Computer Viruses and Malware

Advances in Information Security

Sushil Jajodia
Consulting Editor
Center for Secure Information Systems
George Mason University
Fairfax, VA 22030-4444
email: jajodia@gmu.edu

The goals of the Springer International Series on ADVANCES IN INFORMATION SECURITY are, one, to establish the state of the art of, and set the course for future research in information security and, two, to serve as a central reference source for advanced and timely topics in information security research and development. The scope of this series includes all aspects of computer and network security and related areas such as fault tolerance and software assurance.

ADVANCES IN INFORMATION SECURITY aims to publish thorough and cohesive overviews of specific topics in information security, as well as works that are larger in scope or that contain more detailed background information than can be accommodated in shorter survey articles. The series also serves as a forum for topics that may not have reached a level of maturity to warrant a comprehensive textbook treatment.

Researchers, as well as developers, are encouraged to contact Professor Sushil Jajodia with ideas for books under this series.

Additional titles in the series:

HOP INTEGRITY IN THE INTERNET by Chin-Tser Huang and Mohamed G. Gouda; ISBN-10: 0-387-22426-3

PRIVACY PRESERVING DATA MINING by Jaideep Vaidya, Chris Clifton and Michael Zhu; ISBN-10: 0-387- 25886-8

BIOMETRIC USER AUTHENTICATION FOR IT SECURITY: From Fundamentals to Handwriting by Claus Vielhauer; ISBN-10: 0-387-26194-X

IMPACTS AND RISK ASSESSMENT OF TECHNOLOGY FOR INTERNET SECURITY:Enabled Information Small-Medium Enterprises (TEISMES) by Charles A. Shoniregun; ISBN-10: 0-387-24343-7

SECURITY IN E-LEARNING by Edgar R. Weippl; ISBN: 0-387-24341-0

IMAGE AND VIDEO ENCRYPTION: From Digital Rights Management to Secured Personal Communication by Andreas Uhl and Andreas Pommer; ISBN: 0-387-23402-0

INTRUSION DETECTION AND CORRELATION: Challenges and Solutions by Christopher Kruegel, Fredrik Valeur and Giovanni Vigna; ISBN: 0-387-23398-9

THE AUSTIN PROTOCOL COMPILER by Tommy M. McGuire and Mohamed G. Gouda; ISBN: 0-387-23227-3

ECONOMICS OF INFORMATION SECURITY by L. Jean Camp and Stephen Lewis; ISBN: 1-4020-8089-1

PRIMALITY TESTING AND INTEGER FACTORIZATION IN PUBLIC KEY CRYPTOGRAPHY by Song Y. Yan; ISBN: 1-4020-7649-5

SYNCHRONIZING E-SECURITY by Godfried B. Williams; ISBN: 1-4020-7646-0

Additional information about this series can be obtained from http://www.springeronline.com

Computer Viruses and Malware

by

John Aycock
University of Calgary
Canada

Springer

John Aycock
University of Calgary
Dept. Computer Science
2500 University Drive N.W.
CALGARY AB T2N 1N4
CANADA

Library of Congress Control Number: 2006925091

Computer Viruses and Malware
by John Aycock, University of Calgary, AB, Canada

ISBN-13: 978-0-387-30236-2
ISBN-10: 0-387-30236-0
e-ISBN-13: 978-0-387-34188-0
e-ISBN-10: 0-387-34188-9

Printed on acid-free paper.

Printed in the United States of America.

9 8 7 6 5 4 3 2 1

springer.com

To all the two-legged critters
in my house

Contents

List of Figures

Preface

It seemed like a good idea at the time. In 2003, I started teaching a course on computer viruses and malicious software to senior undergraduate and graduate students at the University of Calgary. It's been an interesting few years. Computer viruses are a controversial and taboo topic, despite having such a huge impact on our society; needless to say, there was some backlash about this course from outside the University.

One of my initial practical concerns was whether or not I could find enough detailed material to teach a 13-week course at this level. There were some books on the topic, but (with all due respect to the authors of those books) there were none that were suitable for use as a textbook.

