage, you will have enough high-level information about what the unknown file does. If you need more information than this, you will have to start disassembling it, or running it through a debugger. This might be necessary when certain parts of the malware are obfuscated. If the obfuscation is obvious, (e.g., XOR encoding), you can try cracking it manually, or using one of the brute force tools.

Disassembling Malware

Disassembly tools have come a long way since their beginning, and are a great help when reverse engineering any unknown code. In most cases, just browsing through the disassembled code without any further action will be enough to see what it does. Malicious authors try to prevent disassembling by creating obfuscated or self-modifying code so, in some cases, disassembly will not be enough. In such a case you will have to go one step further and debug the application.

Probably the best disassembler tool available today for Windows operating systems is DataRescue's IDA Pro. While the latest version of this tool has an integrated debugger, at the moment we will concentrate on its use as a disassembler. IDA Pro has some very powerful features for analyzing the code, so you can get a lot of information just by reading IDA Pro's comments. To make the best use of a disassembler, you will need to be familiar with x86 machine instruction codes.

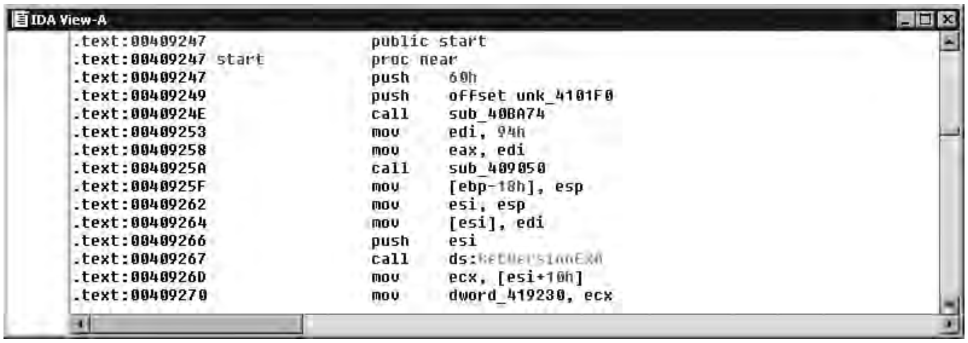
NOTE

X86 is a generic term for the microprocessor family derived from Intel's 8086 (and 8088), including separate math co-processors for early models, compatible processors by other makers such as AMD, and more recent processors such as the Pentium sub-family, which no longer have the X86 model numbering. The details of the instruction set are beyond the scope of this chapter, but these two URLs will give you a flavor.

http://en.wikipedia.org/wiki/X86_instruction_listings
http://en.wikipedia.org/wiki/X86_assembly_language

One amazingly useful function of IDA Pro is its ability to generate program flow charts. IDA Pro can analyze the code and create a nice flow chart from it (essentially a representation of the functions and code blocks called during execution). This is particularly useful when analyzing packers, as in some cases it can clearly show the process flow of an unpacker directing you towards the new OEP. Besides this, IDA Pro, like almost all disassemblers and debuggers, will show and parse imported DLLs and functions, so you can easily jump to code that interests you. For example, when analyzing a downloader, you will most likely be interested in any Internet-related functions. Imported functions are easy to locate, so all you have to do is check input parameters for the function call. This will enable you to identify second-stage malware downloaded by the downloader module you are analyzing.

Figure 9.36 IDA Pro Disassembling the YaY Trojan



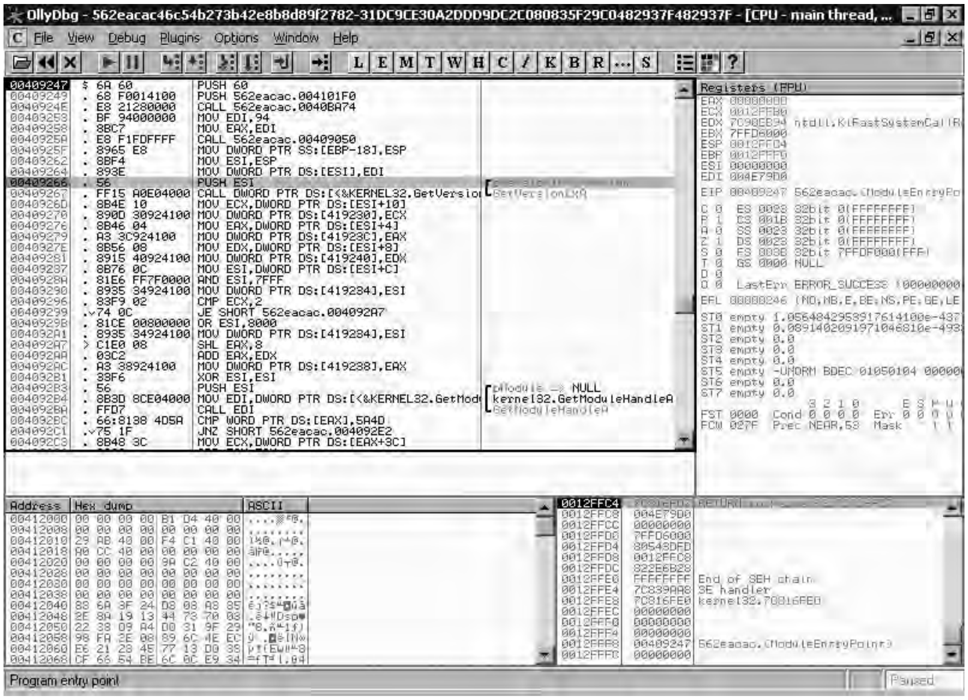
```
.text:00409247                  public start
.text:00409247 start            proc near
.text:00409247                  push    60h
.text:00409249                  push    offset unk_4101F0
.text:0040924E                  call    sub_40BA74
.text:00409253                  mov     edi, 94h
.text:00409258                  mov     eax, edi
.text:0040925A                  call    sub_409050
.text:0040925F                  mov     [ebp-18h], esp
.text:00409262                  mov     esi, esp
.text:00409264                  mov     [esi], edi
.text:00409266                  push    esi
.text:00409267                  call    ds:GetVersionExA
.text:0040926D                  mov     ecx, [esi+10h]
.text:00409270                  mov     dword_419230, ecx
```

Debugging Malware

The final step in any analysis of unknown code is debugging. This step requires the most detailed knowledge of machine code instructions, as well as of internal operating system functions. This is often the most time-consuming part of the analysis. Generally, debugging is needed only when you want to analyze a piece of malware thoroughly, or when you encounter obfuscated code that has to be debugged (observed during execution) in order to see what it is really doing. Even in cases like this, you might first want to perform a dynamic (behavioral) analysis of malware, which might provide you with enough information to avoid the need to spend hours on manual analysis using a debugger.

If you still wish to proceed along this route, you should get familiar with OllyDbg (www.ollydbg.de), the most popular freeware debugger today. OllyDbg, like IDA Pro, also has pretty powerful analysis features that will be of great help when analyzing unknown malware. OllyDbg can enumerate all imported DLLs, and can also recognize API calls.

Figure 9.37 OllyDbg Analyzing a Malicious Program



OllyDbg - 562eacac46c54b273b42e8b8d89f2782-31DC9CE30A2DDD9DC2C080835F29C0482937F482937F - [CPU - main thread, ...

```
00409247  $ 6A 60          PUSH 60
00409249  . 68 F0014100    PUSH 562eacac.004101F0
0040924E  . E8 21280000    CALL 562eacac.0040BA74
00409253  . BF 94000000    MOV EDI,94
00409258  . 8BC7           MOV EAX,EDI
0040925A  . E8 F1FDFFFF    CALL 562eacac.00409050
0040925F  . 8965 E8        MOV DWORD PTR SS:[EBP-18],ESP
00409262  . 8BF4           MOV ESI,ESP
00409264  . 893E           MOV DWORD PTR DS:[ESI],EDI
00409266  . 56             PUSH ESI
00409267  . FF15 A0E04000  CALL DWORD PTR DS:[<&KERNEL32.GetVersio   GetVersionExA
0040926D  . 8B4E 10        MOV ECX,DWORD PTR DS:[ESI+10]
00409270  . 890D 30924100  MOV DWORD PTR DS:[419230],ECX
00409276  . 8B46 04        MOV EAX,DWORD PTR DS:[ESI+4]
00409279  . A3 3C924100    MOV DWORD PTR DS:[41923C],EAX
0040927E  . 8B56 08        MOV EDX,DWORD PTR DS:[ESI+8]
00409281  . 8915 40924100  MOV DWORD PTR DS:[419240],EDX
00409287  . 8B76 0C        MOV ESI,DWORD PTR DS:[ESI+C]
0040928A  . 81E6 FF7F0000  AND ESI,7FFF
00409290  . 8935 34924100  MOV DWORD PTR DS:[419234],ESI
00409296  . 83F9 02        CMP ECX,2
00409299  .v74 0C          JE SHORT 562eacac.004092A7
0040929B  . 81CE 00800000  OR ESI,8000
004092A1  . 8935 34924100  MOV DWORD PTR DS:[419234],ESI
004092A7  > C1E0 08        SHL EAX,8
004092AA  . 03C2           ADD EAX,EDX
004092AC  . A3 38924100    MOV DWORD PTR DS:[419238],EAX
004092B1  . 33F6           XOR ESI,ESI
004092B3  . 56             PUSH ESI                                 pModule => NULL
004092B4  . 8B3D 8CE04000  MOV EDI,DWORD PTR DS:[<&KERNEL32.GetMod   kernel32.GetModuleHandleA
004092BA  . FFD7           CALL EDI                                 GetModuleHandleA
004092BC  . 66:8138 4D5A   CMP WORD PTR DS:[EAX],5A4D
004092C1  .v75 1F          JNZ SHORT 562eacac.004092E2
004092C3  . 8B48 3C        MOV ECX,DWORD PTR DS:[EAX+3C]
```

```
Registers (FPU)
EAX 00000000
ECX 0012FFB0
EDX 7C90EB94 ntdll.KiFastSystemCallR
EBX 7FFD6000
ESP 0012FFC4
EBP 0012FFF0
ESI 00000000
EDI 004E79D0

EIP 00409247 562eacac.<ModuleEntryPo

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDF000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000

EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE

ST0 empty 1.0564842953917614100e-437
ST1 empty 0.0891402091971046810e-493
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty -UNORM BDEC 01050104 00000
ST6 empty 0.0
ST7 empty 0.0
               3 2 1 0      E S P U
FST 0000  Cond 0 0 0 0  Err 0 0 0 0
FCW 027F  Prec NEAR,53  Mask    1 1
```

```
Address  Hex dump                 ASCII
00412000 00 00 00 00 B1 D4 40 00 ....▒╘@.
00412008 00 00 00 00 00 00 00 00 ........
00412010 29 AB 40 00 F4 C1 40 00 )½@.⌠┴@.
00412018 A0 CC 40 00 00 00 00 00 á╠@.....
00412020 00 00 00 00 9A C2 40 00 ....Ü┬@.
00412028 00 00 00 00 00 00 00 00 ........
00412030 00 00 00 00 00 00 00 00 ........
00412038 00 00 00 00 00 00 00 00 ........
00412040 83 6A 3F 24 D8 08 A3 85 âj?$╪.úà
00412048 2E 8A 19 13 44 73 70 03 .è↓‼Dsp♥
00412050 22 38 09 A4 D0 31 9F 29 "8.ñ╨1ƒ)
00412058 98 FA 2E 08 89 6C 4E EC ÿ·..ëlN∞
00412060 E6 21 28 45 77 13 D0 38 µ!(Ew‼╨8
00412068 CF 66 54 BE 6C 0C E9 34 ╧fT╛l.Θ4
```

```
0012FFC4  7C816FD7  RETURN to ...
0012FFC8  004E79D0
0012FFCC  00000000
0012FFD0  7FFD6000
0012FFD4  80543DFD
0012FFD8  0012FFC8
0012FFDC  822E6B28
0012FFE0  FFFFFFFF  End of SEH chain
0012FFE4  7C839AA8  SE handler
0012FFE8  7C816FE0  kernel32.7C816FE0
0012FFEC  00000000
0012FFF0  00000000
0012FFF4  00000000
0012FFF8  00409247  562eacac.<ModuleEntryPoint>
0012FFFC  00000000
```

Program entry point   Paused

To get back to the previous example, once you have located Internet-related functions, analysis with OllyDbg can be extremely easy. All you need to do is set a breakpoint on all function calls. This will cause the program execution to stop when an instruction with a breakpoint is reached. If you set breakpoints on API calls, you can examine their parameters just before they are called. All parameters will be located on the stack, because that is the way parameters are passed to functions.

Tools & Traps

API Emulation

Some malware goes so far as to avoid importing any functions from available DLLs. Instead, it emulates all of the APIs it needs. This means that you cannot list the functions, so you cannot easily set breakpoints on them, as all API calls will just be a part of the malware code.

Continued

In some cases, malware authors even use this to trap reverse engineers: they may import functions that are never used (having used the emulated ones instead). If you set a breakpoint on such fake function calls and allow the program to execute, you might find that your machine has been infected and the breakpoint was never reached.

The most helpful feature for analyzing unknown code that any debugger offers, apart from setting breakpoints, is step-by-step execution of code. With this asset on hand, the only limitation you're likely to meet on exploring the unknown world of malware will be the amount of time that you're prepared to invest.

It is quite common for malware authors to go one step further when they produce obfuscated code. They very often try to detect the presence of a debugger process in order to prevent you from reverse engineering their programs. In cases like this, it might be much faster and easier just to observe the behavior of the unknown program. That might give you enough information on what it does, and you can leave detailed analysis to AV vendors.

In some cases, you will have to resort to using a kernel debugger. The previous tool of choice for most researchers has been SoftIce, but as it is not being developed or supported any more, your best bet in this arena will be Microsoft's debugger (www.microsoft.com/whdc/devtools/debugging/installx86.mspx). This is a full-blown kernel debugger, but it has fewer features and its interface is more difficult to use, compared to the programs previously described.

Dynamic (Behavior) Analysis

While static (code) analysis can demand a great deal of time and knowledge, dynamic or behavior analysis can, in some cases, be much faster and yield practically the same information. Depending on the malware that you are analyzing, behavior analysis can sometimes be as simple as determining by eye what is going on (for example, most adware programs simply open popup windows with advertisements). With the help of some appropriate tools, behavior analysis will take you a long way down the road to accurate analysis. When resorting to it, you should first prepare the environment for running any unknown and potentially malicious code.

Isolated Environments

By now it should be clear that behavioral analysis of any potentially malicious program must be carried out in a strictly isolated environment. The main reasons for this is to prevent any uncontrolled replication of malware, and the avoidance of potential attacks towards your (or someone else's) infrastructure that might be the result of executing malware on a live system. For example, various bots are often channels for DoS attacks. Historically, malware researchers used physical machines that were either disconnected from the network or connected to an isolated segment. Using a physical machine makes reversion to the previous, known state of the operating system much more difficult. You need to either use an imaging utility (such as Symantec Ghost – see www.symantec.com/enterprise/products/overview.jsp?pcid=1025&pvid=865_1), or completely reinstall the operating system.

Virtual machines have recently gained popularity in this area. The ability to run multiple operating systems on virtual machines, hosted on one physical machine, is extremely attractive in a tool for behavior analysis. In fact, most AV labs and researchers today use virtual machines for quick behavior analysis of unknown and suspicious code. In order to set up a virtual malware research lab, you can use practically any virtual machine software, such as Virtual PC (now published by Microsoft: see www.microsoft.com/windows/virtualpc/default.mspx) or VMWare's Server (www.vmware.com/). There are even a few open source options for braver souls. This offers the opportunity to emulate a real world environment with multiple virtual machines.

In the simplest case, your virtual environment will consist of one virtual machine that will be used as a sacrificial goat. An obvious advantage of virtual machines is easy reinstallation. As the whole hard disk is kept as a file on the host operating system, all you need to do is make a copy of this file and use it to return the system back to a known, safe state. If you want to go even further, most of today's virtual machine programs allow you to set up non-persistent virtual hard disks. In other words, as soon as you turn off the virtual machine, all changes to the hard disk are gone. This type of setup makes virtual machines perfect for behavior analysis.

When emulating a real environment, a typical configuration consists of a sacrificial virtual machine connected to an internal (virtual) network and a router/server virtual machine (typically running some sort of Linux operating system). The virtual router/server machine emulates all real-world services, such as DNS, HTTP, Simple Mail Transfer Protocol (SMTP), and so on. When the sacrificial virtual machine is infected with the malware you are analyzing, this setup allows you to monitor all (internal) network interaction from the virtual router/server, so that you can see exactly what is happening on the virtual victim machine.

Finally, you need to be aware of certain dangers and limitations that come with virtual machines. A bug in the implementation of your virtual machine software might allow malware to escape and infect the host machine. While this is an extreme case, it is by no means impossible. Indeed, a known vulnerability found in VMWare Workstation and GSX products in December 2005, enabled a specially crafted program to execute arbitrary code on the host operating system.

Tools & Traps

Malware That Detects Virtual Environments

A major drawback of this approach is that some malware may detect, or attempt to detect, whether it is running in a virtual machine or not. This has become a pretty common case today, as malware authors are aware of benefits that virtual machines offer for malware analysis. Malware that detects virtual machines will typically just exit upon detection, or perform some benign or innocuous action, making useful behavior analysis in a virtual environment almost impossible. In such a case, you might need to revert to physical machines or use static (code) analysis as a replacement or supplement to the virtual environment.

Themida is an example of a well-known packer that detects virtual machines and stops the executable from running.

Peter Ferrie's paper "Attacks on Virtual Machine Emulators" considers known attacks on VMware and Virtual PC, as well as other virtual machines such as Hydra and Xen (http://pferrie.tripod.com/papers/attacks.pdf).

Behavior Monitoring

Once a suspicious has been executed, you will want to gather as much information about its activities as possible. Whether you are performing this on a virtual machine, or on a physical scapegoat system, the procedures and tools you use for gathering information will be the same.

The goals of behavior monitoring are to gather enough information about changes to the environment that have been introduced by the unknown program you are analyzing. Typical changes that you will want to monitor on Windows operating systems include file access and modification, registry modification, and any network connections established by a malicious program.

Most of the tools for behavior monitoring must be installed in the "scapegoat" or sacrificial virtual machine. You will need the following tools:

- Filemon from Microsoft Sysinternals (www.microsoft.com/technet/sysinternals/utilities/filemon.mspx). This utility will let you record all file activities in real time. It can generate a lot of data, but it will give you a good overview of how the unknown program you are analyzing is manipulating the file system.

- Regmon from Microsoft Sysinternals (www.microsoft.com/technet/sysinternals/utilities/Regmon.mspx). This is a utility similar to Filemon, but it monitors all registry-related activities.
- AutoRuns from Microsoft Sysinternals (www.microsoft.com/technet/sysinternals/utilities/Autoruns.mspx) or HiJack This by Merijn (www.spywareinfo.com/~merijn/programs.php). You should use one of these utilities after you have executed the malware, so as to see if it has modified any of the startup parameters. (Most malware adds itself to the Run registry key in order to be executed after system reboots.)
- Wireshark (www.wireshark.org). Wireshark is a network sniffer. You can run this utility on:
  - The sacrificial virtual machine
  - The machine acting as the router/server
  - The host machine, if you allow the virtual machine to access external sites

These four utilities will allow you to gather a wealth of information about the malware you are analyzing. Just by going through logs created by Filemon and Regmon, you can easily spot typical infection cycles. The AutoRuns utility will show you all programs that are set to start on system boot. Going through this list will require a bit of experience, but just by comparing the known state of the system before with the state of the system after you executed the file, you can easily see what has changed. A word of caution: depending on what the malware actually does, it can subvert all these programs. Some malware authors even go so far as to enumerate running processes and kill any known debugging/analyzing programs.

Finally, Wireshark will offer you details about network traffic. If you are running it on the router/server virtual machine, it cannot be subverted by any rootkits on the scapegoat machine. You will always be able to see all network traffic generated by the scapegoat machine. However, if the malware is encrypting its network traffic, the only value you will get from Wireshark logs will be ascertaining the end points to which malware tries to talk.

TIP

Wireshark and its uses in security are considered at length in "Wireshark & Ethereal Network Protocol Analyzer Toolkit" by Angela Orebaugh *et al* (Syngress).

Forensic Analysis

A commonly asked question related to new malware (apart from what it does once it is executed on a machine), is the nature of its infection vector. This question is even more important for administrators of big organizational networks. In these environments, the infected machines are normally just reinstalled (re-imaged), but without knowing what the initial vector was, you cannot be sure that the infection will not happen again.

Analyzing the malware can sometimes show the infection vector (e.g., a network worm that exploits a vulnerability in the target operating system), but this is not necessarily the case. In cases when malware spreads in multiple stages, it might be difficult to determine the infection vector, as can happen when a downloader deletes itself after downloading malware that initiates subsequent stages in the process of infecting and compromising the target system.

It is often also important to determine whether the infection vector included the exploitation of a previously unknown vulnerability (e.g., a vulnerability in the end-user's browser or e-mail client), or if the infection just happened because the user knowingly downloaded and executed a file.

Forensic analysis of malware is intended to determine the infection vector. In this case, we are not necessarily interested in collecting information that could lead to potential legal actions (although in some cases, especially in big organizations, that will be the case). Rather, our aim here is to collect as much information about the spreading mechanism of the malware.

Tools & Traps

The Changing Face of Forensics

The term "forensics" is traditionally applied to the application of scientific methodology for judicial review. However, recent definitions of digital forensics talk about "digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations" (Digital Forensics Research Workshop. "A Road Map for Digital Forensics Research" 2001. www.dfrws.org/2001/dfrws-rm-final.pdf)

Forensic aims may include identification, preservation, analysis, and presentation of evidence. Digital investigations that are or may be presented in a court of law, must meet the applicable standards of admissible evidence. Admissibility is a concept that varies according to jurisdiction, but is founded on relevancy and reliability.

Collecting Volatile Data

Any program that is executed on a machine, leaves a wealth of information about itself. When investigating new malware, this information can be crucial in determining the initial infection vector and it can also reduce the amount of analysis necessary. This is why the process of volatile data collection is important and different from cases when a computer is seized with the ultimate aim of pursuing some form of legal action. In the latter case, the emphasis is on the preservation of the integrity of evidence, rather than on the pragmatic accumulation of information with the intention of reducing the impact of an incident on your business processes.

When collecting volatile data, our primary goal is to enumerate all current activities on the system. In this phase, you should also investigate machine memory, because all process information is lost after it is powered down to collect the non-volatile data.

Tools & Traps

Forensics and the Law

The following are some of the major issues when it comes to forensic processes for presenting evidence in court:

- Evidential integrity is paramount. Risks from extraneous malicious code, mechanical damage, and accidental modification need to be (as far as is practicable) eliminated.
- Standard practice is to work with a disk image rather than original data, in order to avoid cross-contamination and challenges to its legal validity.
- The chain of custody should be thoroughly documented, to show how, when, where, and by whom it was obtained, what it consists of, and who holds the responsibility for securing it.

Even if you don't anticipate submitting your findings to judicial review, it can sometimes be appropriate to conduct your analysis as if you might be required to. Sometimes an incident can turn out to have a legal dimension that wasn't obvious in the beginning. In any case, if your procedures meet judicial rules for the admissibility of legal evidence, it's not likely that it will be rejected internally to the organization you work for.

Rootkits

The biggest problem when collecting volatile data is the amount of trust you have to put in the infected machine. Malware authors have learned to go to extreme lengths to hide their creations from the system owner, and recent developments and take up of rootkit and stealthkit technology have truly deserved the meaning of the word *stealth*.

One of the first steps when performing a forensic analysis on an infected machine is to determine if there are any rootkits installed on the machine. This does not necessarily have to be done as a first step, but if there is a rootkit installed on the machine, you will have to stop it before collecting volatile data, otherwise it will almost certainly not be complete and you will miss crucial information about the malware.

Are You 0wned?

Rootkit Detection

It is very difficult to detect beyond all doubt whether a rootkit is present on the machine. As a general rule of thumb, always run multiple rootkit detection tools and compare their results.

In some cases, it will be almost impossible to remove a rootkit without rebooting the machine, and this will cause all volatile data to be lost. If that is the case, you will have to rely on non-volatile data and boot the machine off safe, trusted media

Some anti-rootkit tools:

- Sophos anti-rootkit software: www.sophos.com/products/free-tools/sophos-anti-rootkit.html
- GMER anti-rootkit: www.gmer.net/index.php
- McAfee Rootkit Detective (beta at time of writing): http://vil.nai.com/vil/stinger/rkstinger.aspx
- Panda anti-rootkit software (beta at time of writing): www.pandasoftware.com/download/documents/help/rkc/en/rkc_en.html
- Trend Micro's Rootkit Buster (beta at time of writing): www.trendmicro.com/download/rbuster.asp
- F-Secure's BlackLight anti-rootkit software: www.f-secure.com/blacklight/
- Microsoft Sysinternals' RootkitRevealer: www.microsoft.com/technet/sysinternals/utilities/RootkitRevealer.mspx

Rootkit information resources:

- Rootkit: www.rootkit.com/
- Antirootkit.com: www.antirootkit.com/
- "The Root of All Evil? – Rootkits Revealed." David Harley, Andrew Lee: www.eset.com/download/whitepapers.php
- "Hide 'n Seek Revisited – Full Stealth is Back." Kimmo Kasslin, Mika Stahlberg,
- Samuli Larvala and Antti Tikkanen. In Proceedings of the 15th Virus Bulletin International Conference, 2005.
- "ROOTKITS, Subverting the Windows Kernel," Greg Hoglund and Jamie Butler (Addison-Wesley)

There are many rootkit detectors available today that are more or less based on a common technique: they perform a high-level scan of the system using available APIs, and a low-level scan of the system (typically a direct read of the hard disk). Once the system has been scanned, results are compared and if there are any discrepancies, there is a high possibility that a rootkit is present on the system. For this purpose, one of the best tools is again Microsoft Sysinternals' RootkitRevealer.

Another recommended tool is Joanna Rutkowska's System Virginity Verifier (http://invisiblethings.org/tools.html). While still in development, it has a different approach as it compares memory locations of APIs in the active operating system memory with those written in files on the disk.

Using these tools, perhaps along with some of the other tools listed above, will make you more confident that a rootkit is (or is not) present on the machine you are analyzing.

Collecting Process and Network Data

Information about processes running on the infected machine and established network connections, is the most interesting volatile information that we want and need to acquire. If you have successfully established whether you can trust the operating system (e.g., that a rootkit is not present), then it is relatively straightforward to collect these data.