I was more surprised to find out that there was a lot of information about viruses and doing "bad" things, but there was very little information about anti-virus software. A few quality minutes with your favorite web search engine will yield virus writing tutorials, virus source code, and virus creation toolkits. In contrast, although it's comprised of some extremely nice people, the anti-virus community tends to be very industry-driven and insular, and isn't in the habit of giving out its secrets. Unless you know where to look.

Several years, a shelf full of books, and a foot-high stack of printouts later, I've ferreted out a lot of detailed material which I've assembled in this book. It's a strange type of research for a computer scientist, and I'm sure that my academic colleagues would cringe at some of the sources that I've had to use. Virus writers don't tend to publish in peer-reviewed academic journals, and anti-virus companies don't want to tip their hand. I would tend to characterize this detective work more like historical research than standard computer science research: your sources are limited, so you try and authenticate them; you piece a sentence in one document together with a sentence in another document, and you're able to make a useful connection. It's painstaking and often frustrating.

Technical information goes out of date very quickly, and in writing this book I've tried to focus on the concepts more than details. My hope is that the

concepts will still be useful years from now, long after the minute details of operating systems and programming languages have changed. Having said that, I've included detail where it's absolutely necessary to explain what's going on, and used specific examples of viruses and malicious software where it's useful to establish precedents for certain techniques. Depending on why you're reading this, a book with more concrete details might be a good complement to this material.

Similarly, if you're using this as a textbook, I would suggest supplementing it with details of the latest and greatest malicious software that's making the rounds. Unfortunately there will be plenty of examples to choose from. In my virus course, I also have a large segment devoted to the law and ethics surrounding malicious software, which I haven't incorporated here – law is constantly changing and being reinterpreted, and there are already many excellent sources on ethics. Law and ethics are very important topics for any computer professional, but they are especially critical for creating a secure environment in which to work with malicious software.

I should point out that I've only used information from public sources to write this book. I've deliberately excluded any information that's been told to me in private conversations, and I'm not revealing anyone's trade secrets that they haven't already given away themselves.

I'd like to thank the students I've taught in my virus course, who pushed me with their excellent questions, and showed much patience as I was organizing all this material into some semi-coherent form. Thanks too to those in the anti-virus community who kept an open mind. I'd also like to thank the people who read drafts of this book: Jörg Denzinger, Richard Ford, Sarah Gordon, Shannon Jaeger, Cliff Marcellus, Jim Uhl, James Wolfe, and Mike Zastre. Their suggestions and comments helped improve the book as well as encourage me. Finally, Alan Aycock suggested some references for Chapter 10, Stefania Bertazzon answered my questions about rational economics, Moustafa Hammad provided an Arabic translation, and Maryam Mehri Dehnavi translated some Persian text for me. Of course, any errors that remain are my own.

JOHN AYCOCK

Chapter 1

WE'VE GOT PROBLEMS

In ancient times, people's needs were simple: food, water, shelter, and the occasional chance to propagate the species. Our basic needs haven't changed, but the way we fulfill them has. Food is bought in stores which are fed by supply chains with computerized inventory systems; water is dispensed through computer-controlled water systems; parts for new shelters come from suppliers with computer-ridden supply chains, and old shelters are bought and sold by computer-wielding realtors. The production and transmission of energy to run all of these systems is controlled by computer, and computers manage financial transactions to pay for it all.

It's no secret that our society's infrastructure relies on computers now. Unfortunately, this means that a threat to computers *is* a threat to society. But how do we protect our critical infrastructure? What are the problems it faces?

1.1 Dramatis Personae

There are four key threats to consider. These are the four horsemen of the electronic apocalypse: spam, bugs, denials of service, and malicious software.

Spam The term commonly used to describe the abundance of unsolicited bulk email which plagues the mailboxes of Internet users worldwide. The statistics vary over time, but suggest that over 70% of email traffic currently falls into this category.[1]

Bugs These are software errors which, when they crop up, can kill off your software immediately, if you're lucky. They can also result in data corruption, security weaknesses, and spurious, hard-to-find problems.