When enumerating both processes and network connections, with time you will get enough experience to spot "strange" instances and anomalies easily. It is, however, generally easier to enumerate network connections. Most users and administrators will be well acquainted with their own, local computer network and the services they are accessing. Besides that, most current operating systems include utilities that allow you to enumerate network connections easily. These operating systems enable you to use the netstat utility or an equivalent, in order to list all network connections and listening services. Below, we list network connections on an infected machine that is connected to an Internet IRC C&C server:

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\User>netstat -ano
Active Connections

Proto    Local Address        Foreign Address          State          PID
TCP      0.0.0.0:135          0.0.0.0:0                LISTENING      1904
TCP      0.0.0.0:445          0.0.0.0:0                LISTENING      4
TCP      0.0.0.0:3389         0.0.0.0:0                LISTENING      1848
TCP      0.0.0.0:9688         0.0.0.0:0                LISTENING      948
TCP      192.168.0.1:139      0.0.0.0:0                LISTENING      4
TCP      192.168.0.1:1043     192.168.100.100:6667     ESTABLISHED    3040
```

From the listing above you can see that there is a connection established from the local address of 192.168.0.1 (the infected machine), port 1043 to the remote machine with the IP address of 192.168.100.100 on port 6667 (which is the default IRC port.) Notice the use of the very useful *–o* option, which allows you to list the PID associated with this connection. This option is, unfortunately, present only on Microsoft Windows XP and Server 2003 operating systems and later. In order to get the same functionality on other, older, Windows operating systems, you will need to install a third-party application, such as Foundstone's Fport (www.foundstone.com/resources/proddesc/fport.html). In this particular case, it was very easy to identify the malicious process as well, since it had a connection established. We were therefore able to see which process caused this behavior, just by checking the PID listed.

In other cases, you will want to inspect processes currently running. While you can theoretically use even the humble Windows Task Manager to do this, we recommend another, more powerful utility called Process Explorer (available from www.microsoft.com/technet/sysinternals/utilities/ProcessExplorer.mspx). Process Explorer lists all active processes on the system, and gives you a wealth of information about them. Just as network connection enumeration becomes easier with practice, with a bit of experience you will be able to easily spot rogue processes in the list produced by Process Explorer. The ability to show the full path to the executable from which the currently running process originates is also a big help. This way, when you see a suspicious process, you can see where it has been started from, and this is likely to give you enough information to determine whether the process is rogue or not. Process Explorer can also show you network connections established by any process (enabling you to confirm whether the data you saw with Netstat corresponds to that shown by Process Explorer). There is also a very useful feature that allows you to print all strings from process memory. This feature is similar to the strings utility we used previously, but this time it works on the memory image of the process, not on a file on disk.

After collecting network and currently running process data, you should have enough information about the current state of the machine to enable you to go forward and collect non-volatile data, in order to determine the initial infection vector.

Collecting Non-volatile Data

Non-volatile data (meaning data stored in files on the infected machine) is easier to collect, since trusting the operating system is less of an issue. The typical approach to collecting non-volatile data includes booting the machine off a trusted (known safe) CD-ROM or other non-writeable medium. By doing this, you can be sure that the underlying operating system has not been compromised by the malware you are hunting.

The following steps depend on actions you want to pursue in the future. If there is a chance that this forensic analysis might involve legal action you should follow proper, specialized incident procedures that you should have in place and written before the incident actually happened. (Best get writing now!) This step typically involves creating at least two images of the hard disk from the compromised system, together with their checksums (the use of SHA-1 checksums is recommended). One of these images is then safely stored so that you can perform forensic investigation on the other image. This way, in the event that the incident ends up as the subject of a court case, there is a chain of custody and it can be proved that the original image has not been tampered with.

If, on the other hand, you are just performing an internal malware analysis and there is no likelihood of legal involvement, you might want to mount the hard disk of the infected machine directly under a live Linux distribution (such as Knoppix-www.knoppix.org), and begin collecting files and data. If you do not feel comfortable working under Linux, you can still create the hard disk image of the infected machine and mount it on a different Windows machine, where you can use Windows tools during the analysis. While this has certain advantages, like being able to scan the acquired hard disk image with your AV directly, be aware that you may be exposing the host machine to malicious software, especially if you decide to execute any files from the acquired hard disk. It is best to have a separate, isolated machine for this purpose.

Determining the Initial Vector

At this point, if you have collected all volatile data, you should have enough information about the infection status of the machine, and you will now be interested in determining the infection vector. Before going into deep technical analysis of the malware, it is always recommended that you approach the user or owner of the machine and ask him or her for as much detail as possible about possible infection vectors. In many cases, the user will at least be able to tell you if he executed a file or casually browsed the Internet. However, do not be surprised if the user breached internal policy (for example, by using a peer-to-peer program) and is hesitant and evasive about what really happened.

The vast majority of infections today happen according to one of the following scenarios:

- The user received an infected e-mail and executed the attachment.
- The user downloaded a file with his Web browser or peer-to-peer software and executed the file.
- The user was infected through a vulnerability in his Web browser.

As the first step, you should try to determine the time the machine was infected. Determining a precise time is not critical here (although it helps), because you will be looking at a time window in which the machine could have been infected. If there is an obvious suspicious program that you detected in the previous step, you should check the time at which that file was modified and then search for any activities on the machine that happened in a narrow time frame before and after the infection. Be aware that malware can change the recorded time associated with its host file, so you cannot always rely on timestamps. Nonetheless, in most cases, the time will be correct.

After you locate all files that have been modified or created near the infection time, you will often have enough information to confirm which of the scenarios described above happened. If the user just executed a file, it is quite possible that will be the only activity recorded, but you might also find additional files that have been dropped by this malware.

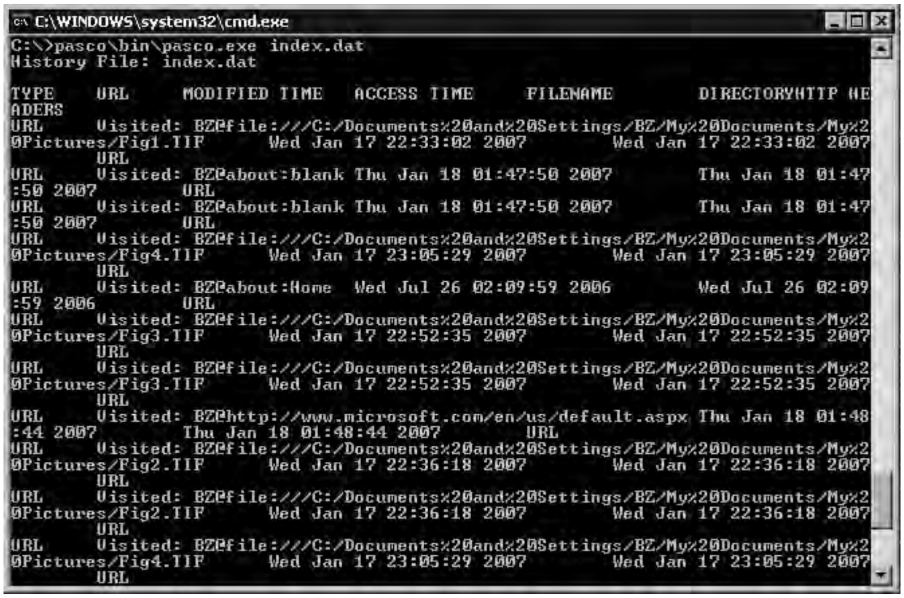
A Lesson from History

If you suspect that the user got infected through a browser (either through a vulnerability in the browser or by downloading and executing a file), you need to analyze browser history and cache files. These files allow you to reconstruct the user's Internet-related activities, unless they have been deliberately deleted. It is very rare for malware to modify the browser history or cache, so if these have been modified, it is most likely a result of a user's action.

While historically Internet Explorer has been riddled with vulnerabilities, other vulnerabilities in Mozilla FireFox, Opera, and so on have also been made public, and malware authors are happily trying to exploit all of them. As you probably already know, all browsers, by default, cache a certain amount of all content downloaded from the Internet, in order to speed up future accesses by the user to the same sites. This leaves, potentially, a wealth of information that can be analyzed, and is extremely helpful when you are trying to determine the initial infection vector where a browser vulnerability has been exploited.

We will start with Internet Explorer as it is the most commonly exploited client on infected machines. By default, Internet Explorer will store all cached files in the directory *C:\Documents and Settings\<username>\Local Settings\Temporary Internet Files\Content.IE5*. There are additional directories inside the *Content.IE5* directory that contain all downloaded files. You can already see that just by inspecting these files, you can get all the information you need about a user's Internet behavior. Typically, just comparing file dates can point you in the right direction. In the event of needing detailed information about a user's Web activities, you can analyze Internet Explorer's history file. This file is located in the *C:\Documents* and *Settings\<username>\Local Settings\History\History.IE5* directory and is called *index.dat*. Unfortunately, this is a binary file, so you cannot inspect it visually, but there are numerous utilities that can parse this file. The most commonly used (and free) one is Pasco by Foundstone (www.foundstone.com/resources/proddesc/pasco.html). Pasco reads *index.dat* files and outputs a tab-delimited history of the user's Internet activities. This file can then easily be loaded into a program such as Excel and analyzed further. In most cases, this will be enough to show you the site where the infection was picked up.

Figure 9.38 Pasco Parsing Internet Explorer History File



```
C:\>pasco\bin\pasco.exe index.dat
History File: index.dat

TYPE    URL     MODIFIED TIME   ACCESS TIME     FILENAME        DIRECTORYHTTP HE
ADERS
URL     Visited: BZ@file:///C:/Documents%20and%20Settings/BZ/My%20Documents/My%2
0Pictures/Fig1.TIF      Wed Jan 17 22:33:02 2007        Wed Jan 17 22:33:02 2007
        URL
URL     Visited: BZ@about:blank Thu Jan 18 01:47:50 2007        Thu Jan 18 01:47
:50 2007        URL
URL     Visited: BZ@about:blank Thu Jan 18 01:47:50 2007        Thu Jan 18 01:47
:50 2007        URL
URL     Visited: BZ@file:///C:/Documents%20and%20Settings/BZ/My%20Documents/My%2
0Pictures/Fig4.TIF      Wed Jan 17 23:05:29 2007        Wed Jan 17 23:05:29 2007
        URL
URL     Visited: BZ@about:Home  Wed Jul 26 02:09:59 2006        Wed Jul 26 02:09
:59 2006        URL
URL     Visited: BZ@file:///C:/Documents%20and%20Settings/BZ/My%20Documents/My%2
0Pictures/Fig3.TIF      Wed Jan 17 22:52:35 2007        Wed Jan 17 22:52:35 2007
        URL
URL     Visited: BZ@file:///C:/Documents%20and%20Settings/BZ/My%20Documents/My%2
0Pictures/Fig3.TIF      Wed Jan 17 22:52:35 2007        Wed Jan 17 22:52:35 2007
        URL
URL     Visited: BZ@http://www.microsoft.com/en/us/default.aspx Thu Jan 18 01:48
:44 2007        Thu Jan 18 01:48:44 2007        URL
URL     Visited: BZ@file:///C:/Documents%20and%20Settings/BZ/My%20Documents/My%2
0Pictures/Fig2.TIF      Wed Jan 17 22:36:18 2007        Wed Jan 17 22:36:18 2007
        URL
URL     Visited: BZ@file:///C:/Documents%20and%20Settings/BZ/My%20Documents/My%2
0Pictures/Fig2.TIF      Wed Jan 17 22:36:18 2007        Wed Jan 17 22:36:18 2007
        URL
URL     Visited: BZ@file:///C:/Documents%20and%20Settings/BZ/My%20Documents/My%2
0Pictures/Fig4.TIF      Wed Jan 17 23:05:29 2007        Wed Jan 17 23:05:29 2007
        URL
```

Parsing Mozilla Firefox's history file is a bit easier, as it stores all information in a text (XML) file, so theoretically you do not need specialized tools to do this. Firefox will store the history file in *C:\Documents* and *Settings\<username>\Application Data\Mozilla\Firefox\Profiles\<random>\history.dat*. The random part will be different on every machine (installation), as Firefox uses it to prevent attacks that rely on starting locally cached data by guessing its full pathname.

TIP

Once you have identified the site that caused the infection, you can try to replicate the infective behavior. If you decide to do this, make sure that you are either using an isolated goat machine that does not have access to any other site or network, or use a safe utility such as Wget for pulling files without risking executing them.

Another place that you will definitely want to check is the Recycle Bin. Normally, deleted files are just stored in the Recycle Bin, which has another nice feature: it stores the exact time when the file was deleted, as well as the original pathname. Foundstone built another useful forensics utility called Rifiuti (www.foundstone.com/resources/proddesc/rifiuti.html) that can be used to analyze Recycle Bin contents.

Case Study: An IRCbot-infected Machine

We will finish this section with a case study of a machine that has been compromised by an IRCbot, a worm which connects to a command and control IRC server, and sits in an IRC channel waiting for instructions from the owner. In most real life scenarios, these bots (zombies) are used either as component systems for Distributed Denial of Service (DDoS) attacks or for sending spam, and it is not uncommon for attackers to use them as their base for future attacks (especially if they can compromise your local network or more remote networks by doing so).

In this case, the administrator has been informed of a machine that is generating high volumes of network traffic and causing problems in its own subnet. The administrator immediately suspects that the machine was compromised, and decides to follow proper forensics procedures until it can be determined what kind of compromise he is dealing with.

The first step in the forensic analysis is to collect volatile data. As we already said above, in order to do that you have to know that you can trust the operating system. Nevertheless, if there is an active attack in progress, as it is the case here, you might want to see if there is anything obviously suspect running on the machine. The first thing the administrator would do in this case is to check all active network connections on the machine using Netstat. The (truncated) output from Netstat is shown below:

```
C:\Documents and Settings\User>netstat -ano
Active Connections

Proto    Local Address       Foreign Address       State          PID
[...]
TCP      192.168.0.1:1044    192.168.200.200:80    SYN_SENT       4111
TCP      192.168.0.1:1045    192.168.200.200:80    SYN_SENT       4111
TCP      192.168.0.1:1046    192.168.200.200:80    SYN_SENT       4111
TCP      192.168.0.1:1047    192.168.200.200:80    SYN_SENT       4111
[...]
TCP      192.168.0.1:5440    192.168.100.100:8555  ESTABLISHED    4111
```

As you can probably see, this looks much like a DDoS attack on a remote site (via port 80, so the Web server is being attacked) and SYN packets have been sent by a process with PID 4111. The administrator logged netstat output in a file, as the volume of output can be pretty big when a DDoS attack has been launched.

At this point in a similar attack, you should disconnect the machine from the network. Although you risk losing some information (for example, whether the attacker was connected to the machine over the network), you still have all the information about network connections logged by netstat. By disconnecting the machine from the network, you will block the DDoS attack as well (at least the part coming from your network).

Now that initial information has been gathered, the operating system should be checked to determine whether it can be trusted. This can be done by running a couple of rootkit detector utilities. In this case, the scan did not detect anything on this machine, so the administrator proceeds with the investigation.

It is obvious that the process with a PID of 4111 caused network attacks and, as can be seen on the last line of netstat's output, it has also established a weird connection to port 8555 on a remote site. By using Process Explorer, the administrator successfully locates this process, which turns out to be a binary named *svchosts.exe*. and located in *C:\WINDOWS\System32* directory. If you are familiar with Windows operating systems (or even read the example earlier in this chapter), you will recognize this as an attempt to hide malware among legitimate system files. Current versions of Windows come with a system process called *svchost.exe*. (Notice the missing "s" character at the end of the main filename; i.e., the part that comes before the filename extension.). Process Explorer's memory string search function is also very useful here, having shown that this is indeed an IRCBot. All of the typical IRC commands are listed (such as */JOIN*, */NICK* and so on) as well as various commands generally associated with DDoS attacks.

After killing the process, all DDoS-related activities immediately stop, so it is clear that the IRCbot is responsible for the DDoS attack and for the network traffic originating from this machine. The executable file can now be archived for later analysis (if needed, just the collection of volatile data provided us with a lot of information about the nature of the compromise). The startup sequence should be checked to ensure that nothing else has been added. In this instance, HijackThis showed only one new process added in the Run registry key, and the executable it started was the IRCbot itself.

At this point, you can decide what to do next. In many cases, when we are talking about a bot installation, we recommend that you simply archive files unequivocally identified as innocent data, and reinstall the operating system and all applications, as you cannot be completely sure what additional malware may have been dropped by the bot after the machine was infected. Besides reinstallation of the system, it is also recommended that you determine the initial infection vector.

As our case study administrator knows the approximate time when the DDoS attack started, he can search the file system on the compromised machine thoroughly, looking for any files that have been created or modified that same day. After checking the results, the administrator locates a small, 5kb file in the Temporary Internet Files directory, with the timestamp of 5 minutes prior to the DDoS attack. Running the *strings* utility on the binary produced no results, and checking with PEiD confirms that the file was packed.

The administrator can now use Pasco to list all Web sites that the user visited. He locates those with an access time close to the DDoS attack, and quickly identifies a suspicious Web site with only an IP address and no name. Finding that the user visited a legitimate site only a second before the suspicious Web site was visited, the administrator cautiously downloads a copy of the legitimate site's HTML Web page, only to find out that it has been compromised and that there was an IFRAME pointing to the site with an exploit. After downloading the exploit Web page, the administrator is able to identify the exploit that has been used; Internet Explorer on this user's machine has not been fully patched.

Notes from the Underground

JavaScript Obfuscation

Almost all browser exploits today use either JavaScript or VBScript. In order to make detection more difficult for inline Web AV programs, malware authors obfuscate their JavaScript programs. It is typically easy to spot JavaScript obfuscation. It will always consist of a function with some mathematical operation and the function call with a big, obfuscated input argument. This JavaScript is automatically evaluated and processed by a Web browser.

In order to safely analyze obfuscated JavaScript without risking infecting the machine, you can use a command-line JavaScript interpreter called SpiderMonkey (www.mozilla.org/js/spidermonkey). SpiderMonkey is part of the Mozilla project, and is the same interpreter that is used in the Firefox browser. When you use the command-line interpreter, the evaluated JavaScript code that the browser will execute is not executed, as SpiderMonkey does not in itself provide a host environment for JavaScript: it will just be printed on your screen so you can further analyze it. One caveat: JavaScript interpreters are different in Internet Explorer and Mozilla FireFox, and there have been cases when malware authors abused this to prevent analysis in Mozilla FireFox.

By analyzing all activities related to this incident, the administrator was able to determine the initial vector (and replicate its behavior in a safe, isolated environment). While determining whether a single patch was missing is relatively easy, by identifying the exact infection vector, our administrator can be sure that his network is now protected from this attack.

TIP

There's another interesting case study in Syngress's "Botnets: The Killer Web App" by Craig Schiller and Jim Binkley et al, in the chapter on botnet detection. (See www.syngress.com/catalog/?pid=4270)

Summary

With the knowledge that you gained in the 101 section of this chapter, you should be much better armed the next time that you get that phone call from your SOC. We would strongly recommend that you practice using these tools until you know how to use them perfectly. The quicker and more proficient you become in the proper use of these tools, the sooner the outbreak will be stopped, and the fewer infected machines you'll have to deal with during the aftermath.

The tools mentioned in this chapter are also by no means the only tools that will accomplish the needed tasks. They are tools that are used by the authors and that are recommended for your use, but there are newer and better tools being created each and every day. A good place to watch for new tools, or perhaps tried and true tools that you didn't know existed, is the listing found on http://sectools.org/. The site maintains a listing of the top 100 security tools. It is well worth looking over the listing, downloading a few, and trying them out. You may just find a new favorite tool that you'll add to your emergency jumpkit.

In the last couple of years, malware has become increasingly difficult to analyze and remove. Most malware authors today are not students who want to show how good they are in programming, but organized crime gangs that seek profit. They go an extra step in making it difficult to remove their malware, to hide it, and to make reverse engineering more complex. This is why the only sure way to deal with infected machines is to reinstall them. However, before doing that, you should make sure that you know what the infection vector was, because otherwise you probably face the same re-infection in the future.

A prepared and tested incident response plan is a must for every organization today. Malware incidents will happen, no matter how much you have invested in protection. When previously unknown malware strikes your organization, you will have to assess the impact and decide on the countermeasures. This can be very difficult in the first few hours of malware spread, as most AV vendors will not yet have definitions and malware descriptions.

By analyzing malware when you need to, you will be able to assess the impact (and the threat) correctly, and ultimately decide on money spent by your organization on damage recovery which can range from reinstalling the infected machine to dealing with stolen intellectual property or customer data.

Solutions Fast Track

Anti-Malware Tools of the Trade 101

- ☑ The first step to take when an incident is suspected, is to isolate a problem machine so that the only remote system it can access is one from which you plan to run your analysis.
- ☑ This should be possible using firewall rules or Access Control Lists (ACLs)

The Basics – Identifying a Malicious File

- ☑ The PSTools package contains a number of extremely useful tools for the administrator's jumpkit. psexec.exe is particularly useful for quickly launching a remote console and running programs remotely.
- ☑ The VNC system allows desktops to be shared and is platform-independent, and therefore, it can be particularly useful in a mixed environment.
- ☑ Once you have a remote control shell open, you can run other tools locally to the machine under analysis: for instance, the other PSTools packages and netstat.

Process and Network Service Detection Tools

- ☑ Three particularly useful tools for analyzing processes and network services are FPORT, tcpvcon, and Handle. FPORT is similar in function to netstat, but includes some significant enhancements.
- ☑ tcpvcon is one of the very useful tools available from www.sysinternals.com, and traces network processes back to the originating pathname.
- ☑ Handle offers rich output on all types of extant handles, including ports, registry keys, synchronization primitives, threads, files, and processes.
- ☑ Archiving sample files with a password ensures that malware samples aren't executed accidentally, or inadvertently removed or modified by security programs.

Web Based Inspection and Virus Analysis Tools

- ☑ It's better to submit suspicious samples directly to your own AV vendor as soon as possible, rather than rely on Web sites that are running multiple online scanners to forward samples in a timely fashion. Sites like Jotti, VirusTotal, and Virus.Org can be very useful as a supplementary check in the early stages of identification, but there are risks in relying on totally trusting their results. You may also find it useful to try samples against the online scanners some anti-virus vendors offer.
- ☑ Some sites also offer sandbox analysis of submitted samples. Using a service like this gives you the opportunity to test a sample and take appropriate countermeasures even before carrying out comprehensive analysis in-house.

Using Packet Analyzers to Gather Information

- ☑ Sacrificial goat machines should not be connected to a production network, for safety reasons, but should be configured so that they closely resembles your production machines. Using a package such as VMware allows you to run a goat machine within a virtual environment.
- ☑ A packet analyzer such as tcpdump, windump or Wireshark is a must-have for your toolkit. It enables you to inspect information traversing the network to and from the machine being inspected.
- ☑ As well as capturing and analyzing packets, Wireshark reads PCAP-formatted output files from tcpdump, windump, snort et al.

Examining your Malware Sample with Executable Inspection Tools

- ☑ Strings is a program that searches a file for strings of Unicode or ASCII characters, which may provide clues to IRC server names, IP addresses, and so on.
- ☑ Not all useful tools are open source or freeware. VIP Utility is a commercial open sandbox program that monitors an executable, using a combination of heuristics and behavior analysis outputting somewhat similar results to some of the Web-based tools described earlier.
- ☑ InCtrl5 takes a snapshot of files, registry keys, text files, and INI files before and after the file being analyzed is executed.

Using Vulnerability Assessment and Port Scanning Tools

- ☑ The most popular and best known port scanning tool is probably Nmap. Its default command-line options can be supplanted or modified for a more aggressive or more passive scan.
- ☑ Superscan offers similar functionality to Nmap's, but with a GUI front end (Nmap also has an optional GUI).
- ☑ The most widely used vulnerability scanner is Nessus, which is flexible and full-featured, yet easy to use. It includes its own scripting language.

Advanced Tools: An Overview of Windows Code Debuggers

☑ A debugger helps you to reverse-engineer a suspicious file. WinDbg is a basic but competent component of the "Debugging Tools for Windows."

☑ OllyDbg is a highly-rated, very popular and full-featured 32-bit assembler level debugger, highly suitable for taking apart suspected malware in the Windows PE format.

☑ DataRescue's IDA Pro is a reasonably priced best-of-breed debugger/disassembler that can handle a huge range of executable formats.

Advanced Analysis and Forensics

☑ The sheer volume of new variants of bots and other malware associated with today's explosion of spam and other breaches, compromises the ability of AV vendors to deploy timely reactive and proactive countermeasures. This in turn puts a strain on in-house security administrators who often have to analyze malicious code themselves, in order to maintain services and deploy countermeasures specific to their organization.

☑ Depending on how detailed an analysis is required, you might want to use static analysis, dynamic analysis, or a hybrid approach. Malware can be analyzed statically or dynamically. Static malware analysis offers in-depth information about the target program, but can be very time consuming, whereas dynamic analysis can be quicker, but may be thwarted by malware using a range of deceptive techniques.

Static (Code) Analysis

☑ Static analysis involves reviewing, more or less by hand, the actual code of a program in order to determine what it does. To review code, we must be able to disassemble it, and to be well-acquainted with the inner workings of assembly language and machine code.

☑ The biggest obstacle in static analysis is the use by malware authors of runtime packers. Packed malware must be unpacked in order to analyze the raw code.

☑ Runtime packers are used by malware authors to compress the size of the executable, shorten the loading time, and, most significantly, to obfuscate the code and make it harder to analyze.

☑ Once the file has been unpacked, it can be examined with a range of utilities, from simple tools like *strings* to top-flight utilities like IDA Pro.

Dynamic (Behavior) Analysis

- ☑ Dynamic or behavior analysis can, in some instances, be much faster than static or code analysis, yet yield almost as much information. It involves monitoring the behavior of the suspect program while it's executed in order to ascertain what it really does.
- ☑ Dynamic analysis needs to be undertaken in isolation. Nowadays, the use of virtual environments makes it easier to simulate a real environment without significant risk to a production environment, but may not be feasible for analyzing malware that checks whether it's running on a virtual host.
- ☑ A number of tools need to be installed on a sacrificial virtual machine, and are likely to include Filemon, Regmon and AutoRuns from Microsoft Sysinternals and the WireShark network sniffer.

Forensic Analysis

- ☑ Documenting and testing an incident response plan is vital. This has to be done before the incident happens, and documentation and procedures need to be reviewed and tested both routinely, and in the light of experience of specific incidents.
- ☑ Volatile data gives you information about malware's behavior when it's active on the machine.
- ☑ Collecting volatile data does not justify leaving the machine actively attacking other machines/network.
- ☑ Always make several copies of the infected hard disk so you preserve original information while working on a copy.

Frequently Asked Questions

Q: Where can I get more information about issues like batch file syntax and command line redirection using >> and >, for instance?

A: You can still get some of this information from Web sites like http://www.computerhope.com/msdos.htm and http://en.wikipedia.org/wiki/Batch_file. Note, however, that information that may be correct when it refers to specific DOS versions may not be accurate for programs run at the DOS prompt in modern Windows operating systems, where the DOS shell is more emulation than core system. For such systems, it's worth examining the help files for Windows as well as the program help and summaries available using the help options at the DOS prompt. In brief, not everything works as it did in real MS-DOS.

Q: What does 127.0.0.1 mean?

A: It's the IP address that denotes the machine you're using. In other words, it's used for a loopback connection between a machine and itself. This has many diagnostic uses, though it can also be used maliciously.

Q: Why do you use an obsolete version of PKZIP for archiving examples, rather than modern tools like WinZip, or one of the many freeware tools?

A: There's nothing to stop you using WinZip, Stuffit and so on, or even Windows' own compression/decompression. PKZIP is convenient because, like many of the security tools used here, it's a command line tool that can be easily incorporated into scripts and batch files, and it's reasonable to expect it to conform absolutely to the ZIP archive standard. If you use other compression formats for submission of samples, for example, you may not be able to assume that they'll be read or decrypted properly at the other end, depending on what software they have access to. In particular, you may need to use standard ZIP 2.0 passwords, not one of the more secure but less standard alternatives such as AES encryption.

Q: If I can use an online virus scanner, why do I need virus scanning on my machine?

A: Online scanners, although they may have more up-to-date virus definitions than desktop scanners, aren't in all respects as reliable in terms of detection mechanisms. Nor do they offer the sort of real-time protection that an on-access scanner on your system can, or the configurability.

Q: What's all this goat stuff?

A: The use of the sacrificial goat metaphor in AV and anti-malware literature probably stems from the practice of hunting tigers by tethering a goat in tiger country and waiting for the tiger to come along in search of an easy meal. (There are many instances of goats used in sacrificial rites, of course.) Another use of the metaphor in this field is the so-called goat file, a file intended to attract the attention of specific types of viral program. The term scapegoat also has a sacrificial origin (Leviticus 16:22, the Talmud and so on), though the use of the term in the context of malware analysis is less common.

Q: What network ports should be open in general?

A: That's too complex a question and too specific to how a particular operation works. A sensible approach is to block everything except services you know you need. That can be very time consuming to implement properly, though. Of course, when you're specifically looking for malicious traffic, you need an environment with a more laissez faire approach.

Q: What platforms are appropriate for malware analysis?

A: Obviously, you need actual or virtual machines representing the vulnerable platforms in your production environment, or your customers'. You might also want access to platforms that aren't believed vulnerable to the same types of threat, not only for safety, but to lessen the risk of inaccuracies introduced by spoofing.

Q: Doesn't forensics always deal with criminal action and legal cases?

A: Traditionally, yes, but the term is now very commonly used in digital forensics and computer forensics to refer to detection and analysis for non-judicial purposes. You should note that some of the approaches discussed here are not compatible with "traditional" forensics for legal purposes, unless they're carried out in the context of strict adherence to standards relating to the preservation of evidential integrity. In other words, if you're likely to have to present evidence that can be challenged in court or elsewhere, you have to be scrupulous about preserving the chain of custody, documentation, and so on, in order that your evidence remains admissible. That's a fascinating subject, but it's out of scope for this chapter. There's a reasonable overview at http://en.wikipedia.org/wiki/Computer_forensics, if you want to start exploring it.

Q: How can you be sure you aren't being fooled by rootkits and such?

A: There are many ways of detecting known and unknown rootkits, though you can fool some of the detector programs some of the time. It's unlikely that there will ever be malware so effective that it can never be detected, but you have to live with the fact that there is always a risk that some of what you're seeing is what some malware wants you to see.

Q: What is packing?

A: Packing is a way of reducing the size of compiled program code and obfuscating it. It is most commonly used by malware authors to increase the complexity of reverse-engineering their program.

Q: Can a process dumper be used on all malware so we do not have to unpack it?

A: No, some advanced packers, such as Armadillo, will unpack only parts of the code and will keep the rest packed in memory. When that part of code is needed it will be unpacked. This means that if such a process is dumped, only a small part will be visible.

Q: Why can some malware not be debugged with OllyDbg?

A: Some malware authors will implement various anti-debugging mechanisms to make analysis more difficult and time consuming. In cases like this, you will either need to use Microsoft's kernel debugger or disable the anti-debugging part of the malware.

Q: Can virtual machines always be used for behavior analysis?

A: No, there are packers and malware that detect the presence of a virtual machine and will either exit or run a benign function. In cases like this, you will have to either statically analyze the malware or run it on a physical machine.

Q: How do rootkits affect volatile data collection?

A: Rootkits can be configured to hide certain network activity, files on the disk, and registry keys. If a rootkit is present on the machine you are analyzing, and you have not disabled it, you will most likely not be able to see any malware-related files and activities.

Q: Do browsers always leave cache/history traces?

A: No, a user can manually delete cache contents and browser history.

Q: How can I de-obfuscate JavaScript files and see what they do?

A: You can run them through SpiderMonkey, a command-line JavaScript interpreter. Additionally, you can find the line which evaluates the output and disable or change it to display the data instead of evaluating it.

Chapter 10

Antimalware Evaluation and Testing

Solutions in this chapter:

- Antimalware Product Evaluation
- Evaluation Checklist
- Antimalware Product Testing
- Sources of Independent Tests and Certification Bodies

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

David Harley and Andrew Lee both desperately wanted to write this chapter. After all, we've both been responsible for security product evaluation for large organizations as consumers, and for product testing as researchers. Testing is a particularly hot topic among AV professionals. For vendors, of course, it's their bread and butter: every bad review has a potentially damaging impact on their sales. However, there aren't many people in this community whose blood *doesn't* boil when they see yet another clueless comparative review, or repetition of yet another misconception about what a product range can or can't do, and we often get the urge to share our prejudices on the subject. So who was going to get to let off steam?

In the end we resolved the "conflict" by sharing the load. After all, while most consumers are influenced by test reports, very few of them carry out their own tests. Besides, we figure that a significant proportion of our intended audience will have to take into account a whole load of factors, not just detection performance. So, in the first sections, David considers evaluation criteria in general, while Andrew discusses the finer detail of detection testing in the third section. Finally, Andrew looks at some of the more capable and trustworthy testing organizations.

Wouldn't it be nice if we could give you the answers on which antimalware packages you should be using, and how to configure and use them to the best advantage? Well, actually, that's what a great many consumer level security books try to do, usually with a few screenshots from the current McAfee and Symantec ranges. Sadly, this chapter will have to remain resolutely vendor-independent (our own links with the industry notwithstanding.) There are a number of other very capable packages, and while some of the core technology is very similar between products, the interfaces can be very different even between individual products in a single vendor's range, reflecting the very different functionalities between them. There's a great deal of difference between a no-cost evaluation product (or a free-for-home-use version), and a full-blown multi-platform enterprise edition with central console management and cascading staging servers. In any case, security products (especially AV and its siblings) change frequently, and sometimes very dramatically, so the chances are that by the time you see this, even the product you use (if we happened to focus on it) would be near unrecognizable from our 16 pages of screen dumps. So we'll content ourselves with what is essentially a checklist of issues to consider when you evaluate, rather than trying to supply a one-fits-all prescription for a best-of-breed solution.

Actually, there isn't, in a sense, much difference between anti-virus (AV) products (shock! horror!). They all detect much the same range of in-the-wild viruses (see the testing section to learn more about what that means). Unfortunately, over the past few years (as we may have mentioned before), viruses have become less of a problem. Not that they aren't a bad thing when they happen to you, but they're only a (shrinking) percentage of the total malware problem. So your choice of product is affected by a whole range of subsidiary detection issues; that is, how effective is it with non-viruses.

How do we test AV software for effective virus detection? Top players in AV testing like Virus Bulletin, ICSA Labs, West Coast Labs, and the AV Test Centre (see the final section of this chapter for details of these testing organizations) use meticulous testing procedures and large, carefully maintained collections of malicious software (malware). Spurious samples are carefully weeded out, and supplementary techniques such as large, carefully vetted false positive (FP) test sets are used. Testing procedures are scrupulously planned, documented, and followed.

However, such tests take time and are technically demanding. They are also difficult to implement without the cooperation of the AV industry, where samples are normally only shared between trusted individuals. The expense entailed means that detailed results may not be readily available to non-subscribers. Sometimes the service is funded by vendors, to the disadvantage of small vendors and community projects.

Many recommendations reported in magazines and even in security circles are informal, based on the apparently trouble-free performance of a live installation. Variables such as configuration and the quality of any test set used have to be taken on trust. As a result, products that display low detection rates of commonly seen viruses can be perceived as more effective than products that stand up much better to formal testing. And if viruses are only part of the problem, how do we test for all those Trojans, backdoors, robots (bots) and Uncle Tom Cobleigh (http://ingeb.org/songs/tompears.html) and all? Read on...

Antimalware Product Evaluation

Some of the most important issues for those responsible for enterprise countermeasures against malware are as follows (not necessarily in order of importance):

- Configurability
- Cost
- Ease of use
- Functionality
- Performance
- Support issues

Tools & Traps

Approaches to Evaluation

There are a number of common approaches to evaluation.

- **Comparing reviews from non-specialist magazines.** Frankly, this is pretty hit-and-miss. Few non-specialist journalists understand the malware problem in general very well (either the technology or the countermeasure strategies and technologies), let alone the booby traps in detection testing. (There are, of course, very competent journalists like Larry Seltzer who've done good work in this area.) Testing methodology is rarely described, especially if detection testing is outsourced. (Outsourcing can be a very responsible way of handling detection testing, but only if the testing organization is competent!) I'll leave it to Andrew to discuss the confusion and misleading conclusions that can result from poor testing. That apart, the reviewer may choose to focus on more subjective aspects. This approach has its own pitfalls: the issues that concern systems administrators or security managers and directors may not be obvious to a non-practitioner. And, sadly, there are still such tests where the editor's choice is suspected of being unduly influenced by the advertiser roster.
- **Reviews in specialist magazines such as Virus Bulletin, or from reputable testing organizations such as the testing facilities at Hamburg and Magdeburg.** (We include a list of dependable testing organizations at the end of this chapter.) These facilities tend to focus mostly on detection (including variations such as false positive testing), rather than a full range of features.
- **Testing within your own environment, or a reasonable facsimile thereof.** This is the best way to test rollout, installation and configuration, compatibility, network performance, update and upgrade distribution mechanisms, rollback, and so on. If you have the resources and expertise to go to detection testing with live viruses, you must use a simulated or isolated environment, using the sort of tools we discuss elsewhere.

If you don't have the time and resources to test in-house (not that rare a scenario), you have two main alternative resources: discussion with peers who have experience of the products on your shortlist (this is one of the advantages of belonging to AVIEN!), and discussion with the vendors (but be sure you talk to techies as well as sales droids!)

Configurability

My guess is that this is not the most pressing item on the average procurement team's agenda. But perhaps it should be. Configurability, or the lack of configurability, has a direct bearing on the functionality of an anti-malware package or suite of packages. It's unlikely that a given package will come out of the box ideally configured for your purposes and environment. Indeed, the fact that you're reading this book suggests that you will want to find out exactly how the package can be configured so that it fits your purposes rather than the preconceptions of the vendor. Since malware management invariably entails a trade-off between draconian security and realistic usability in terms of speed and transparency, you'll need to sort that one out yourself. Here are some considerations to keep in mind:

- Can you lock down the configuration so that end users can't "fiddle"?
- Can you run and maintain preconfigured installations over the network?
- What are the implications of mixed operating systems and environments in your enterprise? (Is the package available for all servers and workstations, and how well do different versions integrate?
- Are updates push or pull, and can they be staggered if necessary to avoid undue pressure on the network? Can they be cascaded through local staging servers?
- Are workstations checked for the currency of their protection by server hosted software or a central console, or both?
- Mobile workers, how are laptops and other mobile devices integrated?
- Can auto-updating mechanisms be supplemented or replaced by manual mechanisms (e.g., login scripts and batch files) if necessary?
- How does the system handle the need to rollback to a previous version in case of a problem?

Cost

In a previous book ("Viruses Revealed" by Robert Slade, David Harley and Urs Gattiker, published by Osborne, but now out of print), we ruefully suggested that the commonest evaluation and procurement methodology is along the lines of:

- Which products have I heard of?
- Which one is the cheapest?
- I'll take it.

Guess what. There's a lot more to cost than unit cost. Distribution, maintenance, support, and education are hard to quantify but need to be taken into account. Single vendor solutions across a range of machines across the enterprise offer many advantages (coherent training and support, packaging deals, product integration), the more so when they cover a range of software types. They may, however, also entail a single point of failure (SPoF). Multivendor solutions are safer (especially at a time when malware-specific detection by "signature" is less effective through sheer glut), but inevitably entail extra costs. Premium support (24/7 helpdesk, paged alerts, onsite engineer callout), and generous SLAs tend to attract premium prices.

Testing administration and maintenance issues are dependent on the organizational structure and philosophy, resources available, and approach to security architecture. They have a major bearing on Total Cost of Ownership (TCO), however, and you must allow for these costs, as well as anticipated additional costs such as incident and other problem management. Don't forget opportunity costs where other staff are unable to work until an issue is resolved, or where you have to use staff not principally employed for anti-malware deployment and maintenance.

- Deployment definitions updates and product upgrades or cross grades. Factor in staff distribution and monitoring, reduced performance during update, notification, and resolution of problems (e.g., the impact on the service desk)
- Incident response.
- Dealing with false positives.