Denials of service Denial-of-service attacks, or DoS attacks,[2] starve legitimate usage of resources or services. For example, a DoS attack could use

up all available disk space on a system, so that other users couldn't make use of it; generating reams of network traffic so that real traffic can't get through would also be a denial of service. Simple DoS attacks are relatively easy to mount by simply overwhelming a machine with requests, as a toddler might overwhelm their parents with questions. Sophisticated DoS attacks can involve more finesse, and may trick a machine into shutting a service down instead of flooding it.

Malicious software The real war is waged with malicious software, or malware. This is software whose intent is malicious, or whose effect is malicious. The spectrum of malware covers a wide variety of specific threats, including viruses, worms, Trojan horses, and spyware.

The focus of this book is malware, and the techniques which can be used to detect, detain, and destroy it. This is not accidental. Of the four threats listed above, malware has the deepest connection to the other three. Malware may be propagated using spam, and may also be used to send spam; malware may take advantage of bugs; malware may be used to mount DoS attacks. Addressing the problem of malware is vital for improving computer security. Computer security is vital to our society's critical infrastructure.

1.2 The Myth of Absolute Security

Obviously we want our computers to be secure against threats. Unfortunately, there is no such thing as absolute security, where a computer is either secure or it's not. You may take a great deal of technical precautions to safeguard your computers, but your protection is unlikely to be effective against a determined attacker with sufficient resources. A government-funded spy agency could likely penetrate your security, should they be motivated to do so. Someone could drive a truck through the wall of your building and steal your computers. Old-fashioned ways are effective, too: there are many ways of coercing people into divulging information.[3]

Even though there is no absolute computer security, relative computer security can be considered based on six factors:

- What is the importance of the information or resource being protected?
- What is the potential impact, if the security is breached?
- Who is the attacker likely to be?
- What are the skills and resources available to an attacker?
- What constraints are imposed by legitimate usage?
- What resources are available to implement security?

Breaking down security in this way changes the problem. Security is no longer a binary matter of secure or not-secure; it becomes a problem of risk management,[4] and implementing security can be seen as making tradeoffs between the level of protection, the usability of the resulting system, and the cost of implementation.

When you assess risks for risk management, you must consider the risks posed to you by others, *and* consider the risks posed to others by you. Everybody is your neighbor on the Internet, and it isn't farfetched to think that you could be found negligent if you had insufficient computer security, and your computers were used to attack another site.[100]

1.3 The Cost of Malware

Malware unquestionably has a negative financial impact, but how big an impact does it really have?[101] It's important to know, because if computer security is to be treated as risk management, then you have to accurately assess how much damage a lapse in security could cause.

At first glance, gauging the cost of malware incidents would seem to be easy. After all, there are any number of figures reported on this, figures attributed to experts. They can vary from one another by an order of magnitude, so if you disagree with one number, you can locate another more to your liking. I use the gross domestic product of Austria, myself – it's a fairly large number, and it's as accurate an estimate as any other.

In all fairness, estimating malware cost is a very hard problem. There are two types of costs to consider: real costs and hidden costs.

Real costs These are costs which are apparent, and which are relatively easy to calculate. If a computer virus reduced your computer to a bubbling puddle of molten slag,[5] the cost to replace it would be straightforward to assess. Similarly, if an employee can't work because their computer is having malware removed from it, then the employee's lost productivity can be computed. The time that technical support staff spend tracking down and fixing affected computers can also be computed. Not all costs are so obvious, however.

Hidden costs Hidden costs are costs whose impact can't be measured accurately, and may not even be known. Some businesses, like banks and computer security companies, could suffer damage to their reputation from a publicized malware incident. Regardless of the business, a leak of proprietary information or customer data caused by malware could result in enormous damage to a company, no different than industrial espionage. Any downtime could drive existing customers to a competitor, or turn away new, potential customers.

This has been cast in terms of business, but malware presents a cost to individuals, too. Personal information stolen by malware from a computer, such as passwords, credit card numbers, and banking information, can give thieves enough for that tropical vacation they've always dreamed of, or provide a good foundation for identity theft.