- Log analysis.
- Reviewing quarantined objects, where applicable.
- Helpdesk queries, from routine maintenance issues to incident notification.
- Backup, recovery, disinfection, and reinstallation time.

TIP

If unit cost doesn't equal total cost of ownership, what does? One much used-formula goes like this:

Total Cost of Ownership = License Cost + Extras + Cost of Evaluation + Cost of Administration

Translating this into real numbers (or even into a spreadsheet) is left as an exercise for the reader. ☺

Tools & Traps

The EICAR Test File

It doesn't seem appropriate to write a chapter on evaluation and testing without mentioning a standard malware testing resource.

The EICAR test file (usually encountered as eicar.com) is not a virus, but a PC assembler program constructed so that it can be typed in with a plain American Standard Code for Information Interchange (ASCII) editor. If the program is actually executed it displays the text *EICAR-STANDARD-ANTIVIRUS-TEST-FILE!*.

To use the EICAR test string, type or copy/paste the following text into a test file:

X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*

An article at the EICAR test site explains the EICAR test file: www.eicar.org/anti_virus_test_file.html.

There's some discussion of the use of eicar.com in the alt.comp.virus FAQ, some of which is reproduced here. (Since I wrote this bit I have no compunction about reproducing it.)

"The EICAR file isn't an indication of a scanner's *efficiency* at detecting viruses, since 1) it isn't a virus and 2) detecting a single virus or non-virus isn't a useful test of the number of viruses detected. It's a (limited) check on whether the program is installed, but I'm not sure it's a measure of whether it's installed correctly. For instance, the fact that a scanner reports correctly that a file called eicar.com contains the EICAR string, doesn't tell you whether it will detect macro viruses, for example. In fact, if I wanted to be really picky, I'd have to say that it doesn't actually tell you anything except that the scanner detects the EICAR string in files with a particular extension.

The string is supposed to trigger an alarm only when detected at the beginning of the file. Some products have been known to 'false alarm' by triggering on files which contain the string elsewhere.

Ease of Use

In a way, detection is a minor consideration. If the product isn't either totally transparent to the user or phenomenally easy to use and difficult to misuse, it's a dead duck, however good its detection is. The end user usually wants protection that requires no input or decision making from them, and has no impact on performance. Hmmm. Of course, administrators would like that too. In real life, it's more feasible to transfer the administration burden from the end user to the administrator, using preconfigured distribution and maintenance packages and server/console-based administration.

If end users or, at any rate, end sites have to take some responsibility for their own AV arrangements, this necessitates ready access to documentation, training courses, and detailed guidelines, supplemented by first-line and second-line support.

Functionality

As the first decade of the 21st century grinds towards its close, the range of functions that add up to a full-blown antimalware package has widened immeasurably. Since the beginning of the decade. It's no longer enough to talk about AV, or even antimalware. In a mixed blackhat economy, hobbyist virus writers are less important than career criminals, and detection of malicious code is all mixed up with the detection of various types of spam (e.g. bot seedings, e-mail worms, phishing messages), using traffic analysis, e-mail textual analysis, Intrusion Detection and Protection technology, and a huge variety of network analysis and reporting tools, as described elsewhere in this book.

We can't realistically address the integration of solutions for all the types of malicious code (viruses, worms, bots, various Trojans, spyware, adware, and so on) that now preoccupy the savvy security administrator. There are just too many combinations and permutations. Not, at any rate, in this short section, though the "Defense-in-Depth" chapter does address this aggregation of issues.

Performance

Did I just say that detection is minor? Only in a scenario where you've already lost the battle before the product has the chance to exercise its detective muscles. A scanner that doesn't detect malware is definitely lacking something. Here's a breakdown of a range of malware types you might want to consider, though we go into much more detail about detection testing in the next section.

- In the Wild (ItW) malware, defined by inclusion on the WildList
- Boot-sector and partition-sector infectors, usually including file and boot multipartite viruses; old-fashioned DOS (MS-DOS, PC-DOS) viruses and infectors for obsolete versions of Windows
- Macro viruses including WordBasic macros and Excel formula viruses.
- Miscellaneous script viruses
- Collection viruses (zoo viruses) of all sorts
- Polymorphic viruses
- Mass mailers/e-mail worms
- Bots (yes, we know that covers a multitude of sins – see the "Big Bad Botnets" chapter)

- Distributed Denial of Service (DDoS) agents
- "Pure" worms (mostly network worms)
- Trojans, including destructive Trojans, password stealers, backdoors, and anything else that you can think of that fits the Trojan label
- Virus kits and generators
- Rootkits
- Cross-platform malware, including latent viruses and heterogeneous malware transmission issues
- Spyware
- Adware
- Jokes
- Games
- Intendeds, corruptions, miscellaneous non-viable malware
- Possibly unwanted applications/programs, greyware

NOTE

"Latent malware" is usually defined as malicious programs that have not yet been executed in the environment under examination: malware detected and blocked on arrival as mail attachments for example, or encrypted binaries that can't be scanned routinely until they're actually executed. Malware in a host environment where it *cannot* be executed is an instance of heterogeneous virus transmission. It may be passed from them onto a system on which it *can* be executed.

Scanning speed is, perhaps, most easily tested more or less informally for on-demand scanning, since it's conceptually simple to get meaningful comparative results on a standard configuration. In a professional testing environment, the configuration will often be minimal compared to the average, cluttered end-user desktop. However, where whole-system scans are regularly scheduled during normal use to supplement on-access scanning, this may have a significant impact on overall machine performance. This impact can vary dramatically according to local configuration and environment, and the same scanner may behave very differently on different sites. This variation is not fully catered for by testing organizations, arguably, it can't be, where it's not practical to simulate a "normal" end user machine too closely. You might, therefore, want to test these factors in-house on "live" production machines, though we don't recommend that you do this with live malware. The same considerations apply with on-access scanning, but more so. Issues for scan speed testing include:

- How long it takes to load on-access scanners at boot up, compared to the baseline speed of an unprotected system.
- The time it takes to load and run an on-demand scan under a variety of configurations and conditions.
- The time to load programs and documents with on-access scanning enabled.
- Does the scan finish before/interfere with backups?
- Configuration options, for example:
  - Speed scanning versus "deep" scanning
  - Executable versus all files scanning
  - Scanning (or excluding from scanning) specific ranges of objects (files, folders, drives)
  - Heuristic levels on a scale from aggressive to minimal. See "Heuristic Analysis: Detecting Unknown Viruses" by David Harley and Andrew Lee at www.eset.com/download/whitepapers/HeurAnalysis(Mar2007)Online.pdf for more consideration of this issue. See also Peter Szor's book "The Art of Computer Virus Research and Defense" for a much more technical look at scanner internals.
  - Integration with some form of checksumming/integrity checking.
  - Options for scanning encrypted objects: routine scanning of all encrypted objects is not technically feasible, but beyond that truism, there's a wide range of possibilities for dealing with known encryption methodologies (e.g., *.zip* encryption.)
  - Options for excluding "trusted" areas or media.
  - Targeting of specific object types for scanning.
  - Targeting of specific malware types to be scanned for.
  - Autodisinfection versus prompting (disinfect, delete, quarantine or ignore?) Clearly, there's an issue around education and user responsibility here.
  - Log management.
  - Alert management.
  - Automatic reporting.

Support Issues

How effectively do the service desk and other support staff deal with actual problems? Well, as variably as in-house support teams, I guess. Unfortunately, it's rare for product reviews to include support testing, though it has been done from time to time. Historically, this has usually been along the lines of describing some symptoms and seeing if the helpdesk identify the probable cause correctly: not invalid, but it only covers a subset of the data you ideally want to gather. You actually want to see two areas covered adequately by support services:

- Product knowledge:
  - How do I do this?
  - Why can't I do that?
  - What does this error code mean?
  - What does that sub-program do?
  - When will your product add this feature?
  - Are you aware of this problem with the product?
- Malware knowledge:
  - What does this malware actually do?
  - Do I need to worry about it?
  - Is this warning a hoax?
  - Do I need to take any extra steps to deal with this problem?

Realistically, the quality of support is a "suck 'n' see" issue. Some vendors are aware of the importance of this issue, and include access to their support services in enterprise evaluation packages, but how well the service traverses the morass of Service Level Agreements (SLAs), escalation protocols, first and second line handover, closure and follow-up, and so on, is most accurately assessed over the long haul.

One trap to look out for: contract negotiation is by no means a purely technical issue, but it's not purely administrative, either. Liaison between technical staff and contract management staff will spot more potholes in the run-up to adopting a product or service than leaving it in the hands of one or the other. Many of the issues are the same, whether your administration is in-house or outsourced, so you may find our chapter on "Perilous Outsourcery" has some bearing on the contract negotiation issue. Here are some issues that you might want to think about:

- Technical and end-user documentation, and general information sharing.
- Monitoring the effectiveness of incident management.

- Incident reporting from the provider to the customer, and the compilation of statistics. I remember with no great fondness a service that included bombarding the mail servers with messages incorporating the EICAR test file, allegedly as a strategy for monitoring performance, and then included those detections in the monthly report. Another trap to beware of: imprecision about what information you are allowed to share with end-users, top management, and so on.
- How well and quickly the vendor reacts to new threats and passes on information accordingly, and the quality of that information.

Upgrades and Updates

Major issues:

- Bugs and vulnerabilities should be dealt with in good time and adequately documented.
- Patches should be easily accessed and applied, and supplied in a secure fashion.
- The distribution of definitions updates should be reliable, as frequent as necessary, and as automated as possible.
- The vendor's quality assurance procedures and rollback procedures should be solid.
- Interim drivers should be readily available (even in a beta version, if necessary) between scheduled updates, where necessary.

Information Flow and Documentation

We could write a book on the issues around information flow and documentation in AV and antimalware evaluation. Actually, we very probably will. In the meantime, though, here are a few key points to think about.

Expansion and development of a product range is not just a marketing issue. Antimalware technology has to change fast to adapt to changes in the threat landscape, so information about forthcoming products, beta programs, and evaluation packages are important. More generalized security briefings are a good measure of the vendor's commitment to the common good. Personally, I'm always more impressed when a vendor manages to present a good guru-to-marketroid ratio. In-depth product training and security training is generally an extra cost, but it's a useful value-add. After all, if your vendors of choice are sound, they probably have skills and information that you don't have. (See Chapter 1 for more consideration of vendor training.)

Security software is an imperfect technical solution to a growing, mutating, and not-exclusively technical problem. It's a good thing if the documentation relating to it is at least accurate in respect of the product features, more so if it's equally accurate in its consideration of the threat landscape it addresses.

WARNING

You'll have gathered from Chapter 1 that I believe that the AV industry in particular gets a raw deal from the media, their customers, "instant experts," and even other security vendors. Actually, some of the brightest, most knowledgeable people I know work in the AV industry. But the threatscape that security managers have to cope with nowadays is far wider than AV, and none of us know everything about everything. Absolute trust in your security vendor is a luxury the savvy administrator can't afford.

Here are some issues you might want to think about in terms of vendor information sharing.

- How do you assess the accuracy of the information supplied to you? It would help to be an expert in your own right. If you're not, you can at least avail yourself of access to social networks like the Anti-Virus Information Exchange Network (AVIEN) and the Anti-virus Information and Early Warning System (AVIEWS) where some very informed people are always ready to share information and expertise with others.
- Is their online information up to date and accurate?
- Does the database list all the individual threats and classes of threat that concern you, even if the package itself may not deal with all those classes?
- How useful is the information? Some technical information is of academic interest only, whereas other detail may be essential, and the compilers of the information are not always best qualified to discriminate.
- How comprehensive is their documentation? Is it clear, and does it fall into the trap of "roll your own" terminology without reference to standard resources and generally accepted basic concepts?
- Does it include contact information and other resources, and is it effectively indexed? Does it include advice over and above "Use our product"? Can you use it to supplement internal resources such as policy formulation and additional tools for deployment and administration?

Evaluation Checklist

Here is an evaluation checklist that covers the basics to be considered, based on one that I used in various forms and stages of development over a number of years of AV/antimalware procurement, testing, installation, and maintenance. Essentially, it's the spine of the previous section, stripped of my personal prejudices and reduced to a list of issues. I'd suggest that you modify it extensively to suit your own requirements and environment.

Core Issues

- Configurability
- Cost
- Ease of use
- Functional range
- Performance
- Support functions

Cost

Unit cost/license terms:

- Per seat
- Per user
- Per workstation
- Per server
- Separate home licenses

Extras costs:

- Helpline support
- Extra media
- Extra documentation sets
- Product training
- Definitions update packages:
- Customizations

Administration costs:

- Installation
- Risk assessment
- Design
- Test

- Distribution
- Maintenance/upgrade/update
- Incident management

Performance

Detection rates:

- ItW
- zoo viruses
- New, high-profile threats
- Unknown threats (heuristic performance)
- Range of threats detected
  - System viruses (hardware/firmware/OS-specific)
  - Parasitic viruses
  - Macro and script viruses
  - Multipartite/multipolar threats
  - Mail-specific malware (mailers, mass mailers, and so on)
  - Web-hosted/borne malware
  - Network worms, worm/virus hybrids
  - Trojans (destructive, password stealers, backdoors, banking Trojans, and so on)
  - Bots
  - Latent viruses
  - Cross-platform viruses/heterogeneous transmission issues
  - Generators
  - Intendeds, corruptions, other non-viables
  - Jokes
  - Spyware
  - Adware
  - Possibly Unwanted Programs/Applications

Accuracy

- Overall susceptibility to false negatives
- Susceptibility to false positives
- Exact identification/generic signatures/heuristic capabilities

Scanning Targets

- ItW scanning
- Zoo viruses
- Malware, jokes, and so on
- Whole system
- Selected volumes
- Selected folders/directories
- Selected files
- Inclusion/exclusion lists
- Heuristics
- High heuristics
- Encrypted files
- Compressed files
- FAT32
- NTFS
- HPFS

Speed of Execution

- Speed of loading
- Quick scanning
- Deep scanning
- Heuristic analysis overhead
- Hybrid checksumming (scan if checksum mismatch)
- Performance overhead/latency

- Compatibility/contention (on-access scanning)
- Archive scanning
- Compression formats supported
- Impact of background scanning on performance
- Encrypted object handling (scan or alert)

Repair

- File viruses
- Macro viruses
- Boot sector replacement
- Clean boot repair
- Dirty boot repair
- Registry repair
- Repair verification issues

Compatibility

- With operating environment
- With utilities (esp. memory-resident)
- With applications
- With other security software

Functional Range

- Platforms protected
- Entry points covered
  - Desktop
  - Local Area Network (LAN) servers
  - Mail gateway
  - Proprietary mail servers
  - File Transfer Protocol (FTP)
  - Hypertext Transfer Protocol (HTTP)

  - Network News Transfer Protocol (NNTP)
  - Firewall plug-in
  - Secure file and data transfer protocols

Ease of Use

- User transparency
- Quality of/need for documentation
- Supplementary in-house documentation required?
- Ease of administration
- Event logging/analysis/reporting
- Tools/documentation for installation/migration
- Tools for deployment/distribution/maintenance/incident management
  - Centralized
  - Local

Configurability

- Sensible defaults
- Reconfigurability
- Scriptability
- Desktop lockdown
- Passworded re-configuration
  - Passworded components
  - Passworded de-installation

Integration with Other Defensive Components

- Integration across platforms and environments
- Detects across platforms
- Server version checks workstation version
- Server version updates workstation version
- Server functionality versus local (client) functionality
- Local functionality backs up server functionality

Testability

- EICAR support
- Error level support (batch files)
- Results logging
- Test facilities

Support

- Helpdesk support
- Quality of technical info
- Quality of product info
- Glitch management
- Acknowledgement of problems

Provision of Workarounds

- Access to 2nd-line support
- Speed of response
- Phone
- fax
- Pager
- E-mail

Program Upgrades/Patches

- Reinstallation documentation
- Stability
- Availability
- Speed of delivery
- Quality Assurance (QA) reliability and trustworthiness

Definitions Updates

- Frequency
- Beta/interim drivers
- Availability
- Push/pull
- Delivery mechanisms

Product Development

- Access to beta software and documentation
- Seminars
- Product information updates

Training/Education/Information Sharing

- General virus management education
- General security education
- Current malware/anti-malware technology
    - Independent speakers?
    - Sales versus research information
- AV training for engineers, administrators
- On-site user training offered
- E-mail alerts
    - Availability, frequency
    - Quality – useful or marketing
- Web site information
    - Quality
    - Currency

Reactivity to New Threats

- Receive/pull suspicious samples
- Documented submission processes

- Speed and accuracy of response
- Secure submission procedure

Outsourced Services

- Documentation compilation/distribution
- Incident management/reports/statistics
- Installation/distribution/maintenance

Documentation

- Virus information database
- Informational accuracy and value
- Range and number of threats listed
- Cross-reference to other vendors, CME and so on
- Uses Computer Anti-Virus Research Organization (CARO) nomenclature or related (WildList e.g.)
- Online Documentation
- Universal reading format (Portable Document Format [PDF], Hypertext Markup Language (HTML), Rich Text Format [RTF], ASCII)
- Paper documentation
- One manual for all platforms
- One manual to one platform
- Clarity and accuracy of writing
- Standard terminology?
- Explain basic concepts?
- Cross-reference to other sources
- Effective indexing?
- Contact info?
- Help with policy formulation - sample policies
- Sample login/update scripts, batch files, *.INI* files and so on

Testing Antimalware Products

Testing anti-malware scanner detection performance has long been a contentious issue, and only a very few testers and testing bodies are recognized as competent in this area by the anti-malware research community. (See the later section on "Sources of Independent tests and Certification Bodies" for more information). The advantage that such testers have over those with no links to the industry is this: they have generally earned the trust of the research community (though by no means universally!), by demonstrating that they test competently, safely, and ethically, while remaining independent. This trusted status means that they often have access to validated virus samples such as those collected, tested, and authenticated by the WildList Organization (see www.wildlist.org as well as the later discussion of In the Wild testing)

Tools & Traps

Researcher Ethics

The antimalware industry tends to regard most other tests with a high degree of suspicion, and often consider them invalid or otherwise inappropriate because:

- The competence of the tester cannot be assumed, and therefore neither can
  - The appropriateness of the methodology (or it's correct application)
  - Adherence to safe and ethical practices in handling and testing samples
  - Adherence to industry ethical codes and standards can't be assumed

Since the early days of the anti-malware industry, the members of the research community have (in most cases) adhered to a code of ethics that has forbidden the exchange of samples with outside bodies. In fact, even within the industry, the sharing of samples has been based on personal knowledge and trust of the receiving researcher. There are several very good reasons why this is so, the first of which was an intention to limit the possibility of the samples spreading further (particularly in the days before the Internet made it possible for replicating to spread around the world in hours). There have also been persistent (and entirely unfounded) rumors that the AV industry itself was responsible for creating the viruses in a cynical attempt to raise sales. Encouraging a high standard of ethical behavior, and expecting the same from all other members of the community, ensured that such a rumor would remain the unfounded nonsense that it ever was.

To be scientifically valid and useful, a test must be repeatable, independently verifiable, and have a sound methodology. The samples in a test set must be validated, classified by group and type, and maintained in good order.

The purpose of testing is not to "trick" the anti-malware program (I shall use the broader term here, as I shall talk about more than just viruses) into showing its weaknesses, but rather showing its capabilities in detecting malware. To do this effectively, it is important to be sure that the samples used to test against are really malware. If they are not, the test cannot be useful for showing the capabilities of the product, nor for helping the user in making informed decisions in evaluating those capabilities. I shall refer to the process of checking samples to ensure that when creating a test set it contains only working malware samples as "verification."

Unfortunately, a huge amount of "junk" (including damaged or corrupted files, non-executable files, intended viruses, and completely innocuous files) exists in many malware collections. Sadly, some of it also exists in the collections of anti-malware companies, but because of the prohibitions against sharing malware samples outside of the anti-malware research community, the majority of junk exists in so called virus-exchange (vx) collections downloaded from the Internet.

Notoriously, testers who are unable to tap into AV community sample pools try to substitute samples pulled off virus exchange Web sites and other dubious resources, which may contain all sorts of non-viral samples and other unverified miscellany.

Curiously enough, difficulties in obtaining validated samples have contributed to (but not caused) a situation where testers, concerned about the effectiveness of a certain scanner against unknown viruses, were testing heuristics when the concept of heuristic barely existed in anti-virus technology. Even before the technology acquired the heuristic label and its 21st century capabilities, they were generating new, unknown variants of known malware. Unfortunately, this typically involved the use of unreliable virus generators, irrelevant virus simulators, random placement of code in other files, or masking of virus code and text strings, and so on. The possibility that any of the samples were non-viral effectively invalidates tests of anti-malware scanners (if the samples were assumed to be viral in the first place). In such a case, the highest detection rate is not necessarily the best, since it may include high volumes of false positives (even supposing that all scanners tested were configured consistently).

Often these creations are "validated" by using anti-malware programs to scan the samples, and if any product detects a sample, it is assumed to be "good."

This method is highly flawed. If a piece of junk is detected by one scanner, it may replicate itself through such badly verified collections (by being shared around) even though it is not a valid (truly replicative) sample. Such samples can end up being detected by anti-malware products, simply because they are known to exist in the sample collections of certain testers.

This poor methodology for test set building simply reinforces the cycle and adds increasing amounts of junk to the detection databases of anti-malware collections. It does not help anyone (unless you're in marketing and want to play the numbers game of "we detect X more viruses than product Y"), and it gives a false impression to the public as to what an anti-malware scanner is actually supposed to do. Detecting malicious software is one thing, but spending your precious processor cycles detecting large amounts of junk simply because some testers don't do their job properly, is a waste of time.

This is fraught with danger, unreliable at best, and downright misleading at worst. In the end, it becomes a vicious cycle of tester incompetence compounded when antimalware companies add more junk detections to improve their scores in tests.

Building a sample collection in such a way is a sure way for a tester to lose credibility and respect in the industry, and seriously lessens the chances of your ever being trusted by the industry. Undeterred, many simply continue making useless tests, misleading the public, and wasting the time of the anti-malware research community.

This section is intended to be a guide to good testing practice. The space constraints mean that it neither discusses the methodology in great depth, nor in a completely comprehensive fashion. It will, however, discuss the verification of various types of malware, look at common types of tests, examine some commonly encountered problems in testing, and give an overview of the various accepted testing bodies and the types of testing offered. Ultimately, the aim is to show that no single test or certification should be taken as indisputable proof of an anti-malware product's superiority or inferiority in comparison with other such products. A range of results, and most importantly, a consistent record across a range of tests is a far better indicator of usefulness than a focus on a single test or type of test.

Replicating Malware

Replicating malware has one defining characteristic; that it replicates! (No, honestly...) That is, it is able under specific circumstances to create a (possibly evolved) copy of itself. If it does not, or cannot do so, it is not a virus. The term virus has unfortunately become a catch-all term for all types of malware, but in anti-malware research circles, that designation is only ever used to refer to replicating malware. Worms are usually considered to form a subset of viruses, with the distinction that worms tend to be self-contained. That is, they do not need to infect a host file to replicate, but their own files (or code) contain everything needed for replication. Traditional or "true" viruses usually infect host code (often a legitimate file or system area) by inserting their code (or a possibly evolved copy of their code) into it, in such a way that it is executed when the original code is run, thus enabling further replication.

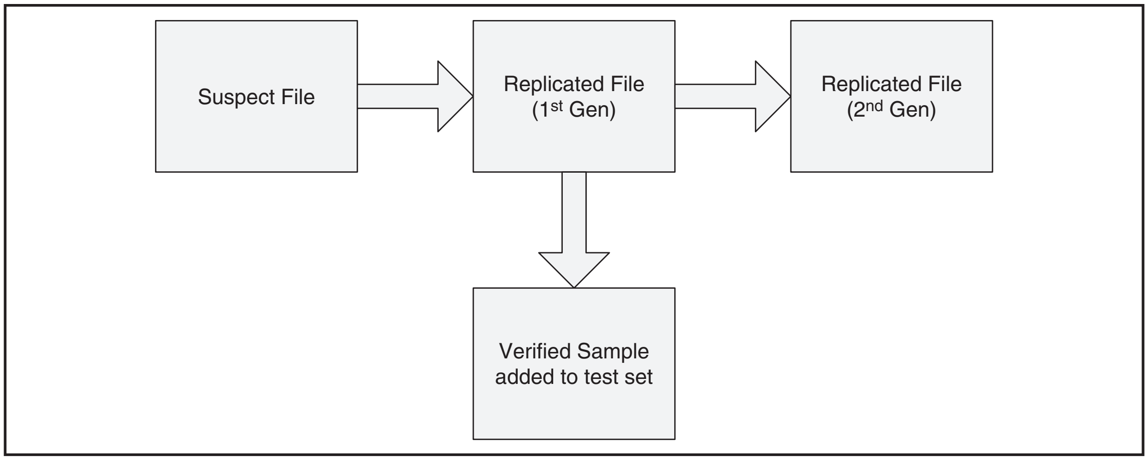
Why is Sample Verification Important?

Viruses are notoriously buggy; after all, virus writers typically don't do much quality analysis. This means that even if one sample is working, it can produce many "offspring," which don't replicate (i.e., are neutered), and sometimes won't even execute. Virus generation kits are a classic example of the hit-and-miss generation of replicating malware. Unfortunately, testers who have no access to verified samples (or don't know how to verify samples) have often used such kits to generate new samples for testing. Quite apart from the ethical undesirability of creating new viruses, existing virus generation kits are known to produce more non-working samples than working. Without subsequent verification of viability (i.e., ability to replicate further), such samples used in tests are meaningless.

That being said, it is relatively simple to verify viruses and worms, as long as you can reproduce the conditions necessary for them to replicate (that is, in a number of cases, the exact condition of the author's computer). It simply requires that each sample be replicated to at least a second generation.

In practical terms, this means executing the virus in an environment where it is able to reproduce itself. In normal circumstances, a working copy of the virus will be produced. Then, a second replication of that replicated object is performed, to ensure that the replicated sample is also valid. For the purposes of testing, the "middle" validated virus should be added to the test set as a confirmed sample, as it is both replicated, and able to replicate (see Figure 10.1). This methodology is valid for "static" viruses and worms (i.e., ones in which the code is identical between generations of replicants).

Figure 10.1 Replication of a Valid Static Sample (Non-polymorphic)


Suspect File
Replicated File (1st Gen)
Replicated File (2nd Gen)
Verified Sample added to test set

In the anti-malware industry, replication is the *condicio sine qua non* of testing.

The purpose of the second generation replication is to ensure that the sample generated is also valid as many viruses produce non-working copies of themselves (for instance the Win32/Magistr.B virus).

Why, then, should we not just use the suspect file once we are sure it replicates? Usually a virus will have infected a host file, and this host file may contain un-needed or undesirable other information. Viruses are usually replicated into so called "goat" files, which are special empty (clean) files, which are able to execute, but have no function. Replicating into a goat file ensures you have a "clean" virus. There are also times when suspect files can contain more than one virus. For example, there are many instances of worms (which are, remember, self-contained rather than needing to infect a host file), which themselves have been infected by a virus. (For instance, I have seen several Win32/Netsky worm files infected with the Win32/Funlove virus). It is important that each file contains only a single threat. In such cases as double infections, the worm may replicate as normal, and the virus within it should also infect the goat file. In this way you can extract the virus, but you can't use the worm sample, because that is likely to be infected with the virus, along with its variants. This is a problem because it can cause problems in testing, if a scanner sees one malicious program before the other. (This isn't necessarily because it can't detect both, but because as soon as it sees a threat in a file, it doesn't need to scan that file anymore, as it knows it is dirty).

Polymorphic Replicative Malware

Polymorphic or metamorphic viruses, rather than replicating a static form of their code, will produce variations on their code each time they replicate. In this way they change their code at each replication, usually through some form of encryption.

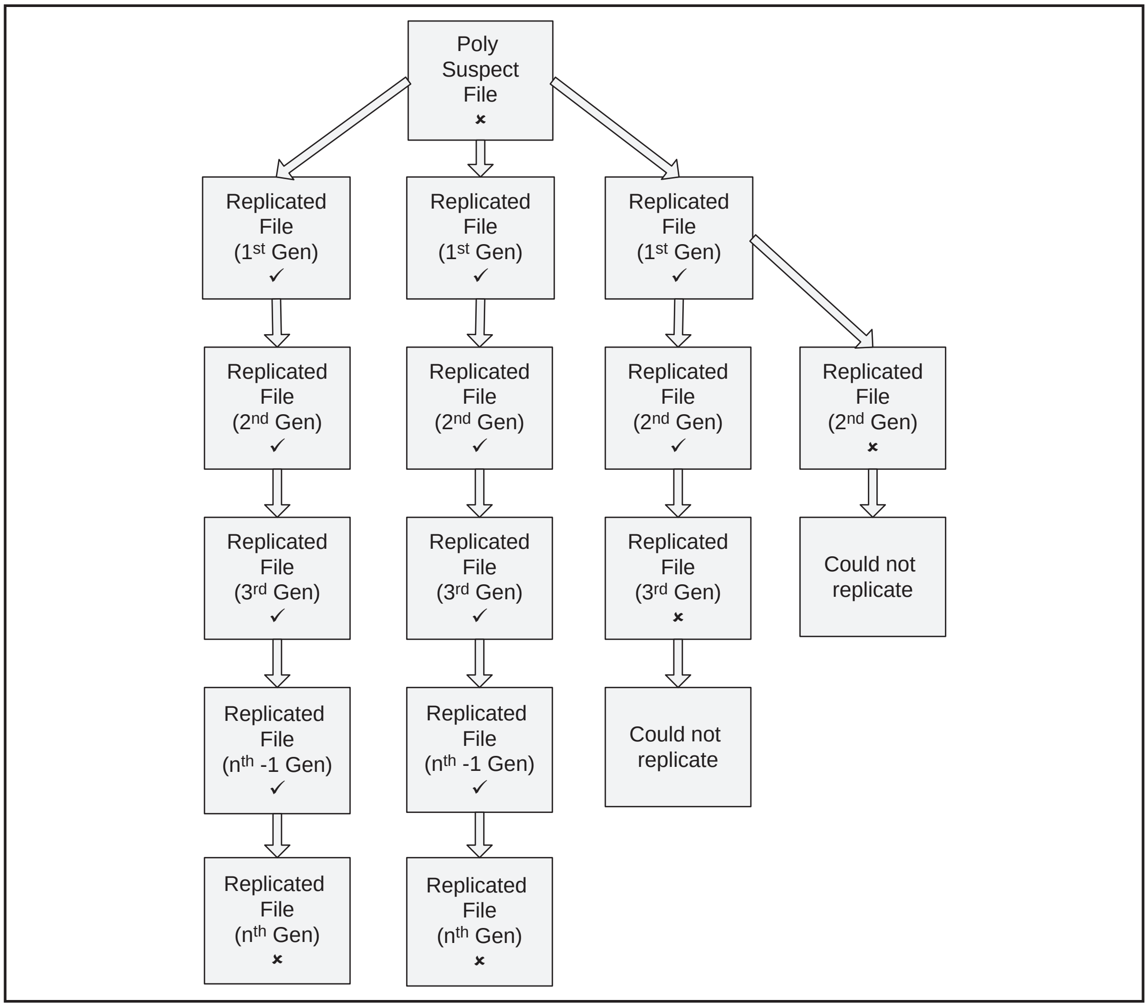
NOTE

By encryption we mean an obfuscation technique designed to avoid detection by simple static signature scanning (for instance, simply hashing the virus code could result in failure to detect more than a single sample).

A good test of detection capabilities with polymorphic viruses contains more than one (and often several hundreds or thousands) of replicated samples. The reason for this is that proving that one sample is detected doesn't say anything about a scanner's overall detection capabilities regarding that virus. Some polymorphic viruses are very complex, and can potentially produce billions of variants. To detect one sample (or maybe even a few) may be possible using a simple signature, but to detect all variants it is often necessary to create a specific algorithm. To test that this has been done effectively, several thousands of replicated samples of the same virus are often needed.

In this case, the form of replication would be from a suspect file, replicated to one or more first generation files, which then are subsequently replicated *n* times, and the sample set contains everything from the second generation to the *n-1th* generation. Any sample that does not subsequently replicate in this process is discarded, and another is generated from the "parent" sample.

Figure 10.2 Replication of Polymorphic Samples, Showing Inclusions and Discards


Poly Suspect File ✗
Replicated File (1st Gen) ✓
Replicated File (1st Gen) ✓
Replicated File (1st Gen) ✓
Replicated File (2nd Gen) ✓
Replicated File (2nd Gen) ✓
Replicated File (2nd Gen) ✓
Replicated File (2nd Gen) ✗
Replicated File (3rd Gen) ✓
Replicated File (3rd Gen) ✓
Replicated File (3rd Gen) ✗
Could not replicate
Replicated File (nth -1 Gen) ✓
Replicated File (nth -1 Gen) ✓
Could not replicate
Replicated File (nth Gen) ✗
Replicated File (nth Gen) ✗

It is important to stress that every file in the set chosen must not only be a replicated sample, but must also able to replicate. This is why in Fig. 10.2, the *nth* generation is discarded, and only the generation *n-1* is retained (where it is able to replicate). All samples that are not able to replicate are discarded. Not only that, but it is good practice to test that samples which produce broken replicants are also able to produce viable replicants (see the third sample in Fig 10.2). Otherwise, although it may produce a single generation, if it is not possible to replicate that further sample correctly, it must in some way be damaged. There are cases where a virus may not replicate beyond a single generation, but these cases are rare.

Environment

Viruses tend to be specific to particular operating systems (there are a very few cases of viruses which can replicate on more than one OS; e.g., Windows and Linux), but within a particular family of operating systems, there may also be variation in ability to replicate. It is something of a misnomer simply to refer to "the Windows ™ Operating System", as there are actually many Windows operating systems, and they are very different. The earliest windowing systems, up to the 9× (including Windows ME) flavors, are built on an old technology called DOS (strictly speaking, MS-DOS and PC-DOS, the IBM variant). The other major branch was built on a "New Technology" (NT): its descendants include Windows 2000, Windows XP, Windows 2003, and now Vista). In terms of replication, viruses tend to be specific to the OS series they were created for. So a virus may replicate under Windows 98, yet not be able to do so under Windows XP. This can lead to a problem where a very specific set of components need to be available for a virus to replicate, which can seriously complicate the replication process. (For example, a version of Windows 98 OSR2 running a Chinese version of Microsoft Office and Internet Explorer version 5.) Although such a combination may be rare, it is common enough that a competent tester will need to have access to such esoteric setups. Often the best way (sometimes the only way) to determine the actual function of a virus is to perform a disassembly to discover exactly what the code is doing (or is intended to do). Although tools for automated disassembly (e.g. IDA Pro – see www.datarescue.com and OllyDebug – see www.ollydbg.de/) are available, a knowledge of assembler and programming is very helpful in disassembly.

Unfortunately, this is another barrier to competent testing, as it is time-consuming and often complex, and many would-be testers simply don't bother and opt for the "easy" route of relying on scanners for "validation" (actually, pseudo-validation.)

In the Wild Testing

Now is a good time to introduce the WildList (WL): more formally, the WildList is a list (really!) of viruses that are counted as existing "In the Wild" (ItW). There are around 90 reporters, (not all active, but all highly competent researchers) who prepare monthly reports of the viruses of which they have confirmed "sightings." These reports are compiled by the administrators of the WL into a full report in two parts. The first part (sometimes called the "top list") consists of viruses that have been reported by two or more of the reporters. These are counted as being really ItW. The second list (bottom list) consists of viruses that are currently reported only by a single reporter. The distinction is to ensure that the viruses reported are really spreading and are not just isolated incidents. It is worthy of note that the WL only contains replicating malware (hence my use of the term viruses), and does not (currently) list any other types of malware (unless they coincidentally contain or are contained by another, replicating malicious program). This list, while not comprehensive in terms of all malware, is a good indication of the state of real-world replicating malware.

It has therefore become the *de facto* standard for conducting testing of AV (specifically) products. It forms a baseline standard of what AV products should be able to detect, and is the basis of the test set used by respected testing bodies such as Virus Bulletin. The WildList also forms the distinction between "ItW" viruses and "zoo" viruses used in testing.

A zoo sample, (as opposed to wild – clever eh?) is one that is part of a collection of (hopefully verified) viruses, which are not on the WL. Zoo samples usually have either been on the WildList but are no longer so, or else they exist only in such test sets (i.e., they have never been publicly released). Surprising though it may be, in the past, virus writers were not always bent on wanton destruction of the world's computer population (or in the creation of malware such as bots for the pursuit of phishing, spamming, and so on). Many created their viruses because they were interested in the techniques (yes, they could have found better things to do) and it was some sort of hobby. This sort of virus writer would often simply send their creation to AV companies to be added to their detection databases. Such viruses never saw the light of day outside of the laboratory.

Other zoo samples can consist of bottom list WL samples, and even older samples, which are only kept for historical reasons (for instance old DOS viruses). Zoo sets typically contain more "junky" types of files, including intended viruses.

NOTE

Intended viruses (or "intendeds") are attempts to write replicative malware that never really worked (i.e., replicated). However, they may be interesting because of a particular technique they used, perhaps one that might get subsequently added to another virus. Also, where the intended is one stage of a work in progress, clever detection of an intended may detect the virus writer's next *working* replicant.

Detection of WL viruses (that is, viruses which have been replicated and verified by the testing organization) is considered to be the minimum standard that AV programs should achieve on a consistent basis. As useful as the WildList undoubtedly is, its limitations should be noted:

- First, it is by no means comprehensive or current (typically the WildList is published three months after the data is compiled).
- Second, it is only relevant when talking about replicating malware. As the threatscape has shifted in the last few years, and there are now far more threats which are non-replicating, this is an important consideration. While there are ongoing discussions about including such non-replicating objects into either the main WL or a supplementary list, the current version does not contain these threats.

However, the anti-malware products available are able to deal with such threats, out of necessity, as they form the majority of the malicious programs currently seen. Non-replicating malware has its own unique set of problems in verification and testing, and I am going to discuss those in the next section.

Non-Replicating Malware

Non-replicating malware is, as you may already have guessed, malware that does not replicate, and is therefore neither a virus nor a worm (nor any of the more esoteric categories of replicating malware).

Traditionally, this third major malware category (or second, if your categorization of worms is lumped in with viruses) has been referred to by the catch-all term "Trojan Horse." I will continue this tradition, as the term can be wide enough to encompass all of the modern non-replicating malware that we need to discuss here, including bots, spyware, rootkits and so on. For brevity, I will simply use the term "Trojan".

First I will need to lay out some definitions (and definitional problems), because this is a somewhat complex area, and one that directly impacts on the effectiveness of testing products against non-replicative malware. I'll briefly discuss some of the problems, and propose a way to deal with some objects that might be hard to classify. Unfortunately, while "Trojan" is very useful as a catch all term, the fact is that it is a very wide reaching term, which encapsulates just about every other form of malware other than replicative worms or viruses. (Actually, there are arguments for including some or all replicative malware under the same umbrella, but we won't even go there...) this very fact also makes it awkward to tie down what the term actually means. Indeed, this ambiguity equally extends into some of the subcategories such as "spyware."

Is It or Isn't It?

I believe I know what I mean when I talk about a "Trojan Horse" (Trojan or trojan: capitalization is disputed); however, the complexity of defining what constitutes a Trojan means that it is less than likely that my definitions will coincide with yours. In fact, even other people in the AV industry will probably disagree with some of the definitions or explanations I could give. However, descriptions are important, so I'll try to give one, but I'll also try to demonstrate a sound way of deciding (even if it's similarly subjective.)

A loose description might be any programmatic code that, when executed or installed onto a system, performs an unexpected action, or which describes its function deceptively. Such code is usually executable, but not necessarily so. For instance, a zero-byte file that is not executable, or a corrupted executable which is no longer executable, may in some circumstances or on some platforms (e.g., Symbian OS) cause unexpected behavior when executed (or installed onto the system).

NOTE

David Harley usually uses the definition "...a program that claims to perform some desirable or necessary function, and might even do so, but also performs some function or functions that the individual who runs the program would not expect and would not want." ("Maximum Security" 4th Edition, Chapter 18.) We're not sure of the exact provenance of this definition, but it's been pretty useful over the years.

We immediately run into a problem here. It is quite rare for even well-documented programs to describe themselves adequately, reveal all their features, or to escape performing unexpected actions at times. In the first instance, description of function, it is possible to argue that lack of or incorrect documentation is the problem. In the second instance, performance of unexpected action, we have a situation that is more normally referred to as a "bug" (that is a piece of code which contains a (non deliberate) error that causes unexpected behavior).

In the case of replicative malware (viruses and worms), there are clear boundaries between what is or is not replicative. If it replicates, it counts, if it doesn't, it doesn't. Of course, the fact that something doesn't replicate doesn't mean it's not malicious, and so we have our third (and nowadays by far the largest category) of malware. That is, the code in question is considered to take some action detrimental to the interests of the user, and that could be considered to be in some way damaging. I should note here that damage does not have to be physical (e.g., deletion of files), though it may be, but could also be financial, social (embarrassment), or simply using up computing resources unnecessarily.

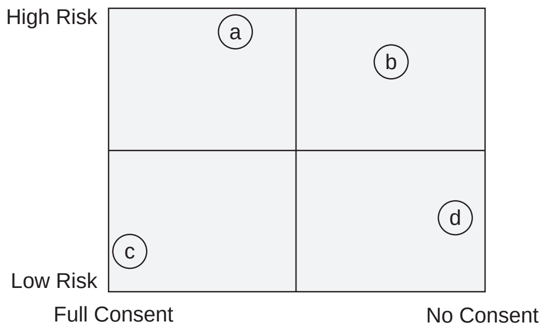
Because of the lack of a strict formal definition (unlike Fred Cohen's famous mathematical description of a virus – see "A Short Course on Computer Viruses", published by Wiley), we need to somehow be able to categorize a program, with a reasonable degree of certainty as being malware (or not). While to a trained security professional "I'll know it when I see it" may often be good enough, it doesn't help us in our quest to find out what constitutes non-replicative malware.

A helpful starting place is to examine objects based on some matrices formed on categories such as:

- Malice
- Risk
- Intent
- Consent
- Disclosure

We may need several of these for a single object, but a simple example is below in Figure 10.3, based on a single matrix, mapping consent against risk.

Figure 10.3 Consent/Risk Matrix



Before I discuss the positioning, I'll explain what I mean by the terms. A Low Risk item might be something that occasionally submits anonymous statistics back to the producer. A High Risk item might send keystroke logs or visited browser Uniform Resource Locators (URLs) back to a server. Consent is fairly obvious (though often confusingly obtained). A clearly laid out message such as "Would you like to send us anonymous statistics" with an optional check box, has a high degree of consent, as opposed to:

- A couple of lines hidden on page seven of a ten-page End User License Agreement (EULA)
- Simply no warning at all
- No option to disable the function, which would be low or no consent

The Risk factors might be anything, such as opening a port, or deleting files and so on. The matrix needs to fit the action of the object, and in some cases consent may not be an issue, but intent may be.

So, if I look at object "a" and it gives me a degree of consent (let's say it has a reasonably clear EULA), then I would mark the Consent Axis towards the left hand side (how far to the right or left depends on the degree of consent). So, that takes care of consent, but there's a high risk associated. Let's say a port will be opened up and a folder shared with the world. Many peer-to-peer (P2P) applications do this. This might not really be a malicious program, particularly if it tells me what it's going to do, but some AV programs at customer request may detect it. Sometimes, it might be detected in a category like "Riskware" or "Potentially Unwanted Application." However, the high degree of risk means I place it in the upper quadrant on the "Risk" axis.

Object "b" is more definitely undesirable. Let's say I've looked at it, and it has a pretty low degree of consent, perhaps something obscure buried in the EULA that the company hopes is good enough as a legal "get out" clause, but it's not really clear (or easy to see). I've decided it also has a fairly high degree of risk; perhaps it's sharing my browsing habits back to the producer, who is examining all the sites I visit, so that they can target me for advertising. Or perhaps it installed an Ad server. (This falls into the Trojan-related category of Adware/Spyware.) In this case, having looked at it and placed it on my matrix, I can see it's something I would call malware.

So I can place objects on the matrix, and decide whether they're doing something that I think is malicious or not. So, "c" would not necessarily be malicious, but "d" might be.

Of course, this sort of analysis may not be enough, nor is it a particularly objective standard. It may be more useful in certain cases than in others. For instance, it's a good way to decide if something exhibits the behaviors of what might be called spyware (a categorization of Trojans with some sort of "call home" function). The program may have an option to give full consent, and be 'low risk' in some circumstances, but I may discover that in some cases, the intent is malicious.

A famous Trojan called BackOrifice claimed to be a legitimate Remote Administration Tool, similar in function to Microsoft's Software Management Server (SMS), and indeed it could be used in that way. But more often it was (and is) used to control someone else's computers without their consent or knowledge. In some cases, companies producing such "utilities" have threatened, or bought, litigation against AV program vendors for detecting their programs. This has become increasingly common with spyware and adware makers.

This process is time-consuming, and in some cases, I wouldn't need to use it. Sometimes it would be obvious: some things, for example an executable delivered in e-mail with a phishing message, are self-evidently maliciously intended. It's not hard to tell that it's going to be malicious. (Of course, I still need to verify that it works!). It also applies when examining more traditional types of Trojan, for instance a file purporting to be a game, but which deletes files instead, or formats the hard drive.

To add to the complications, some Trojans will only work as part of a more complex collection of files. In simplest form, this may simply mean that the Trojan, when run, downloads another file and runs that. Some will drop a file, which is then executed, and that file then downloads another file; and so on. This behavior can be interlinked in very complex ways, and is one of the ways in which malware authors try to avoid re-doing a lot of work by making it harder to detect all of the parts of the malware at the same time. Some spyware objects have been known to download several hundreds of other objects, including viruses and rootkits.

Well, although I've not really provided a firmer definition (quite deliberately) of what a Trojan is, I hope that the discussion has served to increase awareness of the problems in diagnosis and categorization, and showed that in the gray boundary conditions, a common sense approach (combined with some basic tools) can help. With that, I'll move on to an even more troublesome topic. What types of non-replicative malware (which we'll loosely refer to as "Trojans") should be included in test sets against which anti-malware programs are tested?

Two main criteria can be applied here:

- Is it actually a Trojan?
- Does it work?

I've already loosely covered the topic of `Is it actually a Trojan' above, and shown that a very broad range of objects could be considered Trojans. So I'll move on to the next most important criterion.

Does it work?

While it may be simple to determine, in many cases, that something *is* a Trojan, and it's easy to tell if it is intended for malicious purposes, it can be very hard to determine whether it actually works. Not only that, but there are also cases where legitimate utilities can be misused, and so the same object is both a Trojan and not a Trojan. (To deal with this, some anti-malware companies will create special categories of "potentially unwanted programs," "potentially unwanted applications," or a similar term). Some examples of hard-to-pin-down objects include "logic bombs," which simply wait an unspecified amount of time, or for certain conditions and then pounce, usually overwriting or deleting files. As with replicating malware, a Trojan may be incompatible with the system under test: for instance, under Win9x it may be horribly malicious, while under WinXP it does nothing, and vice-versa. Such platform and function specificity can make analysis of such objects very tricky, and sometimes only a disassembly will be adequate. However, it is not only necessary to determine whether something *is* a Trojan, but also whether it works as intended (or indeed, even partially as intended). As we have mentioned earlier, the purpose of a test isn't to show how many objects a scanner can detect, but how well it detects actual malware.

Full disassembly and examination of hundreds of samples is prohibitively inefficient, so other means, even automated, can be used to determine behavior (e.g., changes to a system when the suspect file is run). Adding registry keys to sensitive areas, opening ports, installing services, and deleting files can all help to indicate suspect behavior, but it is easy to find legitimate programs that do the same things, so caution is needed. Ultimately, it is the responsibility of a tester to maintain his collection in good order, and to validate the samples as they come in. Often, there is a discrepancy between what the anti-malware vendor expects of their product and the result of a particular test. Missed files may be requested from the tester (and in most cases such a request will be granted), and further examination takes place. This process will often weed out junk from a tester's collection, and also improves the detection rates of the product. Far from cheating, this actually supports the purpose of anti-malware testing (i.e., improvement in detection of and customer protection against malware). After all, what point is testing if it doesn't improve the situation? Some amateur testers would do well to think on that.

As there is no current WL for Trojan-type objects (at the time of writing, there are discussions going on around the creation of such a list), all tests against Trojans are by definition "zoo" tests. The major commercial certification bodies have built their own collections of objects (and subcategories such as spyware/adware), and they provide certification for products measured on detections against those collections. Unfortunately, controversy abounds, particularly with respect to spyware and adware, as anti-malware companies are increasingly subject to (frequently spurious) legal claims against them for impeding legitimate business. The Anti-Spyware Coalition (ASC – see http://www.antispywarecoalition.org/) was, in part, set up to address this problem. In some cases, purveyors of previously "bad" adware/spyware have cleaned up their act, and now tailor their software to give the consumer more privacy and choice about shared information. Others have briefly cleaned up their acts, lodged complaints that they are being mistreated by ASC members who detect their now 'legitimate' software, forced removal of the detection of their products from the databases of anti-malware products, and then reverted to their old habits. Still others simply attempt to use the legal system to justify their bad practice, and thereby force the removal of detection for their products.

Another significant problem facing the anti-malware industry of late is that of distribution. Many forms of malware are now automatically generated, spammed out through vast robot networks (botnets) of compromised machines, in less than a few minutes. Then the cycle repeats. Not only that, but once such malware is installed, it will invariably update itself to newer code, so that it can attempt to remain invisible to anti-malware products. This means that there may be thousands of variants of a particular item of malware, tweaked sufficiently to avoid detection, and constantly updated and re-released. Often, the malware may be limited by region; for instance, if you have an infected machine in Italy, it may download an update specific to that region, whereas the same Trojan in Germany would download a different update.

This produces a complex problem for testers:

- First, in obtaining the objects (if there is one version of a Trojan per download which one(s) do you test with?)
- Second, in deciding whether this is a genuine threat that should legitimately be handled by anti-malware products (again, if a Trojan is only ever seen in one location at one time, is it a serious threat?).

It is also a problem in terms of verification. Let's say I obtain a new piece of malware, and I discover that it downloads a few other things, and that my product detects all of those things. I dutifully share that sample and those objects with other anti-malware researchers, but when they test the same sample it downloads an entirely different set of objects. This can and does happen, and is a problem in building test sets. When we had a virus, we knew that it would replicate, and it would have a life for a while if it existed in the wild. With this new sort of object, we don't know if it will ever be seen by anyone else, and certainly by tomorrow, most of the things we detected today will have changed.

One solution to this has been to suggest a more statistical model of testing, where what counts is not an exact identification of a specific set of objects, but ongoing detection performance (in real time, or as close to that as possible) against incoming objects that are discovered by the testing laboratory. Effectively, an incoming stream of malware (usually from sources such as honeypots) is cached and fed back to malware scanners, and the results recorded. Unfortunately, as with viruses, many such samples are just junk (not working or corrupted in some way), but it is a reasonably effective method, particularly if some validation is carried out.

The phenomenon of rapid outbreaks is not new, however. It has long been a factor when dealing with worms, some of which have spread incredibly rapidly. This is particularly true in cases where a vulnerability has been exploited (CodeRed, SQL/Slammer, Sasser, MSBlaster); some of these worms have spread like wildfire. This led to the idea of "Time to Update" testing, which purports to measure the time it takes for an anti-malware company to release a detection update for a given threat. A discussion of this type of testing, and its problems will follow. Given that a Trojan may have a "life" of only a few hours, this may be considered a useful measurement; however, the opposite is true. The only measure of success against such malware is to be able to proactively block it, and, to this end, almost all anti-malware companies attempt to create "generic" signatures or use some form of sandboxing to block such new malware in real time, rather than having to release updates. Sadly, in some cases, it is only necessary to actually release an update because a file may exist in the collection of a tester.

Time To Update Testing

Time to Update (TtU) testing is, supposedly, the measurement of the time that it takes between the release of a new piece of malware, and the release of the detection signature that an anti-malware product requires for detecting that malware. It has been referred to as testing the "window of vulnerability," but is actually a measure of the time taken to update a given product after a (rather arbitrarily decided) point in time. In this type of test, a product that detects the malware proactively (i.e., via some means not requiring update using specific detection – this non-specific approach is sometimes called *heuristics*) is given a score of 0, and this is used as the baseline for measuring the other products against in terms of time.

The period of time after the set baseline point (assumed to be the point of first detection) is measured, and as the products update, they are assigned a time according to that baseline. This is done by frequently checking whether an update is available, and/or retrospectively checking those updates against the new malware.

Defining the Problems

At first glance, this all seems very reasonable, however, there are significant problems

- **Problem 1** TtU has been presented by some AV vendors and indeed some testers as a measure of protection against new malware. However, the reality is that it may in fact mislead the consumer into a false sense of security. This is because it doesn't really matter how long it takes you to get detection if you're already infected (yes, it's useful to know after the event, but better to be protected before).

- **Problem 2** When malware is detected heuristically/proactively by one product, and not by another, it may seem logical to set the baseline for time of detection at 0 for the proactive product. Any product that must be updated, is then measured from that baseline against the amount of time it takes to update that product, usually measurable in hours. However, with this approach the baseline is skewed, in some cases by several months. For instance, it is quite possible that specific malware could have been detected even as much as a year before its actual release by some products, a fact demonstrated by the "frozen update" tests discussed later.
- **Problem 3** The actual release time of malware may in some cases only be determined by the time of first detection. As first detection is based either on locality or ad-hoc networks of researchers, this skews the baseline, and puts it in favor of the lab lucky enough to identify the first sample. The basis of sound scientific methodology is identifying (and removing if possible) sources of bias.

I'm going to discuss these problems in more detail in the following pages.

Problem 1: Time to Update as a Measure of Protection Capability

The traditional model of AV protection has been one of reactive update. That is, malware is released, subsequently captured, analyzed, and a "signature" released that will accurately identify this malware. This model has held since the very earliest incarnation of AV scanners, where the maxim "you can't detect what you don't know about" has long held true, or been held to be true. In most cases, replicative code (apart from the obvious network worms) would spread slowly. (Non-replicative code has always needed external vectors, such as piggy-backing legitimate code, for instance on a download site, or attached to spam). Certainly the spread would be slowly enough that a large enough window of time was available to provide detection. However, there was always the "ground zero" problem. In a system infected by an undetected replicating program—that is, infected either before identifying updates were available, or where the system was not updated with available signatures — there was no remedy, nor prevention for such infection. Gradually, there has been a convergence in time of release (of new malware code), and "ground zero," and increasingly large populations have been and are affected by such attacks (this is particularly true of the big-hitting worms and mass spamming of new Trojan variants). The system of update release for some time lagged significantly behind, not simply because it takes time to analyze and identify the code (without causing false positives), but also because it is always reactive. There is almost always a ground zero. From this point of view, it does not matter that the infection occurred one hour, or 15 hours before update was available. The infection (or infestation) occurred in the first place, if a purely reactive system was in place.

If I am lucky, and hope that ground zero will always be "someone else's problem" and I assume that I will always be able to update before infection, I then need to consider the window of vulnerability.

In a game of Russian roulette played with a six-chambered rotating magazine pistol, the probability of a "hit" is determined by the position of the bullet in the magazine relative to the position of the firing chamber. The likelihood of a hit increases by an order of magnitude each time there is a "miss." The only point at which you can absolutely determine that there is a bullet loaded into the chamber is after five misses, at which point the probability of a hit is exactly 1/1 (assuming the gun is not gimmicked in some way). This point is rather a cogent one. Over time, the probability of infection before update is available converges, in the case of a global outbreak, towards one.

In a time-to-update test performed some time ago, only two companies averaged less than four hours to release updates for the Win32/Bagle.AS worm. In the case of Win32/Bagle.AS, a single Internet Service Provider (ISP) showed 3+ percent of messages carrying this worm in the first hour of outbreak (identified heuristically by the ISP's systems). In the second hour, 7+ percent of messages carried it, and by the third hour, over 32 percent of messages carried the worm. The probability of receiving an infected message is better than one in three an hour before the average time to update of the best performing companies. The simple truth is that even those who could release the update in the shortest time still did not provide adequate protection against the problem in these circumstances. The fact that such exponential growth of replicating infections occurs time and again, is testament to the fact that the model is flawed.

Why is this important? The main reason is that this test sends a false message to the consumer. It tells them that it's better if you can update faster, without giving them the full facts of the situation. If it takes four hours until an update is available, then we must also factor in download and deployment time, which may significantly increase the window of vulnerability. In a network situation, it is impossible to guarantee that update has occurred on all machines immediately an updated is released.

The message that quicker updates offer better protection is not generally true, and only holds true at all in a very limited set of circumstances. What it really measures is "how quickly will I know that something bad happened?" Taking into account the modern malware that has a very short lifespan, four hours would be a disaster!

Problem 2: Baseline Setting for Heuristic/Proactive Detections

The problem of heuristic detections is subtler, but directly relates to the first problem. Setting the baseline to 0 hour in the case that the heuristic detects a new replicative malicious program is fundamentally wrong, not only because it supports the already discussed idea that time-to-update is a valid measurement of protection from infection, but also because it fails to take into account the *actual* time that detection would have been available.

Consider the imaginary case of the Wiggly (please, don't anyone ever use this name for a real worm!) family of worms. Let us say that Product X1 does not detect variants of Wiggly without update in any scenario. Product X2 detects 50 percent of the variants without update (proactively), and 50 percent require updates. Product X3 proactively detected 100 percent of the variants.

Let us be generous to Products X1 and X2 and say that they always released an update (if necessary) within one hour of the release of the worm (of course, we should note that the time of release is not necessarily determinable by any other metric than of "first detection," but that is our third problem).

In every case, baseline for X3 is set to 0, in 50 percent of cases X2 is 0 and in 50 percent X2 is 1; X3 is always 1. What is the problem? Simply this; if we expand this capability over time, the pattern resets, where X2 and X3 are arguably offering better protection.

Imagine a scenario where on consecutive days there are new releases of a single virus. I'll update all the products at midnight day one and check that all three products detect all currently known variants, and call that T0. For the purposes of testing, no product will be updated unless it does not detect one of the new viruses. On day 2 at midnight (T1) (this is a well behaved example!) a new virus (V1) is discovered. X1 doesn't detect it initially, but updates in one hour; products X2 and X3 do not require updates. In the original time-to-update test, we would set the baseline at 0 for X2 and X3, giving X1 a time to update of +1 hour; but in reality, product X1 is actually +25 hours late (T0 +24h +1h for update of X1 = T1′), as the update of X2 and X3 was already effective at T0.