1.4 The Number of Threats

Even the exact number of threats is open to debate. A quick survey of competing anti-virus products shows that the number of threats they claim to detect can vary by as much as a factor of two. Curiously, the level of protection each affords is about the same, meaning that more is not necessarily better.

Why? There is no industry-wide agreement on what constitutes a "threat," to begin with. It's not surprising, given that fact alone, that different anti-virus products would have different numbers – they aren't all counting the same thing. For example, there is some dispute as to whether or not automatically-generated viruses produced by the same tool should be treated as individual threats, or as only one threat. This came to the fore in 1998, when approximately 15,000 new automatically-generated viruses appeared overnight.[102] It is also difficult to amass and correctly maintain a malware collection,[103] and inadvertent duplication or misclassification of malware samples is always a possibility. There is no single clearinghouse for malware.

Another consideration is that the reported numbers are only for threats that are known about. Ideally, computers should be protected from both known *and* unknown threats. It's impossible to know about unknown threats, of course, which means that it's impossible to precisely assess how well-protected your computers are against threats.

Different anti-virus products may employ different detection techniques, too. Not all methods of detection rely on exhaustive compilations of known threats, and generic detection techniques routinely find both known and unknown threats without knowing the exact nature of what they're detecting.

Even for known threats, not all may endanger your computers. The majority of malware is targeted to some specific combination of computer architecture and operating system, and sometimes even to a particular application. Effectively these act as preconditions for a piece of malware to run; if any of these conditions aren't true – for instance, you use a different operating system – then that malware poses no direct threat to you. It is inert with respect to your computers.

Even if it can't run, malware may carry an indirect liability risk if it passes through your computers from one target to another. For example, one unaffected computer could provide a shared directory; someone else's compromised computer could deposit malware in that shared directory for later propagation. It is prudent to look for threats to all computers, not just to your own.

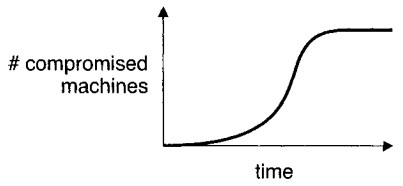

Figure 1.1. Worm propagation curve

1.5 Speed of Propagation

Once upon a time, the speed of malware propagation was measured in terms of weeks or even months. This is no longer the case.

A typical worm propagation curve is shown in Figure 1.1. (For simplicity, the effects on the curve from defensive measures aren't shown.) At first, the worm spreads slowly to vulnerable machines, but eventually begins a period of exponential growth when it spreads extremely rapidly. Finally, once the majority of vulnerable machines have been compromised, the worm reaches a saturation point; any further growth beyond this point is minimal.

For a worm to spread more quickly, the propagation curve needs to be moved to the left. In other words, the worm author wants the period of exponential growth to occur earlier, preferably before any defenses have been deployed. This is shown in Figure 1.2a.