Now, on day 3 at midnight (T2), a second variant (V2) is discovered. Once again, X1 doesn't detect it, but updates in one hour, and this time X2 also requires an update, which takes an hour, while X3 does not require update. Now, both product X2 and X1 should score +49 hours (T0 +24h +24h + time to update of X1 & X2 = T2′) hours. This seems counter intuitive, until you consider the original test; the baseline was set to time of first detection (and called 0 hour). Product X2 could not detect V2 from T0 (X2's last update), X1 always requires update, and X3 is still detecting the viruses from the first day update of T0.

Now, to complicate matters still further, roll forward a further 24 hours, to midnight on day 4 (T3), where another of our obligingly regular viruses appears. This time, both X2 and X3 detect the variant, while X1 does not, but releases an update in 1 hour. Now, the time to update for X1 is +73 hours (T3′), but the time to update for X2 could be either +73 hours (T3′) or +25 hours. Again, this is counter intuitive, but depends on the fact that it is indeterminate whether the previous update to X2 at T2′ added the functionality to detect the virus released at T3. In the first failure of X2, the baseline was T0 because the update at T2′ did not affect the detection of the virus at T1, but in this case, because the detection at T3 occurred subsequent to update at T2′, it is possible that it was influenced by the update at

T2′. This is reinforced in the case of X1, where in no case was any update applied that affected subsequent detection.

Of course, this is a very contrived example, and in real terms, it doesn't as such matter that X3 could detect all sample V3 73 hours before product X1, particularly as V3 probably didn't exist that long ago! However, it is important to note that X3 was able to detect it proactively, and in a sense X1 failed to protect the customer.

There is another part to this problem, concerning the way the results are averaged. I'll return to the three scanner analogy. Let us say that product X3 was detecting 80 percent of new viruses, but on the 20 percent of occasions that it missed updates, it took 30 hours to update (this is of course extreme) the detection for those viruses. Let us say that X2 actually took 8 hours to update each time it required a new signature (50 percent of the time).

Now let us say that there are 20 new viruses released that month, and we take an average of the time-to-update. Table 10.1 gives the results.

Table 10.1 Time-to-Update Results

Product	Actual Time to Update/% missed	Average TtU
X1	1 hour at 100%	1 hour
X2	8 hours at 50%	4 hours
X3	30 hours at %20	6 hours

Now, in this scenario, product X1 is the clear winner in terms of time to update, even though it missed every single virus until it was updated. Product X3 fell into last place, even though it had protected against 80 percent of the new viruses, and Product X2, with detection rates 50 percent better than X1, came in second.

Of course, a smart reader looking at these results will see that there is a further problem. If we were really interested in the actual time-to-update, then we would actually need to exclude the times when no update was needed from our averaging, in which case the table would read as follows.

Table 10.2 Time to Update Results with Exclusions

Product	Actual Time to Update/% missed	Average TtU
X1	1 hour at 100%	1 hour
X2	8 hours at 50%	8 hours
X3	30 hours at %20	30 hours

It is still arguably a more impressive result from X3, but one might be concerned about the lack of response in some cases. From this point of view, there is a value to such tests, but the flaws are undeniable.

While it certainly still seems to be a problem that AV systems are not always updated in a timely manner, perhaps due to misunderstanding of subscription models, or because it's simply not possible, it is inestimably more problematic when systems are updated regularly, but still fail to catch viruses. The abilities of proactive heuristic systems are still subject to update; however, the general principle is one of (hopefully) applying such updates before malware is released. Testing such as the time-to-update test demonstrates only that some products have their failures corrected more quickly than others. It is certainly not a bad thing that products are updated quickly in case of failure, that is not in question; but it is misleading to suggest that it is a decent measure of effectiveness. Thankfully, many companies are moving towards providing a greater degree of proactive detection.

Problem 3: Time of Release vs. Time of First Detection

The time of first detection is often misrepresented as the time of release.

This is important simply because if baseline for a time-to-update test is going to be set at 0 as soon as one product detects it (whether by update or heuristic), the first to see it will have an advantage. The AV industry is a competitive one, it is business, and even the most altruistic researcher will probably do at least a preliminary analysis (and may even provide detection) before releasing a sample to other labs. This is not necessarily a bad thing. At least it should eliminate too much junk being passed around, a situation which would soon lower the credibility of the sender, were it constantly repeated. However, in the case of this test, the first lab receiving the sample has an update available faster than anyone else.

A second issue is that there are always prioritizations. With hundreds of new pieces of suspicious code arriving in AV labs every day, the priority of analysis has to be determined for each piece of code. If a threat looks like it is growing, that is likely to bump something in the queue. But if, for example, one of a vendor's larger customers is having a problem with a particular new malicious program, then that program may be considered a higher priority.
In some (quite possibly in many) cases, the threat is not immediately global, or may be significant only in certain localities (for instance, some Win32/Sober variants only gained prevalence in Germany, so an Asia-based lab may have had higher priorities for that day). A testing lab in a certain locality may test time-to-update and penalize vendors based in different localities, on the basis that they did not react so quickly to an infection localized to a different locality.

Samples are usually obtained in one of three ways:

- Direct submission to a vendor (or several vendors) by a customer or virus writer
- Capture of the sample in a honeypot or other automated capture system
- Submission/sample sharing from another researcher/vendor

In terms of a direct submission, it should also be borne in mind that locality, size, and market penetration may determine which company receives samples. In English speaking markets, the larger players will likely comprise the submission list. In, for instance, Asia Pacific markets, there may be a different pattern for submission. Locality also determines the likelihood of capture in a honeypot (a system deliberately designed to entice malware to infect/attack it). A localized worm such as Win32/Sober.A would have been far less likely to have been caught in a honeypot in Australia than one in Germany (where the worm was almost exclusively found). Submission by other vendors or researchers is not necessarily based on locality, but rather on ad-hoc networks and personal relationships, meaning that those not party to such groups may be excluded, or find significant delay in obtaining samples. So, the problem of "first touch" (i.e., who gets the sample first) is a very hard one to resolve. It may be completely impossible to discern whether the sample was available earlier to company A, enabling them to provide detection faster than company B. This returns us to the question of the importance of time to update. While it is true that the message that time-to-update matters is wrong, even were it right, in the case of something like Win32/Sober.A, it would matter less if a company with no (or very few) customers in Germany did not provide fast detection of that worm.

Determining a release point for a specific piece of malware is similar in difficulty to determining where the wind started: only effect is measurable. Even if it were possible to have a consensus on release date, it would be then necessary to roll back the test set of products to the update each product received before that time, and test from that point. Of course, this does not change the baseline issues discussed in our first problem.

Tools and Traps

Time to Update Conclusion

The message that time-to-update is a simple measure of effectiveness in protection against viruses is demonstrably false, and detracts from the real issue—that of proactive detection. Concentrating on speed of update is surely sending the wrong message to the consumers, giving them the false impression that buying a product that releases a lot of updates very quickly is going to offer them better protection.

Proactive detection and fast update speed is a better guide to performance. However, time-to-update can be a useful guide to a company's general responsiveness if done correctly. A robust test would need to account for all of the factors mentioned (and possibly some beside), or at least show that there was a level playing field for all the products in the review.

It might include two figures for timing, and it might be better combined with a "frozen update" style test.

I'm not necessarily suggesting that such testing is without value, or that it should not be continued; however, *caveat emptor* (let the buyer beware): the results require careful examination and interpretation. (N.B. Some less scrupulous testers have commissioned malware for time-to-update tests, posing other problems.)

To get around some of the problems presented by time-to-update testing, (which can only ever hope to deal with large outbreaks—an increasingly rare phenomenon), a newer type of test has recently come to the fore – the "frozen update" retrospective test.

Frozen Update (Retrospective) Testing

As previously discussed, in the current climate, a real measure of real-world performance is not how fast a product can detect new objects via update, but whether or not the threat can be detected (and blocked) proactively. To address this, testing the heuristic (proactive) capabilities of a scanner is a perfectly valid objective (especially now that most scanners have at least some heuristic capabilities). However, it is as important for such a test to be carried out competently and safely as it is for testing known virus detection. In the absence of a competently administered baseline test set, there is no guarantee that scanners are being tested against valid, working viruses. Some testers, whose competence is already questionable because of lack of direct interface with the AV research community, create further difficulties for themselves and for those who rely on their testing, if they don't publish information on their testing methodology, especially sample validation. Unfortunately, (whether due to time or financial constraints) popular magazine reviews have frequently fallen prey to such problems, though in recent years, many have outsourced the actual malware testing portion to more competent bodies such as AV-Test (see section on testing bodies).

I've already mentioned that (for good reason) the AV industry is reluctant to condone creation of new malware or viral code, even if it's just for testing. That said, it isn't actually necessary for anyone to create viruses to test heuristics.

"Retrospective testing" involves testing a scanner that hasn't been updated for a period of time—i.e., the product updates are frozen at that time (three months is a period commonly chosen)—with validated malware that has appeared since the last update applied to the test-bed scanner. This provides reasonable assurance that heuristic capability is being tested, rather than the detection of known viruses by virus-specific algorithms. This kind of test by no means lessens the need for competent testing, but it avoids the ethical and practical difficulties associated with the creation of new viruses for testing purposes. However, it by no means eliminates the need to validate samples, or to carefully construct meaningful tests.

Almost all of the major AV vendors provide daily (or more frequent) detection updates, so testing a scanner when it's three months out of date doesn't say very much about what it detects today. Bear in mind that both heuristic and virus-specific detection patterns will be added over that period. A more valid approach might be to test the capabilities at different points, or to test with a specific virus to determine the first point at which detection occurs. Clearly it's worth noting if a scanner was capable of detecting malware before it was known to exist.

Both AV-Test and AV-Comparatives have performed some type of retrospective frozen update testing, with interesting results. It is worth bearing in mind though, that statistical significance is an issue. If testing purely with WildList viruses, the sample set may be comparatively small, and could give biased results. It's a decent enough indication of good heuristic capabilities, though, if all that is taken into consideration.

A Few Words on False Positives

False positive (FP) testing is a valuable indicator of a product's suitability for use in a production environment. A product that produces false alarms on common files can cause as many problems for network administrators as real malware might.

A good FP test set includes a large set of clean files (i.e. files that do not contain viruses or any other type of malicious code). These are usually files that are found on a normal system, and as far as possible, they should be un-archived and in their native state. There may also be some files that are known to cause problems for scanners, giving rise to FP detections, and these too may be included in such sets. FP test sets require a degree of maintenance also, as it is arguable whether a FP against an esoteric file from 10 years ago is as relevant as a FP on the latest utility suite of a software manufacturer.

In recent years, as the threatscape has changed, I've begun to question the real need for FP testing (at least as a criterion for disqualification from certification) of a product, not because it would be easier to allow FPs, but because there are so many gray areas now, especially around spyware and adware. However, it is desirable to reduce to an absolute minimum the FPs caused by a product, and so it forms a legitimate and interesting part of most anti-malware product testing.

A Checklist of Do's and Don'ts in Testing

In this section, I'm going to list a few *do's* and *don'ts* with brief reasoning, which summarize the preceding discussions. The reader who doesn't feel like wading through the full text may find this of interest. There are also some items that don't need too much extra discussion, so they only appear in this list.

Testing AV products is not just a case of finding a few files and scanning them to see if they are detected. Testing must be done in a rigorous, methodical manner to ensure that a correct result is achieved. This guide aims to point out the most important criteria for testing, and give some guidelines for testing the various different types of AV scanner.

If you're an aspiring tester, here are some good tips: if you're evaluating tests, because you may base a purchasing decision on them, then look out for testers who show that they avoid the don'ts and do the do's!

First of All, Here's What Not to Do!

Don't test against viruses that you've modified or specially created. It is considered highly unethical in the AV research community, to create new viruses for any purpose, particularly for testing. In some cases, testers have added parts of a virus to other files (for instance text files) to see if they are still detected. This is nonsensical for all the reasons of correct validation stated in the fuller discussion in the introduction to this chapter.

Don't test against renamed virus files (i.e., files with the filename extension changed so that the operating environment sees it as non-executable; in other words, turned into a latent virus.) Although most products now scan all extensions by default, there is a good argument against detecting a non-executable file, as it is not, in that state, a threat. In some cases, a file renamed to a harmless state might not be detected, but on renaming it would be. Testing against the real file, in its natural state puts the test into the real world.

Don't test against damaged/non-viable malware samples. In other words, validate the samples you are going to test against, to ensure they really are malware. Testing isn't about how many *objects* an anti-malware product will detect, but about how many *real-world* malware samples it will detect.

Don't test against files that you haven't validated, whether on the grounds that it is detected by a specific product, or because it came to you from another tester/researcher. Do your own verification. A detection of a non-malware object counts as a FP (except in the specific circumstances where the damaged offspring of a virus might be detected because many such files are produced, and customers may want to have them detected to reduce nuisance value. Such files are usually detected with an extension like .dam (e.g., W32/Magister.b.dam)). Intended but failed malware is a valid target for identification, as long as it isn't misidentified as active malware.

Don't alter malware. Viruses should be unaltered from their normal ItW state, and should be viruses that already exist. Don't be party to the creation of new viruses, or to the alteration of viruses to create new variants for any purpose, including the purposes of testing. Altering any virus (presuming that it is replicable afterwards) effectively creates a new virus. If it isn't replicable, it shouldn't be detected (or at any rate identified) as a virus.

Don't test with incorrect settings. You might well ask what settings *are* correct. Typically, tests are done with default (out-of-the-box) settings, or with best possible setting (i.e., everything enabled). Testing with other settings (unless the methodology is clearly stated, and the products under test are *all* capable of being similarly configured), is prone to flaws, because it becomes a test of apples against oranges. As far as possible, the playing field should be a level one.

Don't test over a network. Files should be on a local hard drive or a locally mounted drive, unless you are specifically testing scanning on network drives. This is particularly important if scan speed is being measured. This also avoids problems caused by dropped packets or performance issues caused by the network rather than the product.

How to Do it Right!

Do use real malware samples, unaltered samples that can be verified as working (in the case of viruses this should be through replication). (See the section on sample verification earlier in this document.)

Do use clean files for FP testing! These should be normal files that would exist on an end-user's system, rather than files that have been specially created to look like malware or to "trick" AV scanners. Tests are supposed to show something about the real-world capabilities of anti-malware products.

Do ensure statistical integrity. Sample selection should be made on a statistically sound basis, and the number of samples used in a test is important. Testing against two or three samples, or even 10 or 15 samples, does not constitute a statistically significant set of samples. The WildList contains several hundreds of different viruses on its top list (the most significant list) of currently ItW viruses. There are several thousands of new malware objects reported weekly. The smaller the set tested against, the more statistical errors will be introduced into the test. Larger test sets can also reduce the impact of corrupted samples or poor verification.

Non-detection Testing Parameters

The scan speed and performance impact of anti-malware products are important considerations (although not as important as malware detection). In running any application, there will be some performance impact, and this is no different when scanning for malware. A normal system can contain hundreds of thousands of files. A scanner that slows the machine down significantly, perhaps causing it to be unusable while scanning, is undesirable, but it is also important that any performance impact should be for as short a period as possible. For this reason, scanners need to operate fast, and to consume as few system resources as possible. The impact of scanning on processing needs to be carefully measured. To measure the true speed of a scanner, make sure that it is not caching results for subsequent runs. (This is a perfectly legitimate way of improving performance, but if you are trying to measure actual scan time or performance impact against new objects, it must be taken out of the equation in order to keep the playing field level.)

Scanning speed is relatively simple to determine. Select a group of files and scan them, measuring how long it takes each scanner to complete the scan. Repeat the test a few times (three to five) and calculate the average time. The files used should not contain compressed or archived files, unless archive scanning is specifically being measured. Many scanners do not scan archives in default mode, so this will bias the result. Some tests do show scanning speed figures against various types of file. In particular, Virus Bulletin publishes figures for archive (*.zip*) scanning and OLE object scanning.

It is important to use clean files. If you know something is "dirty" then it's quite legitimate to take action such as stopping the scan at that point, until some resolving action is taken. Performance testing (i.e., measuring the amount of overhead or latency on a system in normal use with an on-access scanner running) is often done with specialized benchmarking software. It's important to ensure that the test is well thought out, and accurately reflects a normal working system.

Conclusion

Once the minefield that surrounds testing has been successfully navigated, it's important to read published methodology for a given test. (If it doesn't have a full published methodology, then you should wonder whether it is a worthwhile test). Reading the methodology gives you an idea of what you should be getting from the test, and it will also give you some idea of how sound the results are.

Not only should the methodology be sound, but it is also important that it is consistently applied. Not all products operate in the same way, nor do they produce their results, log files, or detections in the same way. However, the criteria laid down for the test must be broad enough to deal with such problems. Even if the methodology is not ideal, the application should be consistent in every case.

For instance, I could decide that I'm going to do a test where I will not allow any detections that are not specifically identified by name (i.e., I will discount all suspicious notifications). If one product has an output against a virus called W32/ImaginaryVirus.A that says "a variant of W32/ImaginaryVirus" and another that says "W32/Imaginary.gen," should I in this case count one, both, or neither? Absolutely strictly, if my criterion is exact identification, I can accept neither, even though I may feel that the generic detection *.gen* is good enough. Actually, the two are roughly equivalent in terms of their meaning, and in any sane test, they would both count as detection. However, it is important that if I've defined a methodology (and it should be clear, written down, and comprehensive), I stick to it. If the methodology proves to be flawed, I should revisit that for the next test, rather than apply different standards to different products.

Independent Testing and Certification Bodies

The most well known (currently active) testing organizations, which test and/or certify antimalware products, and which are generally considered to be competent in this area are:

- Virus Bulletin (http://www.virusbtn.com) with theVB100 award.
- ICSA Labs (http://www.icsalabs.com) with ICSA Labs certification.
- WestCoast Labs (http://westcoastlabs.org) with CheckMark certification.
- AV-Test.org (http://www.av-test.org) (mainly custom tests for magazine reviewers)
- AV Comparatives (http://www.av-comparatives.org) (uses a statistical model of scoring)

Each organization has slightly different goals and methods, and while they are all commercial organizations, their focus (and methods) differ somewhat. I'll give a brief overview of each organization and their testing methods. The information is largely summarized from the Web sites of the organizations themselves.

VB100 Awards

Virus Bulletin is the AV industry journal; its testing methods and reputation are internationally known and respected. The VB100 award is particularly well known as a test benchmark to which all AV products aspire, but there are some misconceptions about what it actually means. Most importantly, the award of a VB100 award does not imply that the product can detect every known virus (this was a prevalent myth when the award was called the "VB 100%" award.). Rather, it means that it has met a well defined set of tested criteria, and is suitable for use in real-world situations.

The VB comparative tests focus on several functional aspects of on-demand and on-access scanners:

- Detection rates
  - ItW viruses
  - "Zoo" viruses
- FP rates
- Scanning speed
- Performance overhead

With regard to the actual VB100 award, there are two portions to the test, and these are the only criteria that count towards the actual award of the certification.

Firstly, there is the test against ItW viruses, as defined by the WildList. To receive the award, the tested product must detect every one of the samples on this list. The second criterion is that the product must *not* detect any objects in the VB clean set (see the discussion on FPs for more on this). In some cases a 'suspicious' flag has been permitted where the object may fall into a 'grey' area such as joke files. See also the Note on "Tools & Traps" at the end of this chapter. If those two criteria are met, the product receives the VB100 award, regardless of misses in the zoo sets, or performance issues. Those other parameters are shown for purposes of interest. It is important to note that this test is a "one chance only" test, and the results are not made known to the product vendors before they are published.

As mentioned above, VB tests also examine several other areas, which, while they do not have a bearing on the award of the 100 percent certification, demonstrate the capabilities of the tested product against a wider set of malicious software, and its impact on the system it is deployed on. The performance of a product in these other tests forms the basis of the written review that each product receives to accompany the detection test results.

The magazine review also details any reasons for failure to achieve the VB 100 percent award.

ICSA Labs (a Division of Cybertrust)

The goal for ICSA Labs Certification is to enhance and improve security implementations of network and Internet computing. This, in turn, will improve commercial security and its use of appropriate security products, services, policies, techniques, and procedures.

ICSA use a very similar testing methodology to Virus Bulletin. Whereas the VB 100 is awarded to a specific product on a "snapshot" basis, with no failure correction period allowed, ICSA Certification concentrates more on requiring a product to achieve and maintain a particular level. Products are tested each month, and there is a grace period (usually a week or so) for the company to rectify any failures.

Thus certification is not awarded to a single version of the product, but inherently certifies all versions and updates of the product. This is achieved by a contractual agreement with the company whose product is under test, that the product will be maintained to the tested standard throughout the certification period.

All products are recertified yearly (depending on the performance).

ICSA do not publish test results *per se*, but a product that holds ICSA certification is assured as meeting their stringent standards for testing.

ICSA offers a range of certifications, not all related to anti-malware: they also certify firewall and Intrusion Detection Systems (IDSes). It is common for products to have several ICSA certifications in different areas of competence.

NOTE

Newsflash! As we were completing this chapter, the news broke that Verizon was in the process of buying CyberTrust, which means they will own ICSAlabs and therefore the WildList Organization. The effect this will have remains to be seen.

Checkmark Certification

The Checkmark certification is operated by West Coast Labs, a division of Haymarket Publishing. They operate various certifications, which certify products to given levels. A brief overview of the meaning of each certification is presented below (definitions are summarized from the Westcoast Labs Web site).

Anti-virus Level 1

West Coast Labs has a virus library with many thousands of different viruses and variants, collected through a proprietary harvesting process. For a product to be certified to AV Checkmark, Level One, the product must be able to detect all those viruses that are ItW. This gives a clear and independent indication to end users of those AV products that they can be relied on to perform to Level One functionality.

Anti-virus Level 2

For a product to be certified AV Checkmark, Level Two, the product must comply with Anti-Virus Checkmark, Level One and, in addition, disinfect all viruses on the ItW list that are capable of disinfection.

Disinfection is defined as meaning that the virus is removed (or substantially removed) from the infected object and the object is usable. The application/program should not hang or crash the machine; the machine should boot up correctly, and an infected document should be restored as far as possible to its former uninfected state without loss of data.

Trojan

For a product to be certified to the Trojan Checkmark, the product must be able to detect all Trojans in the West Coast Labs Trojan test suite, which is regularly updated. The product should not cause any false alarms (based on testing against the West Coast Labs false alarm test suite).

Anti-Spyware

For a company to be certified to the Spyware Checkmark, the product must be able to detect Trojans, keyloggers, cookies, and other assorted malware in the West Coast Labs Spyware test suite. The product should not cause any false alarms (based on testing against the West Coast Labs alarm test suite).

Companies with a current certification in each level are listed on the Westcoast Labs website. See www.westcoastlabs.org/checkmarkcertification.asp for more details on Checkmark certification.

AV-Test.org

AV-Test.org is a project of the Business-Information-Workgroup at the Institute of Technical and Business Information Systems at the Otto-von-Guericke-University Magdeburg (Germany), in association with AV-Test GmbH. At regular intervals they test AV, anti-spyware and personal firewall software on behalf of the producers and for magazines.

AV-Test provides testing services for many magazines, which are then able to publish their results in comparative reviews.

AV-Comparatives.org

AV-Comparatives.org provide independent comparatives of AV software. They limit their selection to the best-performing anti-malware products, and they only test products that fulfill their conditions and minimum requirements.

They perform quarterly testing, alternating between standard tests against their malware database with fully updated products, and frozen-update retrospective tests.

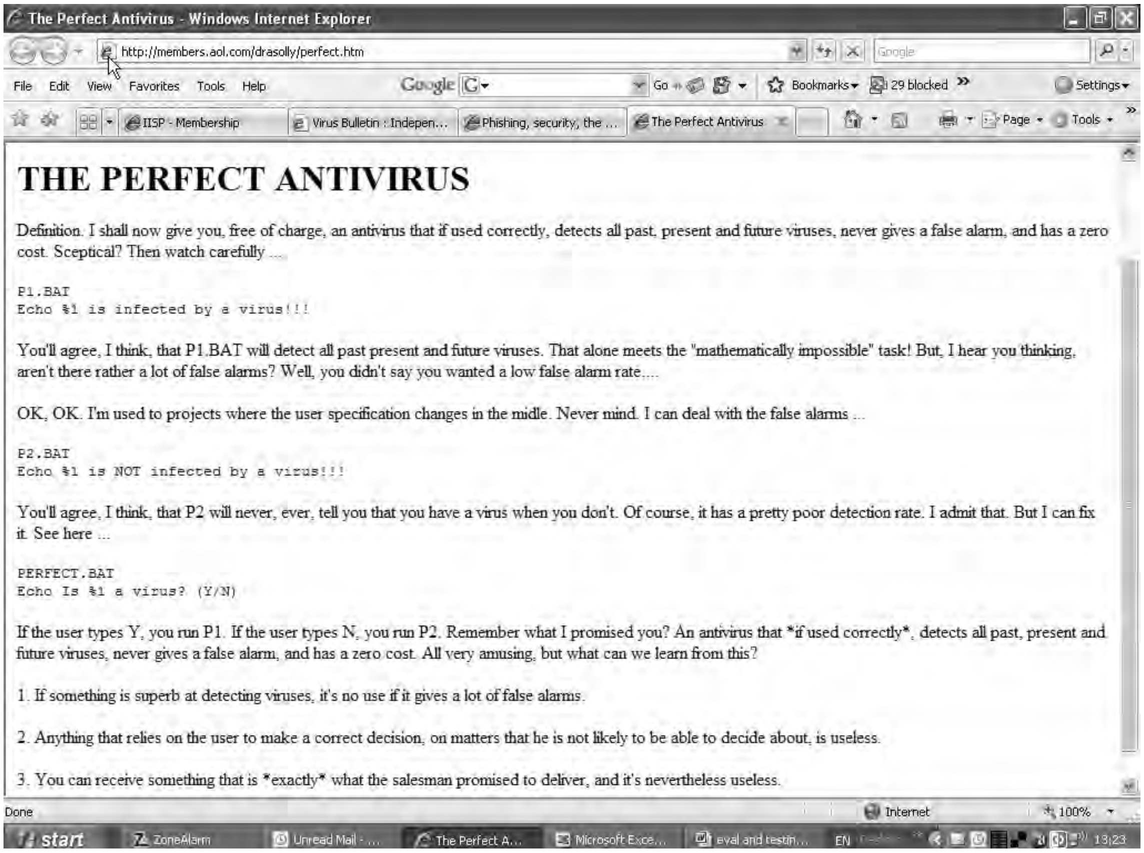
Summary

We'd like to think that we've told you everything you need to know about evaluating malware management solutions, not to mention how to test them. Of course, we haven't, and we can't. If there's one thing we've learned from our long association with AVIEN and AVIEWS and the security world in general, it's that there are no "one size fits all" solutions, and if you find yourself looking at a TOAST solution ("The Only Antimalware Solution That you'll ever need," to slightly misquote Padgett Peterson), expect less than you are promised.

Evaluation in the real world isn't about finding the best-of-breed solution that lets you tick that box and move on to the next job. It's about painstaking research to find the imperfect solution that best matches your particular environment, and a future involving lots of monitoring, reviewing, tweaking, filling gaps and cracks, and being prepared to re-evaluate your present approach. We can't even tell you all the issues you need to look at. All we can do is list those that we've found most important over the years, and leave you to pick and choose the ones that are most applicable from where you stand.

As for testing, I'm afraid we haven't been able to tell you everything you need to know to set up your own testing facilities, either. Of course, if you put the information on testing in this chapter together with some of the other chapter sections on Do It Yourself (DIY) analysis and defense-in-depth, you might have some useful testing groundwork, but our main intention here was to give you enough information to avoid some of the most common pitfalls, either as testers yourselves or as readers of test reports. Furthermore, we intend to come back to this fascinating topic at greater length in another project. It occurs to us that you may feel that we've promised more than we've delivered. So allow us to introduce to you to Dr. Alan Solomon, AV pioneer, whose "perfect AV" mini-article (used by permission, and with due acknowledgement to the author) makes very pertinent points about AV (and antimalware) technologies. You can find the whole article at http://members.aol.com/drasolly/perfect.htm. And yes, you can automate that key capture process with CHOICE (found in some versions of DOS and Windows), or a little assembler code and the IF....ERRORLEVEL commands. (If you're really sad enough to want to try this out, feel free to contact us for more information on how to do it.)

Figure 10.4 Dr. Solomon's Perfect Antivirus


The Perfect Antivirus - Windows Internet Explorer
http://members.aol.com/drasolly/perfect.htm
File Edit View Favorites Tools Help
IISP - Membership
Virus Bulletin : Indepen...
Phishing, security, the ...
The Perfect Antivirus
Done
Internet
100%
start
13:23

THE PERFECT ANTIVIRUS

Definition. I shall now give you, free of charge, an antivirus that if used correctly, detects all past, present and future viruses, never gives a false alarm, and has a zero cost. Sceptical? Then watch carefully ...

```
P1.BAT
Echo %1 is infected by a virus!!!
```

You'll agree, I think, that P1.BAT will detect all past present and future viruses. That alone meets the "mathematically impossible" task! But, I hear you thinking, aren't there rather a lot of false alarms? Well, you didn't say you wanted a low false alarm rate....

OK, OK. I'm used to projects where the user specification changes in the midle. Never mind. I can deal with the false alarms ...

```
P2.BAT
Echo %1 is NOT infected by a virus!!!
```

You'll agree, I think, that P2 will never, ever, tell you that you have a virus when you don't. Of course, it has a pretty poor detection rate. I admit that. But I can fix it. See here ...

```
PERFECT.BAT
Echo Is %1 a virus? (Y/N)
```

If the user types Y, you run P1. If the user types N, you run P2. Remember what I promised you? An antivirus that *if used correctly*, detects all past, present and future viruses, never gives a false alarm, and has a zero cost. All very amusing, but what can we learn from this?

1. If something is superb at detecting viruses, it's no use if it gives a lot of false alarms.

2. Anything that relies on the user to make a correct decision, on matters that he is not likely to be able to decide about, is useless.

3. You can receive something that is *exactly* what the salesman promised to deliver, and it's nevertheless useless.

Tools & Traps

Are you Positive?

There is a minor point of contention in FP testing, in that there is an ambiguity in the way some files (or other objects) are flagged as suspicious.

- If a non-malicious file is flagged as malicious, that's a clear FP. (We won't get entangled in the possibility that a file might be replicative yet not malicious!)
- If a malicious file is not flagged, that's a clear false negative.
- If a malicious file is flagged as suspicious, that's usually taken as a correct heuristic detection.

- If a non-malicious file is flagged as suspicious, that isn't usually taken as a FP.

If the last of these eventualities takes place in the real world, the end site or end user may decide to play safe and assume malice. So it can be argued that the vendor evades the risk of FPs by transferring the decision and therefore the risk to the customer. (See point two in the conclusion to Dr. Solomon's article.) However, most reputable testers seem comfortable with this paradox, which may be further mitigated if the product makes the distinction between "malicious" and "suspicious" very plain to the customer or offers good technical support to customers wherein such an issue can be swiftly dealt with.

Before we wrap this chapter up, we want to point to a couple of other resources: "Real World Anti-Virus Product Reviews And Evaluations – The Current State Of Affairs" by Sarah Gordon and Richard Ford (http://csrc.nist.gov/nissc/1996/papers/NISSC96/paper019/final.PDF) discusses problems and alternatives relating to AV product evaluation, including input from the research community after the original paper. While it's a little dated, it's still an excellent discussion of some basic issues.

"A Reader's Guide to Reviews," originally published in "Virus News International" and credited to Sarah Tanner, was actually written by Dr. Alan Solomon (is there no getting away from this man?). Since it dates back to 1993, it's obviously not current, but as a hint of the many ways that an antimalware product review can be biased (intentionally or otherwise) it's indispensable. (www.softpanorama.org/Malware/Reprints/virus_reviews.html)

Finally, we'd like to emphasize that we haven't finished with the topic of evaluation and testing. We will revisit it elsewhere, at much greater length. See www.smallblue-greenworld.co.uk/pages/avienguide.html for further information.

Solutions Fast Track

Antimalware Product Evaluation

☑ Product evaluation falls largely into six main areas: Configurability, Cost, Ease of Use, Functionality, Performance, and Support Issues.

☑ Evaluation methodologies include comparing reviews from general information resources, specialist reviews, and in-house evaluation and testing.

☑ Configuration is highly subjective, due to the wide variation between requirements in different environments.

☑ There's a great deal more to cost than unit cost. Consider also deployment costs, administration costs, support costs (in-house and external), and incident management.

- ☑ As the range of threats has widened, so it has become less practical to think in terms of a single point solution (antivirus, for example).
- ☑ It's not only detection performance that matters, but speed, latency, and impact on overall system performance.
- ☑ Support issues cover two main areas: product knowledge and malware knowledge. Consider issues such as information flow and documentation as well as direct access to first- and second-line support.

Product Evaluation Checklist

- ☑ Cost breaks down to unit cost/licenses, extras, administration, and incident management.
- ☑ Performance breaks down to detection rates, accuracy, scanning targets, execution speed, repair, and compatibility.
- ☑ Functional range includes platforms and entry points covered, and integration with other defensive components.
- ☑ Support can cover many aspects: tech support, upgrades and patches, definitions updates, product development information, training and education, and documentation.

Antimalware Testing

- ☑ The antimalware industry has concerns about testing in the areas of ethics, safety, competence, and methodology. One particularly sensitive issue is the risk of skewing results by mistaking junk for truly malicious programs.
- ☑ You can't validate a sample simply by accepting the diagnosis of one or more scanners. If your sample verification is unsound, your whole testing methodology is invalid. Viruses have to be replicated at least to the second generation.
- ☑ This section is a guide to good testing practice, but it isn't "everything you need to know to be a malware product tester."
- ☑ Testing polymorphic viruses requires the use of multiple samples. Detecting one sample is no proof that other samples of the same polymorph will be detected.
- ☑ The WildList maintains a collection of validated samples of replicative malware known to be ItW. (However, competent testers re-validate samples they receive.)
- ☑ Non-replicative malware (Trojans) pose extra problems in detection testing, due to difficulties of defining essential characteristics.

- ☑ Time-to-update testing measures the time taken for a product to update after an arbitrary point in time. It can, however, be very misleading as a measure of protection capability, baseline setting for heuristic detection, and in terms of confusing time of release with time of first detection.
- ☑ Retrospective testing enables the testing of heuristic capabilities without necessitating the creation of new malware. However, it's not always clear how much the performance of a "frozen" version reflects the capability of the current version.
- ☑ FPs can be very damaging to an enterprise's business processes, so FP testing can be very important to the test report reader, but it's debatable how much weight should be given to FP problems in a certification test.

Independent Testing and Certification Bodies

- ☑ There are a number of bodies considered reliable and competent by the AV community to test and certify: in particular, Virus Bulletin, ICSA Labs, West Coast Labs, AV-Test.org, and AV-Comparatives.
- ☑ These usually focus on testing criteria such as detection, FP rates, and performance, all of which are comparatively easy to test objectively.
- ☑ They vary widely, however, in terms of the range of malware types they address.

Frequently Asked Questions

Q: Surely all antimalware products detect much the same range of malicious programs?

A: This is less true than it was a few years ago, for two main reasons.

- The range of malicious programs commonly encountered now is much wider than it used to be, and non-replicative programs in particular pose more problems in terms of heuristic detection. Also, while what we used to call AV programs detect a good many non-viral malicious programs, they aren't all equally focused on the whole range of malware. So while the major players all tend to perform comparably on replicative malware, they may vary dramatically on other malware types.
- Modern malware (replicative or not) presents quite different problems in detection. Despite all the advances made in scanning technology (such as advanced heuristics), it's easier for the bad guys to evade detection with short spamming runs, multiple packing, and so on.

Q: Who has the resources to do in-house testing?

A: Good, safe detection testing takes appreciable time as well as expertise, and an isolated network. However, testing for compatibility, configuration, deployment, network impact, update and upgrade, and so on can be carried out on spare machines on a production network, and is fairly painless.

Q: I understand the need to get the best possible performance, but my budget is very restricted.

A: Been there, done that, set fire to the tee shirt out of sheer frustration. In fact, once, after negotiating a particularly good deal one year, I discovered the next year that my budget had disappeared altogether, in the expectation that I'd be able to repeat the coup in perpetuity. It's true that most of us have to fight the bean counters tooth and nail every time, and mostly they are infinitely more impressed by low unit cost than by performance metrics.

Q: Can you recommend a resource for malware-related metrics?

A: There've been many attempts to provide easy plug-in spreadsheets and other forms of modeling over the years. Unfortunately, the malware management field has proved particularly resistant to standards of measurement. Andrew Jaquith's book "Security Metrics: Replacing Fear, Uncertainty and Doubt" (Addison-Wesley, 2007) won't give you all the answers, but it's well worth reading as an introduction to metric techniques in general.

Q: Why do you say that detection isn't important?

A: That's not quite what I said. What I'm trying to say is that it doesn't matter how good detection is if the product is unusable, or beats your business processes to a pulp. Detection is very important, of course; however, the days of near-100 percent detection of all the threats you need to worry about are long gone.

Q: What's the difference between the latent malware problem and heterogeneous malware transmission.

A: HMT is concerned with the spread of malware, especially replicative malware, from an infective or infected system to a vulnerable system via systems that aren't themselves vulnerable. (The expression was probably coined by Peter Radatti: at any rate, I first encountered it in the 1992 paper "Heterogeneous computer viruses in a networked UNIX environment" in the Proceedings of the First International Virus Prevention Conference and Exhibition (NCSA), Washington, DC.) A latent virus is one that hasn't been executed in its present environment, but that doesn't necessarily mean it can't be.

Q: How do latent viruses relate to latency in other virus issues.

A: In "Viruses Revealed" I suggested that "dormancy" might be a better term in this respect, since the term latency is used to denote impact on performance (e.g., the impact of firewall processing on network throughput).

Q: Isn't it rather convenient for the AV industry to protest that virus testing methodology and creation of viruses for test purposes is unethical?

A: You could look at it like that, but there are sound reasons for this viewpoint.

- Many in the industry are adamant that it's inappropriate for those within the industry to create new malware, even for research purposes (that probably has a lot to do with the persistent allegations that the AV industry creates viruses in order to keep itself going).
- There is a safety issue. You may not think it's that hard to keep a test network isolated, but there's a feeling that those who don't understand the other points may not understand the importance of safe practice, either.
- The use of invalidated samples, poorly conceived modifications to malware, and newly created malware can seriously bias the results in obvious and less obvious ways. The most obvious problem is that where there is an invalid sample which is incorrectly identified as viral or malicious by product A and not by product B, product B is unfairly and inappropriately disadvantaged. Even if there is no intention to skew the results, there is a question of ethical responsibility.

Q: Aren't macro viruses specific to multiple operating systems?

A: In a sense. Macro viruses and some forms of script virus are actually specific to an application, not to an operating system. However, their ability to replicate and the effectiveness of any payload may vary according to the operating system, or between versions of a given operating system.

Q: Isn't the fact that the WildList is published monthly at most a drawback in terms of WildList testing?

A: It does lessen its usefulness as the main component of a detection test. But it still offers a useful way of assessing a scanner's ability to detect a baseline set of properly validated samples that a front-running product shouldn't normally miss.

Q: Why is there no WildList for Trojans?

A: The idea has been discussed. The difficulties include:

- Sheer volume of samples, arising from present-day patterns of distribution, raising resource difficulties in terms of validating a core test set.
- The additional technical difficulties of defining and automating the detection of Trojans

Chapter 11

AVIEN and AVIEWS: the Future

The Anti-Virus Information Exchange Network (AVIEN) was born in a very different time to the present. Viruses dominated the threat landscape, and mass mailers were a serious problem and getting worse. Other malware, however, was a comparatively minor problem.

- Zombies were a specialized form of Trojan that antivirus programs dealt with easily by detecting them on the desktop.
- Most people thought that robots (bots) were something that Isaac Asimov used to write stories about (http://en.wikipedia.org/wiki/Robot_Series).
- Anti-virus (AV) programs did catch some Trojans (they always had), but when asked specifically about improving Trojan detection, some vendors would still say "Trojans? But this is antivirus software!"
- Spam was someone else's problem.
- Spyware was something to do with James Bond.
- Rootkits were something that a few UNIX administrators worried about.
- Some vendors had a "Gods and Ants"/"Nanny knows best" view of customers. "This is what we detect, this is how we detect it, this is how you should install it, and this is what you are entitled to expect from us. Just buy the packages and we'll take care of the rest."

Today's world is very different. Certainly the threatscape has changed. Not that AVIEN (or AVIEWS, the Anti-Virus Information and Early Warning System) can claim all the credit or discredit for that. The "hey, look at me!" virus, reflecting the writer's yearning for fame, notoriety, and appreciation of their Überhacker status, took second place to stealth crime-ware, intended to stay hidden and active for as long as possible in order to maximize profit. With stealth comes a diminishing of the effectiveness of older models of detection based on "fast and far" dissemination. Viruses have by no means disappeared, but their "market share" has declined, and the antivirus industry and its customers have had to come to terms with that. Certainly, the fact that this is very much a book about malware rather than about viruses reflects that shift in focus.

Nowadays, AVIEN members are rarely restricted in the scope of their work to virus management, or even malware management. Packages and appliances that address these areas of security have to be seen as part of a multilayered strategy, often essentially customer-driven, not solution-driven. The name of the game is not antivirus or antimalware, but network security, application security, desktop security, and so on.

In the same way, many of the companies we used to call AV companies have moved away from that single market view, and market themselves as security vendors rather than AV vendors. Their product ranges include at least some of the following: anti-malware? desktop firewalls, spam filtering, antiphishing measures, education, outsourced service provision, and so on. Of course, they may choose to cover any other area of security (or outside security).

AVIEN has changed, too. Most dramatically, with the inception of AVIEWS, a meeting place for the high octane enterprises represented in AVIEN, the security industry, and smaller customer groups that don't qualify for AVIEN. Some independent and industry researchers have always enjoyed productive formal or informal links, but AVIEWS extended those networks. In the same way, of course, other groups such as the Anti-Spyware Coalition (www.antispywarecoalition.org/) have moved towards incorporation of consumer groups rather than restricting membership to product vendors. The WildList Organization has long included corporate reporters as well as AV industry researchers, despite its association in the popular mind with the AV establishment. (While the WildList has lost some of its immediacy and relevance to the overall threatscape, its importance to antimalware testing remains, for the moment. Furthermore, like AVIEN, the WildList Organization itself is undergoing changes intended to maintain its relevance to the modern threatscape. How these changes will be affected by the acquisition of the parent company by Verizon remains to be seen, however.)

Where the AV industry was originally publicly uncomfortable with the existence of a vociferous consumer group, it quickly seemed to see the advantages of interfacing with a community that saw some kinds of threat quicker than the industry did, and was able to supply meaningful input into what was affecting enterprises and how they needed to address it. The security industry and its customers understand each other a little better than they did, and some of that *is* down to AVIEN. Meanwhile, initiatives such as the AVIEN online conferences and sponsored events at Virus Bulletin conferences, ensured that AVIEN and AVIEWS stayed prominent in AV circles

However, the balance has shifted a little. Again... Because of changes in the commonly seen malware types, dissemination patterns and media, and so on, there are fewer attempts to broadcast a single malicious program to the entire internetworked universe. Also, vendors have augmented their methods of acquiring samples through means such as honeynets and honeypots. Virus discussion, though still of intense interest to some among us, is not enough.

So AVIEN and AVIEWS are changing again. Changes to the Web sites are already underway, and there is much discussion about broadening our organizational image as a security forum, rather than as a niche group dedicated to the discussion of viruses. Other discussions center around moving to a more open/communally responsible organizational structure. To this end, Robert Vibert, who has long guided AVIEN in his role as Administrator/CDO, is working with David Harley, who takes over as Transitional CDO to nurse AVIEN through its restructuring, and Andrew Lee, Administrator of AVIEWS. AVIEN has always been based on the agreement and ideas of the membership, augmented by the efforts of volunteers such as the Adjunct Administrators, a CDO Advisory Board, and project leaders, but we figure that someone has to dust the floor and take out the trash while the membership is rebuilding the walls. But while some of the detail on where we're going next is still blurry, here are some of the ideas that have been raised.

- Ongoing and expanded collaborations and information sharing with other security groups
- Development of the online conferences
- Other collaborative ventures such as blogging and other communal Web spaces
- Adopting a Warning, Advice and Reporting Point (WARP) or WARP-like model (see www.warp.gov.uk and the chapter on education for more on WARPs)
- Development of the present book into other publishing directions. Some of us have worked with Syngress previously, and hope to do so again, but there may be smaller projects that AVIEN can take on better independently.

We therefore invite you to watch this space—no, not the one between these clauses, but this one at www.avien.org, and this other space at www.aviews.net—and, if you aren't already a member of one or both organizations, consider joining us in shaping what we think will continue to be a major landmark in the security landscape for years to come. If you've enjoyed this book and found it useful, think how much more useful it could be with your ideas as input.

David Harley, Transitional CDO
Robert Vibert, Immediate Past CDO
Andrew Lee, AVIEWS Administrator

Appendix A

Resources

Solutions in this chapter:

- Customer Power
- Stalkers on Your Desktop
- A Tangled Web
- Big Bad Bots
- Crème de la CyberCrime
- Defense in Depth
- Perilous Outsorcery
- Education in Education
- DIY Malware Analysis
- Antivirus Evaluation and Testing
- Other Resources

Introduction

When I look at a security book, either as a potential purchase or because I'm reviewing it, there are a couple of things I look for straightaway: the quality of the index (I do own excellent, unindexed books, but they are a pain to look up a reference in a year or two after you last read 'em!), the references, and the resources (pointers to further information.) I have a severe mistrust of any book whose author believes that the security landscape will never change and that his book is the only one on the subject that you'll ever need to read.

Well, the indexing is largely taken care of by the publishers, here as with most of the books I've participated in. The references are largely inline to the chapter text, but, for your convenience, some of them are in this section too, neatly bundled in one place with a few more resources that aren't in the text, just to keep them company. I haven't included obvious sources like Wikipedia. I did consider reviewing briefly some of the books I've mentioned: instead, though, I'm going to give you the URL for the best security reviewing resource on the web: http://victoria.tc.ca/int-grps/books/techrev/mnbk.htm. This is the root page for Robert Slade's Computer and Technical Reviews. Actually, Rob by no means restricts his reviewing to security: the topics include:

- Year 2000 Problem/Millennium Bug
- Applications, Operating Systems, and Platforms
- Artificial Intelligence and Artificial Life
- Data Communications Books
- Dictionaries and Glossaries
- Fiction (with Computer and Technical Themes)
- Computer and Technical History
- Internet and Related Books
- Miscellaneous
- Programming and Languages
- Virus and Security Books

So that ought to give you something to be getting on with. You might prefer, though, to flick through the hypertext index at http://victoria.tc.ca/int-grps/books/techrev/review.htm. Your choice.

Customer Power

AVIEN web site: www.avien.org
About AVIEN: www.avien.org/avien.html
Joining AVIEN: www.avien.org/join-avien.html
Some of the people in AVIEN: www.avien.org/people_in_avien.htm
AVIEWS web site: www.aviews.net
Joining AVIEWS: www.aviews.net/content/category/3/8/30/
Virus Bulletin: www.virusbtn.com
"Setting the Record Straight" (David Harley): in Virus Bulletin, November 2001.
WildList Organization: www.wildlist.org
EICAR: www.eicar.org
CARO: www.caro.org
Rob Rosenberger: www.vmyths.com
SANS: www.sans.org
AusCERT: http://www.auscert.org.au/
ESET papers, including a series by David Harley & Andrew Lee: www.eset.com/download/whitepapers.php
Fred Cohen & Associates: www.all.net
"I'm OK, You're Not OK": David Harley, Virus Bulletin, November 2006 (www.virusbtn.com/virusbulletin/archive/2006/11/vb200611-OK)
Sarah Gordon: www.badguys.org
(ISC)2 (International Information Systems Security Certification Consortium, Inc.): www.isc2.org
CISSP certification experience requirement changes: www.isc2.org/cgi-bin/content.cgi?category=84
GIAC (Global Information Assurance Certification): www.giac.org
McAfee training: https://mcafee.edu.netexam.com/
Sophos training: www.sophos.com/companyinfo/events/
Symantec training: www.symantec.com/en/uk/enterprise/training/virtual_academy/index.jsp
Team Anti-Virus Education Website: www.teamanti-virus.org/edu.html

Stalkers on Your Desktop

"Dr. Solomon's Virus Encyclopedia": Solomon, Gryaznov (S&S International, 1995)
"Survivor's Guide to Computer Viruses": anthology of VB articles (Virus Bulletin, 1993)
"Robert Slade's Guide to Computer Viruses" 2nd Edition (Springer, 1996)
"Viruses Revealed": David Harley, Robert Slade, Urs Gattiker (Osborne, 2001)

Peter Szor's "The Art of Computer Virus Research and Defense" (2005, Symantec/ Addison-Wesley)

The CARO web site is, at time of writing, down for maintenance, but Vesselin Bontchev's paper explains the CARO naming scheme very clearly: www.people.frisk-software.com/ ~bontchev/papers/naming.html.

"The Root of All Evil – Rootkits Revealed": David Harley and Andrew Lee (http://www.eset.com/download/whitepapers/Whitepaper-Rootkit_Root_Of_All_Evil.pdf)

Chey Cobb, Stephen Cobb, M.E. Kabay: "Penetrating Computer Systems and Networks". In "Computer Security Handbook 4th Edition", ed. Bosworth & Kabay (Wiley, 2002).

"Trojans" David Harley. In "Maximum Security" (SAMS, 2002).

"Windows Rootkits of 2005" Parts 1–3. James Butler and Sherri Sparks. http://www.securityfocus.com/infocus/

"Sony, Rootkits and Digital Rights Management Gone Too Far". Mark Russinovich, and related items: http://www.sysinternals.com/blog/2005/10/sony-rootkits-and-digital-rights.html; http://www.sysinternals.com/blog/2005/11/more-on-sony-dangerousdecloaking.html; http://cp.sonybmg.com/xcp/english/updates.html; http://cp.sonybmg.com/xcp/english/form14.html

"Hide 'n Seek Revisited – Full Stealth is Back". Kimmo Kasslin, Mika Stahlberg, Samuli Larvala and Antti Tikkanen. In Proceedings of the 15th Virus Bulletin International Conference, 2005.

"Subverting Vista Kernel for Fun and Profit" Joanna Rutkowska (http://theinvisiblethings.blogspot.com/)

"SubVirt: Implementing malware with virtual machines". Samuel T. King, Peter M. Chen, Yi-Min Wang, Chad Verbowski, Helen J. Wang, Jacob R. Lorch (http://www.eecs.umich.edu/virtual/papers/king06.pdf)

Martin Overton's Rootkits paper: http://cluestick.info/papers/VB2006-Rootkits-1.0.2.pdf

Phishing

BBC Informational sites: http://www.bbc.co.uk/crime/support/cardfraud.shtml, http://www.bbc.co.uk/crime/prevention/cardtheft.shtml

APACS (Association for Payment Clearing Services) sites: www.apacs.org.uk/, www.banksafeonline.org.uk/, http://www.cardwatch.org.uk/

Banking Code Standards Board: www.bankingcode.org.uk/, www.bankingcode.org.uk/ pdfdocs/bankcode.pdf, http://www.bankingcode.org.uk/pdfdocs/busbankcode.pdf

The Anti-Phishing Working Group:

- www.antiphishing.org/consumer_recs.html
- www.antiphishing.org/consumer_recs2.html
- www.antiphishing.org/resources.html

- www.antiphishing.org/report_phishing.html
- www.antiphishing.org/phishing_archive.html
- www.microsoft.com/security/protect
- www.staysafeonline.info

Martin Overton's Electronic Ephemera FAQ, papers etc.: http://cluestick.info/hoax/
"Social Engineering FAQ", David Harley (www.smallblue-greenworld.co.uk)
"Dealing with Internet Hoaxes/Alerts", David Harley (www.smallblue-greenworld.co.uk)
"E-Mail Abuse, Internet Chain Letters, Hoaxes and Spam", David Harley (www.smallblue-greenworld.co.uk)
"The Email of the Species", David Harley (Virus Bulletin Conference Proceedings, 2000)
"Man, Myth Memetics and Malware", David Harley and Urs Gattiker (EICAR Conference Proceedings, 2002)
"Spam, Malware Deception and Impersonation: a Taxonomical Approach", David Harley (EICAR Conference Proceedings, 2004)
"ISSE 2004 – Securing Electronic Business Processes", David Harley: chapter on "Massmailers: New Threats Need Novel Anti-Virus Measures" (Vieweg 2004).
"Fact, Fiction, and Managed Anti-Malware Services", David Harley (Virus Bulletin Conference Proceedings, 2003)
"Spam Mitigation Techniques": NISCC Technical Note (http://www.cpni.gov.uk/Products/technicalnotes/default.aspx?id=tn-20040227-00102.xml)

A Tangled Web

Google Accelerator: http://www.windowsdevcenter.com/pub/a/windows/2005/05/24/google_accelerator.html).
Google PageRank:: "The PageRank Citation Ranking: Bringing Order to the Web" by Larry Page, Sergey Brin, R. Motwani, and T. Winograd, at http://citeseer.ist.psu.edu/page98pagerank.html. See also www.google.com/technology/; www.google.com/corporate/tech.html; http://www.google.com/support/webmasters/
Robots: www.robotstxt.org/wc/exclusion-admin.html; http://www.robotstxt.org/wc/exclusion-user.html; www.w3.org/TR/html4/appendix/notes.html#h-B.4.1.1
Defacements:
http://vil.nai.com/vil/content/v_100488.htm; http://www.lurhq.com/berbew.html;
http://www.microsoft.com/security/incident/download_ject.mspx;
http://cnscenter.future.co.kr/resource/security/application/deface.pdf
Index hijacking: "Manipulating the Internet" (http://download.nai.com/products/mcafee-avert/WhitePapers/IMuttik_VB2005_Manipulating_The_Internet.pdf)

"Scanning on the Wire", by I. Muttik (in "Proceedings of the International Virus Bulletin 2006 conference", Montreal, Canada. 10–13 October 2006, pp.120–125.)
Muttik I. "Comparing the comparatives" http://www.mcafee.com/common/media/vil/pdf/imuttik_VB_conf_2001.pdf.)
Martin Overton in his 2005 Virus Bulletin conference paper.

Big Bad Bots

"Botnets: the Killer Web App", Craig Schiller & Jim Binkley (Syngress, 2007)
"The World of Botnets", Gadi Evron and Dr. Alan Solomon (Virus Bulletin, September 2006),
"A Taxonomy of Botnets", David Dagon et al (www.math.tulane.edu/~tcsem/botnets/ndss_botax.pdf)
Martin Overton's 2005 Virus Bulletin conference paper: "Bots and Botnets: Risk, Issues and Prevention" http://cluestick.info/papers/VB2005-Bots_and_Botnets-1.0.2.pdf
"Anti-malware Tools: Intrusion Detection Systems", Martin Overton. http://cluestick.info/papers/EICAR2005-IDS-Malware-v.1.0.2.pdf
"The Network is the Infection: Botnet Detection and Response," www.caida.org/workshops/dns-oarc/200507/slides/oarc0507-Dagon.pdf
"Covert Zombie Ops", John Aycock (Virus Bulletin, May 2007)
"Defeating IRC Bots on the Internal Network", Vinoo Thomas, Nitin Jyoti; "Web Server Botnets and Hosting Farms as Attack Platforms", Gadi Evron, Kfir Damari, Noam Rathaus (both in Virus Bulletin, February 2007)
www.honeynet.org/papers/bots/
Proxy abuse: http://www.lurhq.com/proxies.html
John Canavan tells us in "The Evolution of Malicious IRC Bots," a white paper for Symantec (www.symantec.com/avcenter/reference/the.evolution.of.malicious.irc.bots.pdf),
Shadowserver (www.shadowserver.org)
NANOG (North American Network Operators Group): www.nanog.org
"Network telescope" (http://www.caida.org)
Cymru Darknet project (http://www.cymru.com/Darknet/)
IMS (Internet Motion Sensor): http://ims.eecs.umich.edu/
"Know Your Enemy" papers: http://project.honeynet.org/papers/kye.html.

Crème de la CyberCrime

A quick way to revisit alt.comp.virus: http://groups.google.com/group/alt.comp.virus/topics
iDefense: http://labs.idefense.com/
Websense: www.websense.com/global/en/
Sarah Gordon's papers: www.badguys.org

"Sendmail: Theory and Practice 2nd Edition", Paul A. Vixie and Frederick M. Avolio (Digital Press, 2001)
Anti-Phishing Working Group resources page: www.antiphishing.org/resources.html
"Refloating the Titanic – Dealing with Social Engineering Attacks", David Harley (EICAR conference proceedings, 1998)

Defense in Depth

"The Handbook of Corporate Malware Protection", Ken Bechtel (https://www.regnow.com/softsell/nph-softsell.cgi?items=9678-2)
Tripwire: www.tripwire.com/company/index.cfm
The Jericho forum: www.opengroup.org/projects/jericho/index.tpl
Snort: www.sourcefire.com; http://momusings.co.uk/publications.aspx; www.bleedingsnort.com
"Heurist Analysis: Detecting Unknown Viruses", David Harley and Andrew Lee (http://www.eset.com/download/whitepapers/HeurAnalysis(Mar2007)Online.pdf)
Fred Cohen: "A Short Course on Computer Viruses" Second Edition (Wiley, 1994)

Perilous Outsorcery

George Sadowsky, James X. Dempsey, Barbara J. Mack,lan Schwartz: Information Technology Security Handbook, 2003 The International Bank for Reconstruction and Development, The World Bank - www.infodev-security.net/handbook/
OODA loop and "hamster wheel": http://en.wikipedia.org/wiki/OODA_Loop
NIST Special Publication 800–35 "Guide to IT Security Services"
IT Infrastructure Library (ITIL): www.itil.co.uk; www.itsmf.com

Education in Education

"Coming of Age: an introduction to the new world wide web" 2nd Edition: http://web2booklet.blogspot.com/2006/10/draft-table-of-contents-16.html; www.terry-freedman.org.uk/db/web2/
"Vandalism in Cyberspace: Understanding and Combating Malicious Software": http://tscp.open.ac.uk
BECTA: http://www.becta.gov.uk (home page)
"The Real Reason for the Decline of the Macro Virus", Vesselin Bontchev (http://www.virusbtn.com/pdf/magazine/2006/200601.pdf)