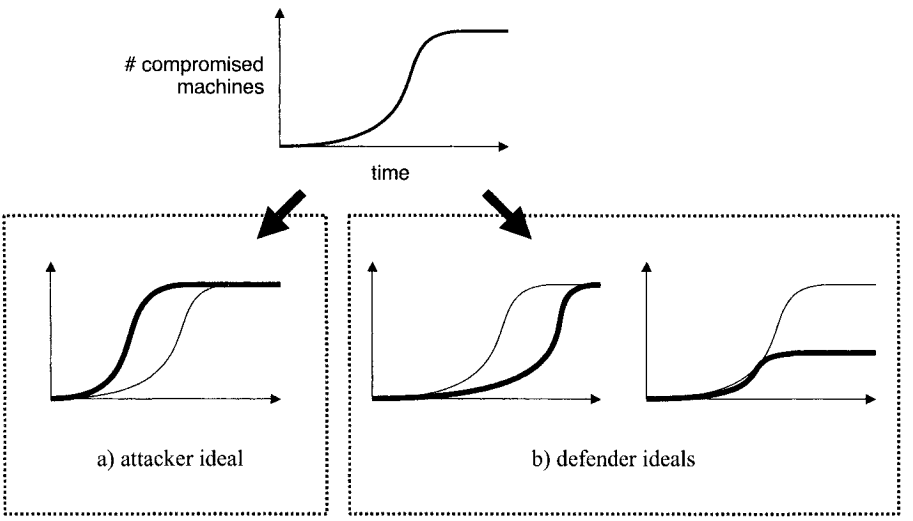

Figure 1.2. Ideal propagation curves for attackers and defenders

On the other hand, a defender wants to do one of two things. First, the propagation curve could be pushed to the right, buying time to construct a defense before the worm's exponential growth period. Second, the curve could be compressed downwards, meaning that not all vulnerable machines become compromised by the worm. These scenarios are shown in Figure 1.2b.

The time axis on these figures has been deliberately left unlabeled, because the exact propagation rate will depend on the techniques that a particular worm uses. However, the theoretical maximum speed of a carefully-designed worm from initial release until saturation is startling: 510 milliseconds to 1.3 seconds.[6] In less than two seconds, it's over. No defense that relies on any form of human intervention will be fast enough to cope with threats like this.

1.6 People

Humans are the weak link on several other fronts too, all of which are taken advantage of by malware.

By their nature, humans are trusting, social creatures. These are excellent qualities for your friends to have, and also for your victims to possess: an entire class of attacks, called social engineering attacks, are quick to exploit these desirable human qualities.

Social engineering aside, many people simply aren't aware of the security consequences of their actions. For example, several informal surveys of people on the street have found them more than willing to provide enough information for identity theft (even offering up their passwords) in exchange for chocolate, theater tickets, and coffee vouchers.[104]

Another problem is that humans – users – don't demand enough of software vendors in terms of secure software. Even for security-savvy users who want secure software, the security of any given piece of software is nearly impossible to assess.

Secure software is software which can't be exploited by an attacker. Just because some software hasn't been compromised is no indication that it's secure – like the stock market, past performance is no guarantee of future results. Unfortunately, that's really the only guideline users have to judge security: the absence of an attack. Software security is thus an anti-feature for vendors, because it's intangible. It's no wonder that vendors opt to add features rather than improve security. Features are easier to sell.

Features are also easier to buy. Humans are naturally wooed by new features, which forms a vicious cycle that gives software vendors little incentive to improve software security.

1.7 About this Book

Malware poses an enormous problem in the context of faulty humans and faulty software security. It could be that malware is the natural consequence of the presence of these faults, like vermin slipping through building cracks in the real world. Indeed, names like "computer virus" and "computer worm" bring to mind their biological real-world counterparts.

Whatever the root cause, malware is a problem that needs to be solved. This book looks at malware, primarily viruses and worms, and its countermeasures. The next chapter lays the groundwork with some basic definitions and a timeline of malware. Then, on to viruses: Chapters 3, 4, and 5 cover viruses, anti-virus techniques, and anti-anti-virus techniques, in that order. Chapter 6 explains the weaknesses that are exploited by malware, both technical and social – this is necessary background for the worms in Chapter 7. Defenses against worms are considered in Chapter 8. Some of the possible manifestations of malware are looked at in Chapter 9, followed by a look at the people who create malware and defend against it in Chapter 10. Some final thoughts on defense are in Chapter 11.

The convention used for chapter endnotes is somewhat unusual. The notes tend to fall into two categories. First, there are notes with additional content related to the text. These have endnote numbers from 1–99 within a chapter. Second, there are endnotes that provide citations and pointers to related material. This kind of endnote is numbered 100 or above. The intent is to make the two categories of endnote easily distinguishable in the text.

A lot of statements in this book are qualified with "can" and "could" and "may" and "might." Software is infinitely malleable and can be made to do almost anything; it is hubris to make bold statements about what malware can and can't do.

Finally, this is not a programming book, and some knowledge of programming (in both high- and low-level languages) is assumed, although pseudocode is used where possible. A reasonable understanding of operating systems and networks is also beneficial.