http://schools.becta.org.uk/index.php?section=is&catcode=ss_to_es_pp_aup_03
http://www.safety.ngfl.gov.uk/ (Now absorbed into BECTA schools website)
CERT: Computer Emergency Response Team (www.cert.org).

CSIRT: Computer Security Incident Response Team (www.cert.org/csirts/)
JANET-CERT (www.ja.net/cert/)
WARPs: www.warp.gov.uk
"Teach Your Children Well: ICT Security and the Younger Generation": David Harley, Eddy Willems and Judith Harley (Virus Bulletin Conference Proceedings, 2005)
Internet Watch Foundation (specific information on and reporting of child abuse etc.): http://www.iwf.org.uk/
INHOPE (http://www.inhope.org/en/index.html) represents and links international child abuse hotlines.
The Association of Chief Police Officers web site includes documentation relating to child-related problems: http://www.acpo.police.uk/policies.asp
General child-safety resources:

- http://www.bbc.co.uk/webwise/course/safety/menu.shtml (online safety guides and quizzes)
- http://www.besafeonline.org/English/safer_use_of_services_on_the_internet.htm
- http://www.getsafeonline.org/
- http://www.safekids.com/
- http://www.staysafe.org/
- http://www.microsoft.com/athome/security/default.mspx
- http://www.thamesvalley.police.uk/chatsafe/young.htm#tips (Safety guide for internet chat rooms)
- http://www.chatdanger.com/ (Guide to safe use of chat rooms, IM, on-line games etc.)
- http://www.childnet-int.org/publications/ (Childnet International resources)
- http://www.fkbko.co.uk/EN.php?lang=EN&&subject=0&&id=0&&level=0 (Cyberspace Research Unit)
- http://www.gridclub.com/teachers/t_internet_safety.html (Part of the Internet Proficiency Scheme developed by BECTA, DfES, and QCA)
- http://www.internetsuperheroes.org/ (comic-book oriented safety site)
- http://www.websafecrackerz.com/default.aspx (quizzes)
- http://uk.docs.yahoo.com/parents_guide/ (safe family surfing guide)

Cyber bullying:

- http://www.antibullying.net/cyberbullying1.htm
- http://www.cyberbullying.ca/pdf/feature_dec2005.pdf

- http://www.bullying.org/
- http://www.kidscape.org.uk/childrenteens/cyberbullying.shtml
- http://www.wiredsafety.org/gb/law/spam/uk_cyberbullying.html
- http://www.bullyonline.org/related/cyber.htm virus book mini-reviews

Security book mini-reviews

- http://www.mysecurityplan.net/score/
- http://www.mysecurityplan.com/litescore/
- http://www.mysecurityplan.com/quiz/
- http://www.mysecurityplan.com/safetyscore/

DIY Malware Analysis

VNC: http://ultravnc.sourceforge.net/; www.realvnc.com/; www.realvnc.com/products/enterprise/
SysInternals: http://www.sysinternals.com; www.microsoft.com/technet/sysinternals/Security/
Sample identification/submission sites: www.virustotal.com; http://virusscan.jotti.org; http://scanner.virus.org
Sandbox sites: www.cwsandbox.org/; http://research.sunbelt-software.com/submit.aspx; http://sandbox.norman.no/live.html
VMware: www.vmware.com
Virtual PC: www.microsoft.com/windows/products/winfamily/virtualpc/default.mspx
"Wireshark and Ethereal: Network Protocol Analyzer Toolkit", by Angela Orebaugh et al. (Syngress)
Wireshark: www.wireshark.org; www.wireshark.org/docs/
FreeSpace Internet Security (http://www.freespaceinternetsecurity.com),
nmap: www.insecure.org
Superscan: http://www.foundstone.com
Nessus: www.nessus.org
WinDbg: www.microsoft.com/whdc/devtools/debugging/default.mspx
Ollydbg: http://www.ollydbg.de/
IDA Pro: www.datarescue.com

Antivirus Evaluation and Testing

EICAR test file: www.eicar.org/anti_virus_test_file.htm.
WildList: www.wildlist.org
Anti-Spyware Coalition: www.antispywarecoalition.org
Virus Bulletin VB100: www.virusbtn.com
ICSA Labs: www.icsalabs.com
WestCoast Labs CheckMark: http://westcoastlabs.org; www.westcoastlabs.org/checkmarkcertification.asp
AV-Test.org: www.av-test.org
AV Comparatives: www.av-comparatives.org
"Real World Anti-Virus Product Reviews And Evaluations – The Current State Of Affairs", Richard Ford and Sarah Gordon (http://csrc.nist.gov/nissc/1996/papers/NISSC96/paper019/final.PDF)
"A Reader's Guide to Reviews", Dr. Alan Solomon, first published in Virus News International credited to Sarah Tanner (http://www.softpanorama.org/Malware/Reprints/virus_reviews.html)
Virus Research Unit, University of Tampere - http://www.uta.fi/laitokset/virus/
Virus Test Centre University of Hamburg - http://agn-www.informatik.uni-hamburg.de/vtc/naveng.htm

Additional Resources

Books

"Handbook of Information Security", Ed. Bidgoli (Wiley)
"The Handbook of Computer Networks", Ed. Bidgoli (Wiley)
"Computer Security Handbook", Eds. Bosworth, Kabay (Wiley)
"Practical Unix and Internet Security", Spafford, Garfinkel, Schwartz (O'Reilly)
"Malicious Mobile Code: Virus Protection for Windows", Roger Grimes (O'Reilly)
"Combating Spyware in the Enterprise", Baskin et al. (Syngress)
"Phishing Exposed", Lance James (Syngress)
"How to Cheat at Managing Information Security", Mark Osborne (Syngress)
"The Manager's Handbook for Corporate Security – Establishing and Managing a Successful Assets Protection Program", Gerald L. Kovacich, Edward Halibozek (Elsevier)
"Defense and Detection Strategies Against Internet Worms", Jose Nazario (Artech)
"Malware: Fighting Malicious Code", Ed Skoudis (Prentice-Hall)
"Governance Guidebook", Fred Cohen (ASP)
"The CISO Handbook", Mike Gentile, Ron Collette, Tom August (Auerback)
"Information Security Management Handbook", Harold F. Tipton/Micki Krause (Auerbach)

"IT Security Project Management", Susan Snedaker (Syngress)
"Bigelow's Virus Troubleshooting Pocket Reference", Ken Dunham (Osborne)
"Essential Computer Security", Tony Bradley (Syngress)
"Combating Spyware in the Enterprise", Baskin et al (Syngress)

Additional Resources

BSD security: www.freebsd.org/security/
Center for Education and Research in Information Assurance and Security:

- www.cerias.purdue.edu/
- www.cerias.purdue.edu/tools_and_resources/hotlist/

CERTS etc.:

- AusCERT: www.auscert.org.au/
- FrSirt: www.frsirt.com/english/mailing.php
- National Infrastructure Security Co-ordination Centre (NISCC), is now CPNI (Centre for the Protection of National Infrastructure) (www.cpni.gov.uk/default.aspx)
    - www.cpni.gov.uk/Products/advisories.aspx
    - www.cpni.gov.uk/Products/briefings.aspx
    - www.cpni.gov.uk/Products/technical.aspx
    - www.cpni.gov.uk/Products/guidelines.aspx
    - www.cpni.gov.uk/Products/generalProtective.aspx
    - www.warp.gov.uk/
    - www.itsafe.gov.uk
- US CERT et al:
    - www.us-cert.gov/nav/t01/ – tech users
    - www.us-cert.gov/nav//nt01/ – non-tech users
    - www.us-cert.gov/federal/ – US Govt. users
    - www.us-cert.gov/cas/signup.html – mailing lists
    - www.us-cert.gov/reading_room/ – publications
    - www.us-cert.gov/resources.html
    - INFOCON Mailing List: http://www.iwar.org.uk
    - www.cert.org/ CERT/CC

- www.cert.org/nav/index.html
- www.first.org/ Forum of Incident Response and Security Teams
- www.cyberpartnership.org/ National Cyber Security Partnership

CME naming scheme – http://cme.mitre.org/
Common Vulnerabilities and Exposures (CVE®) – http://www.cve.mitre.org/
Common Criteria – http://www.commoncriteriaportal.org/public/consumer/
Computer Associates Virus Information Center: http://ca.com/virusinfo
F-Secure: www.f-secure.com/virus-info/; www.f-secure.com/weblog/

Linux:

- www.linuxsecurity.com/
- www.linuxsecurity.com/content/blogcategory/0/76/ – advisories
- www.linuxsecurity.com/content/view/101892/155/ – resources
- www.linuxsecurity.com/content/view/86098/68/ – newsletters
- http://packetstorm.linuxsecurity.com/
- http://packetstorm.linuxsecurity.com/assess/
- http://packetstorm.linuxsecurity.com/alladvisories/
- http://packetstorm.linuxsecurity.com/papers/
- http://www2.packetstormsecurity.org/cgi-bin/cbmc/forums.cgi

Macintosh:

- www.securemac.com/
- www.apple.com/support/security/
- www.apple.com/macosx/features/security/
- http://docs.info.apple.com/article.html?artnum=61798 [updates]
- http://lists.apple.com/mailman/listinfo/security-announce

Network tools:

- www.samspade.org/
- www.samspade.org/d/tools.html
- www.samspade.org/t

- www.ripe.net
- www.zoneedit.com/lookup.html?ad=goto&kw=nslookup
- www.dnsstuff.com/
- www.honeynet.org/
- www.ripe.net/tools/
- www.whois.net/
- www.cymru.com/

NIST (National Institute of Standards & Technology): http://csrc.nist.gov/publications/fips/index.html

SANS:

- SANS http://www.sans.org/newsletters/
- www.sans.org/rr/ – reading room
- http://isc.sans.org/ – Internet Storm Center

Security Focus Newsletters

- www.securityfocus.com/newsletters (SFNews, Microsoft News, Linux News)
- SecurityFocus virus papers – http://www.securityfocus.com/virus
- www.securityfocus.com/archive/1 – BugTraq archive
- www.securityfocus.com/vulnerabilities
- www.securityfocus.com/tools
- www.securityfocus.com/columnists
- www.securityfocus.com/archive – Mailing Lists

Peter Szor's papers – http://www.peterszor.com/
F-Secure: Security Information Center – http://www.europe.f-secure.com/virus-info/
McAfee AVERT Virus Information Library – http://vil.nai.com/vil/default.asp
Sophos virus analyses – http://www.sophos.com/virusinfo/analyses/
Sophos white papers – http://www.sophos.com/virusinfo/whitepapers/
Symantec Security Response – http://securityresponse.symantec.com/avcenter/vinfodb.html
Trend Micro Virus Encyclopedia – http://www.antivirus.com/vinfo/virusencyclo/

Appendix B

Glossary

Introduction

Clearly, I can't include a major glossary resource here to compare to Robert Slade's "Dictionary of Information Security," which I recommend with enthusiasm as a great general security resource.

In fact, I can't do justice to the full range of terms and concepts addressed in this already rather large book. So while I've largely concentrated on malware-related topics, which is, after all, the book's main focus, I – indeed, we – have become aware that there's a lack of an up-to-date, specialist glossary resource in the malware arena, which this short glossary cannot address fully. However, we'll be addressing that gap properly quite soon. See www.smallblue-greenworld.co.uk/pages/avienguide.html or the AVIEN home page for more news about that project.

David Harley, Lead Author/Technical Editor

Adware Programs that initiate or display some sort of advertisement, or divert a web browser to a commercial site. If installed without the knowledge or permission of the user, or intentionally difficult to uninstall, may be considered to be a form of Trojan.
A-I-C See CIA.
Almost Exact Identification Recognition of a virus where the identification is good enough to ensure that an attempt to remove the virus will not result in damage to the host object by the use of an inappropriate disinfection method, but without uniquely identifying every section of the non-modifiable parts of the virus body (which is a brief definition of "actual" Exact Identification.)
AM Infrequently used abbreviation for antimalware (or anti-malware).
API Hooking When a request for system information is generated, it calls an API function which forwards the request to the kernel, and returns the requested information back to the requesting application via the same "execution path". Hooking, especially in the context of rootkits, describes the process of hiding data by diverting the execution path through a filter, so that the information returned is incomplete or modified.
AS Antispam (or anti-spam)
AUP (Acceptable Use Policy) Guidelines for users on what constitutes acceptable usage of a system, network services *etc*.
AV Antivirus (or Anti-Virus)
Availability One of the three basic principles of information protection/security. Data should be available whenever it's needed. See CIA.
AVAR: Association of Anti Virus Asia Researchers (www.aavar.org), an organization for Anti-Virus Researchers with interests in or located in Asia.
AVIEN Anti-Virus Information Exchange Network (www.avien.org) an organization of anti-virus support personnel. This organization was established for the free exchange of ideas, with out vendor influence.

AVIEWS Anti-Virus Information and Early Warning System: all members of AVIEN are also members of AVIEWS, but AVIEWS also encourages members from the security industry, small businesses, and individuals.
Backdoor Remote control software, normally installed unknown to the user, to allow the malicious user to exploit the compromised machine.
Backdoor-Ali (AKA ierk or slanret A backdoor Trojan often described as a rootkit
Backscatter Responses) from virus scanners, mail servers and so on to email with forged sender addresses (spam and mass mailers, for example) that don't take into account the possibility of forged addresses, and so misdirect responses to the apparent sender.
Bcc (Blind Carbon Copy) Email receiver field: recipients listed in the Bcc field are not shown in the copies sent to the primary recipient(s) in the To field, or to the secondary recipient(s) in the Cc (Carbon Copy) field. Sometimes used by spammers and scammers, for instance, to conceal the fact that an apparently personal message has been sent to many people.
Bot herder, botmeister, bot master Someone who controls a botnet for diverse criminal or malicious purposes such as spam distribution, malware distribution, and DDoS orchestration. Strictly, bot herding is the practice of migrating ranges of bots between C&C servers.
Botnet Network of bot-compromised systems under the control of a bot master or herder.
C&C Server Command and Control server. Used to control, update and receive data from component systems on a botnet.
CARO Computer Antivirus Researcher Organization
CDO Honorific traditionally bestowed on the AVIEN administrator. If you want to know what it stands for, you'll have to join.
Chain Letter Originally a letter promising good luck if you pass it on to others or misfortune if you don't, thus breaking the chain. Hoax emails are usually a special case of electronic chain mail: the recipient is urged to pass on a warning, often about a "virus," to everyone he knows.
Choke Point Servers or devices through which all traffic or specific types of traffic must pass. Examples include proxy servers and mail gateways. Choke points are a good place to apply security filters and countermeasures to ensure that
CIA Mnemonic for Confidentiality, Integrity and Availability, the classic security "triad." Some sources, notably (ISC)2 prefer to refer to it as A-I-C, perhaps because it puts the (arguably) most "important" attribute, availability, ahead of the other two. Note that while this is the most used model, it's by no means the only one. Donn Parker has advocated a "hexad" consisting of availability and utility, integrity and authenticity, confidentiality and possession of information. Some commentators – well, me – have also advocate promoting accountability to this top level. So it's really horses for courses. But some security certification tests will always ask you about the classic triad.
Confidentiality One of the three basic principles of information protection/security. Data should be accessible only to people who are entitled to access it. See CIA.

Corruption In the context of malware, especially viruses, a maliciously intended program that behaves unpredictably or is unable to function because of an unintended modification or truncation.

Cybersquatting Registration of a domain with the intention of benefiting in some way from a perceived but deceptive or malicious association between the registrant and a legitimate site or business.

Daemon or demon In UNIX, a server or system process that runs in the background, as opposed to being executed by the user. The end user executes a client application which communicates with the daemon. For example, fingerd is a process that runs on (some) networks waiting for requests from the finger client on remote PCs and responding appropriately.

DDoS Distributed Denial of Service attack: a DoS attack amplified by delivery through a network of compromised systems, especially a botnet.

Deepdoor Proof of Concept rootkit by Joanne Rutkowska

Destructive Trojan A Trojan (see Trojan) that has a deliberately destructive payload. Sometimes described as an arf-arf after an early Trojan that displayed the string "Gotcha. Arf, arf" while erasing the victim's hard drive. (That snippet comes from Robert Slade, by the way.)

DoS Denial of Service attack: an attempt to prevent a computer or network from functioning normally. Often associated with extortion, where the criminal threatens to prevent systems from operating so that the victim cannot carry out normal business operations.

Dropper A program that installs a virus or other malware.

Dumpster Diving (Scavenging, Trashing) Searching waste for useful information (especially discarded paperwork, electronic media *etc.*).

EICAR European Institute for Computer Anti-Virus Research. One of the oldest anti-malware research organizations, formed in Europe. More information can be found at http://www.eicar.org

EICAR test file A program file which can be typed in using an ASCII file: most anti-virus programs recognize as a test program, and respond in a very similar way to that in which they respond to viruses. However, the EICAR file is not a virus and has no malicious payload: it simply displays a string identifying itself.

Electronic Garbage, Leftover Data, passwords etc. left in memory or on disk where a knowledgeable intruder might be able to access them.

Ethics In the context of the malware research and security communities, we address the topic here of what is often called 'normative ethics', relating to a code of moral values based upon accepted social norms, but applied specifically to the use of computers in the security context.

EULA End User License Agreement. The express terms of use for software that binds users (both individual and corporate) to the vendor of that software: a contract between the vendor and the customer.

Exact Identification Detection where every section of the non-modifiable parts of the virus body are uniquely identified. In general, it is too resource-intensive to do this for every virus/malicious program in a scanner's definitions database, but scanners are usually pretty good at taking safe shortcuts.

False Negative Failure by a scanner or other security software to recognize the presence of malicious code.

False Positive Misidentification of a file or object as being malicious code. An object identified as "suspicious" or blocked because it's a member of a proscribed class (e.g. .EXE mail attachments) is not usually seen as a false positive (FP).

finger A utility which allows a network or internetwork user to find out (1) account names on a remote server (2) whether the holder of a known account name is logged onto the system. This can enable someone who doesn't have an account on a given system to match an account name to a real person, and determine various data including the last time they logged in. *fingerd* is more often than not disabled nowadays, for security reasons: it's included here because it's used as an example of a daemon.

Garbage files Probably fairly self-descriptive. A garbage file, when found in a virus collection, is something that isn't a virus or anything like.

Generic Antonym of virus- or threat-specific, used to describe security programs that don't recognize specific threats but defend against them using a method that will block a whole class or whole classes of threat. A generic signature, definition or driver is a special case of this scenario: a whole set of variants are detected and processed by a single definition rather than by individual definitions for each variant.

Generic Driver or Generic Detection In anti-virus, a virus signature or definition which has been generalized so as to detect a family of viruses or variants, rather than detecting a single unique variant. This is almost the same as the distinction between exact and almost exact identification, since almost exact identification is usually associated with generic disinfection. A good source of further information is Peter Szor's "The Art of Computer Virus Research and Defense".

Generic Driver, Generic Signature or Generic Detection In anti-virus, a convenient but not universally used term to describe a virus signature or definition which has been generalized so as to detect a family of viruses or variants, rather than detecting a single unique variant. This is similar to but not the same as the distinction between exact and almost exact identification, since almost exact identification is usually associated with generic disinfection. In-depth discussion of these concepts is beyond the scope of this paper. A good source of further information is Peter Szor's "The Art of Computer Virus Research and Defense".

Germ A "generation 0" virus: usually the original code without a parasitized object, or a "forced" infection that couldn't happen "naturally."

Governance (Information Governance) An information management framework, particularly with regard to policy and compliance issues.

Grayware, Greyware A somewhat fuzzy class of software that may include adware, spyware, joke programs, remote access programs etc. Some form of concealed functionality is usually presumed. May be applied to PUPs and PUAs (q.v.)

Grayware, Greyware A somewhat fuzzy class of software that may include adware, spyware, joke programs, remote access programs etc. Tends to include a presumption of concealed functionality.

Hacker Originally, someone with a strong (often experimental) interest and knowledge of computers and software. Commonly used as if synonymous with 'cracker' (a term coined in an attempt to distance computer intrusion from 'legitimate' hacking.)

Hashbuster Some spam filters use a database of "hashes" to identify spam messages: an MD5 hash or message digest, for instance, represents a string or message as a sequence of 32 hexadecimal digits. Spammers often include random text in the subject or body of a message, so as to generate random changes from one spam iteration to another, thus defeating filters that rely on checksums or hashes. Similar techniques may be applied to image spam.

Heterogeneous virus transmission A scenario where a virus travels reaches a potentially vulnerable or infectable system by way of a system which isn't in itself vulnerable to that particular threat. For instance, a Windows virus retrieved from a Mac or Unix server by a Windows PC. The concept is valid for other types of malware. The term was probably originally coined by Peter Radatti in the Unix context, and borrowed by David Harley when writing about Mac issues.

Heuristic detection/scanning Scanning for code attributes that suggest malicious or replicative behavior.

Heuristics A blanket term applied to a range of techniques for proactively detecting currently unknown malware and variants.

Identification The ability of an antivirus program (usually a scanner to detect the malware and recognize it by a unique name, allowing a third party to understand which particular malware it is without possessing a sample. (This is more difficult than it sounds!)

Integrity One of the three basic principles of information protection/security. Data should be protected from accidental or deliberate corruption, deletion *etc*. See CIA.

Intended A virus that doesn't work for one reason or another.

Joke programs A program which may have malicious intent (e.g. to humiliate or frighten the recipient – for instance, one which pretends to have formatted the hard disk), but doesn't have a permanent damaging effect on the system. An example of a less alarming joke or prank is one which opens a window which moves around the screen so that you can't close it.

Keyloggers A program that monitors keystrokes, especially appertaining to authentication (passwords).
Known Virus Scanning, Virus-Specific Scanning Scanning for a specific, known virus.
L33t, 3l33t Elite in hackerspeak (look at me, I'm ultracool!)
Latent Malware Malware that hasn't been executed/activated.
Loop See Recursion
Lottery scam Advance fee fraud that works by telling the victim they've won a huge amount of money, but have to pay some mon ey upfront before they can receive it
Lrk A well-known, well-established Linux rootkit. Described at http://staff.washington.edu/dittrich/misc/faqs/lrk4.faq
Malware A collective term (contraction of MALicious software) including viruses, worms, Trojan horses, backdoors, and so on. Sometimes used to differentiate non-replicative malicious programs (Trojans, spyware etc.) from replicative code (viruses and worms.)
Mule In organized crime, the term has a number of meanings, including a courier for drugs, money etc. In phishing and related crimes, usually refers to someone who is involved (sometimes unwittingly) with money laundering by receiving and forwarding fraudulently acquired funds, goods or services.
Multi-Level Marketing (MLM) A legitimate business model which may have a hierarchical/ pyramidal structure, but aims to build up a legitimate sales model rather than a pyramid scheme.
Opener (AKA Renepo): Macintosh malware sometimes described as a rootkit. OSXrk is another example of a Mac more-or-less rootkit
OS X The current Macintosh operating system, based on BSD UNIX but with an interface that continues the long-running Mac user-friendly "look" as well as access to a more conventional UNIX command line.
Out of Band Communications via a channel outside a communications channel already established.
Password stealers A program that steals passwords. No, honestly. ☺
Pennystox, Penny Stocks, Small Caps Businesses with low initial stock values characteristically victimized by criminal groups executing pump and dump schemes.
Phreaking Using electronics to make free phone calls or make them at the expense of others, access phone company computers *etc.*
PoC Proof of Concept: in this context, a potentially malicious program intended to prove a point, rather than to be disseminated into the wild in its present form.
Privacy See confidentiality.
Programmatic Threats Malicious code (malware) such as viruses and Trojan Horses. Also sometimes referred to as malicious logic, malware, or malcode.

Proxy Server A proxy server mediates between an organization and the Internet. It improves performance by fulfilling a request directly rather than passing it to a remote site, if the necessary information is available. Proxy server also serve to block unauthorized activity – outgoing or incoming.

PUA, PUP Possibly Unwanted Application or Possibly Unwanted Program. A term used by antimalware vendors for a class of software that may or may not be legitimate and authorized by the system owner. Detecting such software may be an option rather than a default

Pump and Dump A form of stock fraud in which the value of stock is artificially inflated so that dishonest speculators can make a profit by selling off when the price is high. This works well for the scammer, but not, usually, for the company, or for the scam victims whose contribution to the raising of stock value goes into the pocket of the "dumper."

Pyramid Scheme Scam in which new entrants pay for the chance to move towards the top of a pyramid and take a cut out of payments towards later entrants. The major returns on the scheme go to the initiator and, maybe, some of the earliest entrants. Participation is illegal in some jurisdictions.

Self launching Applied to worms (or, less frequently, viruses), means that the malware doesn't need the victim's intervention in order to activate.

Retrospective testing Testing the heuristic capabilities of a scanner by "freezing" it without update for a set period, in order to test its detection performance against viruses

Recursion See loop.

Rootkit Program or suite of programs intended to escalate privilege ("get root") and/or maintain covert access and control of a system, i.e. without the knowledge of its owner.

Samurai Hackers for Hire': hackers or crackers who sell their skills.

Scan String, Search String A sequence of bytes known to occur in specific malware but not (hopefully) found in legitimate/uncompromised programs. (If the string is found in a legitimate program, this is a False Positive (FP.)

Shadow PoC (Proof of Concept) rootkit by James Butler and Sherri Sparks, partly based on 's FU rootkit.

Shell The command processor which translates commands entered at a terminal or keyboard into the system's instruction set. The term may also describe chain of processes launched by the command processor (spawning a shell or shelling out.) A privileged process may be referred to as a root shell. Some programs run as root (or an equivalent, such as "administrator") without extending root privileges to the user, but vulnerabilities such as overflows can be used to escalate privileges ("get root.").

Shoulder-Surfing Looking over someone's shoulder to ascertain their password or PIN.

Signature scanning In AV, searching for the presence of a virus by checking for a static byte sequence. Even basic AV scanning has long used more complex and effective techniques, using algorithmic approaches, and wildcards for example. The term is still routinely used in intrusion detection. It is sometimes used to denote a checksum or hash as a database entry used to verify a *legitimate* program, using an application such as tripwire for integrity management.

Signature: Confusing synonym for scan string: may be applied to a static search string, but best avoided altogether, particularly as it often misleads people into thinking there is a single byte sequence used by all virus scanners to recognize each virus or variant.

Social Engineering Term applied to a wide range of techniques for causing a desired change in behavior or gaining some advantage by psychological manipulation of an individual or group. As used in security it almost invariably involves some form of deception, malice or fraud.

Spam Unsolicited electronic mail, especially commercial/junk e-mail. Also applied to the practice of posting large numbers of identical messages to multiple newsgroups. Can include virus backscatter, hoaxes, phishes and 419s and so on, depending on the precise definition in use.

Spyware Generic term for a range of surreptitious malware such as keyloggers, Remote Access Trojans, and backdoor Trojans, especially those that allow remote surveillance of passwords and other sensitive data. When used for frankly criminal activities may also be referred to as crimeware.

Stealth Adjective: often used in security as an alternative to "stealthy", by analogy to "stealth aircraft" and other military terminology relating to concealment strategies.

Stealth:Adjective: often used in security as an alternative to "stealthy", by analogy to "stealth aircraft" and other military terminology relating to concealment strategies.

Stealthkit Stealthware that consists of a set of tools rather than a single program. May include rootkits.

Stealthware Software (usually malware) that uses stealth techniques to conceal itself: the definition may include rootkits, especially if (1) it has several components (2) it attempts to escalate privilege as well as maintain covert access.

Stealthware Software (usually malware) that uses stealth techniques to conceal itself

Trojan Horse A program that disguises its true function, especially a malicious function such as installing spyware or destroying data without the knowledge or permission of its owner.

Tsunami scams A range of charity scams and hoaxes allegedly raising funds for victims of the 2004 tsunami. Examples include many 419s and phishing mails.

Typhoid Mary An individual who can host a disease, without showing the symptoms (a carrier). The computer equivalent hosts infected files, but the host machine itself remains uninfected: see http://history1900s.about.com/library/weekly/aa062900a.htm.

Virus There is no universally agreed definition. A common definition is "a program that modifies other programs to contain a possibly altered version of itself," or similar. In other words, a virus requires a host to attach itself to, either by modifying the host, or inserting itself in some way into the "chain of command."

Virus generator program A program that generates viruses, rather than being a virus itself.

Vishing The use of VoIP as a vector for phishing attacks: for example, by directing the victim to a spoofed phone number to verify sensitive data.

VoIP Voice over IP: telephony and related services that work over the Internet rather than over older messaging media and protocols.

WeaponX Kernel based rootkit for OS X

WildList A list of viruses known to be "In the Wild", compiled each month by volunteers, using reports from antivirus researchers world-wide. The test set that underpins it is, at least in part, the basis of most competent AV detection tests.

WildList Organization The organization that compiles the WildList: see http://www.wildlist.org.

Worm Worms replicate, but don't attach themselves to a host program. In the words of Simon Wildlake, as "viruses infect, worms infest."

Zombie A compromised system controlled remotely and maliciously, especially as a component of a botnet.

Index