1.8 Some Words of Warning

Self-replicating software like viruses and worms has proven itself to be very difficult to control, even from the very earliest experiments.[7] While self-replicating code may not intentionally be malicious, it can have similar effects regardless. Of course, the risks of overtly malicious software should be obvious. Any experiments with malware, or analysis of malware, should be done in a secure environment designed specifically for that purpose. While it's outside the scope of this book to describe such a secure environment – the details would

be quickly out of date anyway – there are a number of sources of information available.[105]

Another thing to consider is that creation and/or distribution of malware may violate local laws. Many countries have computer crime legislation now,[8] and even if the law was violated in a different jurisdiction from where the perpetrator is physically located, extradition agreements may apply.[106] Civil remedies for victims of malware are possible as well.

Ironically, some dangers lurk in defensive techniques too. Some of the material in this book is derived from patent documents; the intent is to provide a wide range of information, and is not in any way meant to suggest that these patents should be infringed. While every effort has been made to cite relevant patents, it is possible that some have been inadvertently overlooked. Furthermore, patents may be interpreted very broadly, and the applicability of a patent may depend greatly on the skill and financial resources of the patent holder's legal team. Seek legal advice before rushing off to implement any of the techniques described in this book.

Notes for Chapter 1

1 Based on MessageLabs' sample size of 12.6 billion email messages [203]. This has a higher statistical significance than 99% of statistics you would normally find.

2 Note the capitalization – "DOS" is an operating system, "DoS" is an attack.

3 In cryptography, this has been referred to as "rubber-hose" cryptanalysis [279].

4 Schneier has argued this point of view, and that computer security is an untapped market for insurance companies, who are in the business of managing risk anyway [280].

5 Before any urban legends are started, computer viruses can't do this.

6 These numbers (510 ms for UDP-based worms, 1.3 s for TCP-based worms) are the time it takes to achieve 95% saturation of a million vulnerable machines [303].

7 For example, Cohen's first viruses progressed surprisingly quickly [74], as did Duff's shell script virus [95], and an early worm at Xerox ran amok [287].

8 Computer crime laws are not strictly necessary for prosecuting computer crimes that are just electronic versions of "traditional" crimes like fraud [56], but the trend is definitely to enact computer-specific laws.

100 Owens [237] discusses liability potential in great detail.

101 This section is based on Garfink and Landesman [117], and Ducklin [94] touches on some of the same issues too.

102 Morley [213]. Ducklin [94] has a discussion of this issue, and of other ways to measure the extent of the virus problem.

103 Bontchev [39] talks about the care and feeding of a "clean" virus library.

104 The informal surveys were reported in [30] (chocolate), [31, 274] (theater tickets), and [184] (coffee vouchers). Less amusing, but more rigorous, surveys have been done which show similar problems [270, 305].

105 There are a wide range of opinions on working with malware, ranging from the inadequate to the paranoid. As a starting point, see [21, 75, 187, 282, 288, 312].

106 Although U.S.-centric, Soma et al. [295] give a good overview of the general features of extradition treaties.

Chapter 2

DEFINITIONS AND TIMELINE

It would be nice to present a clever taxonomy of malicious software, one that clearly shows how each type of malware relates to every other type. However, a taxonomy would give the quaint and totally incorrect impression that there is a scientific basis for the classification of malware.

In fact, there is no universally-accepted definition of terms like "virus" and "worm," much less an agreed-upon taxonomy, even though there have been occasional attempts to impose mathematical formalisms onto malware.[100] Instead of trying to pin down these terms precisely, the common characteristics each type of malware typically has are listed.

2.1 Malware Types

Malware can be roughly broken down into types according to the malware's method of operation. Anti-"virus" software, despite its name, is able to detect all of these types of malware.

There are three characteristics associated with these malware types.

1 *Self-replicating* malware actively attempts to propagate by creating new copies, or *instances*, of itself. Malware may also be propagated passively, by a user copying it accidentally, for example, but this isn't self-replication.

2 The *population growth* of malware describes the overall change in the number of malware instances due to self-replication. Malware that doesn't self-replicate will always have a zero population growth, but malware with a zero population growth may self-replicate.

3 *Parasitic* malware requires some other executable code in order to exist. "Executable" in this context should be taken very broadly to include anything that can be executed, such as boot block code on a disk, binary code