The Future of Malware

Stephen Trilling and Carey Nachenberg
Symantec Corporation

About the Authors

Stephen Trilling

Stephen Trilling is the Director of Research at the Symantec AntiVirus Research Center. Trilling currently oversees a worldwide team responsible for adding new anti-virus technologies to Symantec's complete range of Norton AntiVirus products. This team also analyzes all viruses received from Symantec customers. Trilling began his career with Symantec in 1995 as a senior developer on the Norton Utilities team, working on Norton Utilities for Windows 95 and Windows NT. His work in the anti-virus field began as a senior developer on the Norton AntiVirus team.Prior to joining the Symantec team, Trilling pursued a career as a stand-up comedian. He holds a bachelor's degree in computer science from Yale University as well as a master's in computer science from the Massachusetts Institute of Technology. He is a member of IEEE.

Carey Nachenberg

Carey Nachenberg is Chief Architect of the Symantec AntiVirus Research Center. He researches, designs and develops new anti-virus technologies for the award-winning Norton AntiVirus line of products. He has worked at Symantec for six years as a software engineer and architect on Norton Commander, Norton Desktop for DOS and Norton AntiVirus. Nachenberg holds both a bachelor's and a master's degree in Computer Science and Engineering from the University of California at Los Angeles. His master's thesis covers the topic of polymorphic computer virus detection

Mailing addresses: Symantec Corporation, 2500 Broadway, Suite 200, Santa Monica, CA, 90404 USA; Phone: +1 310 453 4600; Fax: +1 310 453 0636; E-mail: cnachenberg@symantec.com, strilling@symantec.com

Descriptors

virus, Trojan Horse, malware, malicious code, e-mail, anti-virus, payload, ActiveX, Java, worms, Internet, wildlife

Reference to this paper should be made as follows: Trilling. S and Nachenberg, C. (1999) 'The Future of Malware', EICAR 1999 Best Paper Proceedings.

EICAR Proceedings 1999 Edited by Urs E. Gattiker, Pia Pedersen and Karsten Petersen
© 1999 EICAR c/o TIM-World ApS, Aalborg Denmark

The Future of Malware

Abstract

In this paper, we explore the current and potential future landscape of the malicious code problem. We first discuss each of the known types of malicious code threats (both viruses and Trojans) and attempt to assess their possible growth in the future. We also speculate on potential future malicious code threats. We then cover the various delivery mechanisms by which a user might receive each of these threats. Next, we discuss current and possible future payloads that could be delivered to users through each of these malicious mechanisms. Finally, we suggest a variety of possible technology options that could be used to combat each of these threats.

Malicious software has existed since the dawn of computing and it has become increasingly prevalent in recent years. One of the most famous malware experiments was the Internet worm. In 1988, Robert Morris, a college student, created a computer worm that infected thousands of computers connected to the Internet (Denning, 1990). At the time, this was probably the most well known malware ever created. However, as early personal computer users know, this was just one of many digital threats to emerge in the 1980s.

The "computer bulletin board", or BBS became popular amongst computer users in the 1980s (Gordon & Chess, 1998). Computer enthusiasts used this medium to meet people, send e-mail and exchange programs. Unfortunately, a small number of these programs were malicious in nature. Designed to format the user's hard drive, delete files or open a back door in bulletin board software, these early Trojan horses became increasingly prevalent. In fact, early BBS users created lists and advisories to alert users to these threats. One of the most popular was the "Dirty Dozen" list. This list described the most common malware of the time: the file names of the offending Trojan horses, their payloads, destructive capabilities, and other key characteristics (Newhouse, 1988).

It wasn't much time before computer viruses started emerging (Bassham & Polk, 1992). In 1986, the Pakistani Brain virus began making the rounds. Unfortunately, the Jerusalem virus and tens of thousands of others were close on Brain's heels. Today, the malicious code problem is greater than ever. The growth in prevalence of the problem can be attributed to four factors. First, the standardization and widespread use of a computer platform, namely the "WINTEL" PC, has had a huge impact on the number of malicious programs. Second, computer users are becoming increasingly more technically savvy and skilled in the area of computer programming. Third, malware authors are increasingly sharing their source code and know-how with other hackers. Finally, programming tools are becoming easier to use, lowering the bar and allowing virtually anyone to create both useful and malicious programs.

The nature of the problem is likely to evolve in the future as the Internet becomes increasing ubiquitous and more people become continuously connected. This paper will examine the current landscape of the malicious code problem. It will then attempt to predict what are the most likely threats that will be encountered by users in the future based on observed and predicted trends. We will discuss all aspects of these threats including replication mechanisms, payloads, and methods of delivery. Finally, we will discuss the different anti-malware technologies that could be developed and deployed to address these threats.

In the course of this paper we will use the term *malware* to broadly refer to all types of potentially malicious code, including viruses, worms, Trojans (including ActiveX and Java).

Potential Future Malicious Code Types

The malware of the future will most likely be an extension of the malicious threats we encounter today. Future changes to users' computing environments including changes in operating systems, e-mail systems, networking capabilities, etc. will have a profound influence on the types of threats we may eventually encounter. Regardless of exactly how the scenario plays out, these threats are likely to always fall into two major categories:

- Wildlife
- Trojans

We now discuss each of these types of threats in more detail.

By contrast, we will use the term *Trojan* to describe all other non-replicating threats. We now discuss each of these two types of threats in greater detail.

Wildlife

For the purposes of this paper, we will define the term *wildlife* to refer to malicious code that spreads on its own through a computer system or computer network; this term is often used to describe viruses and worms. We now discuss these two potential types of wildlife in more detail.

Viruses

As mentioned above, viruses are a form of wildlife, namely they are computer programs capable of spreading on their own. For a more formal definition and general discussion of computer viruses, see (Cohen, 1994). Viruses continue to remain a threat to computer users. However, the prevalence of different types of viruses has been evolving (see Appendix B). We now discuss the potential future landscape of each of the major virus types.

DOS file viruses and boot viruses. Viruses infecting DOS files and boot records were the earliest types of wildlife threats. As shown in the statistics in Appendix B, the prevalence of boot and file viruses has generally diminished in recent years. Our expectation is that this threat will continue to decrease. Newer operating systems, such as Windows 95 and Windows NT effectively neuter boot record viruses. Furthermore, DOS file viruses will continue to diminish as virus writers concentrate on macro viruses and Windows-based virus threats.

Macro viruses. More recently, we have seen an upsurge in the creation of viruses which infect macros in Microsoft *Word* and *Excel* (and, more recently, *Access*) documents. Furthermore, bugs in *Microsoft Word 6 and Word9*5 appear to be causing random mutations in macro viruses, creating new variants (Chess,

1997). Viral submissions that we have received at Symantec lead us to believe that as many as 70% of all new macro viruses are created because of this phenomenon. Luckily, this problem has not been identified in *Microsoft Word97* or later versions. Because of this, as users migrate to *Office97, Office2000* and later versions, it seems possible that we will see a decrease in the growth rate of new macro viruses.

In addition, Microsoft has added a number of new generic anti-virus technologies directly into the latest Office products (*Office97* and beyond) (Microsoft, 1997). These anti-virus mechanisms severely limit the replication techniques that viruses may employ, and will effectively neuter many existing macro viruses. This will have a two-pronged effect. First, it will reduce the overall number of new and existing macro viruses that users must deal with. Second, it will limit the types of logic which can be used to write replicating code, making heuristics a more viable detection option.

We will, however, continue to see new "proof of concept" viruses designed to attack various macro and scripting systems.

We now discuss more recent types of viral threats.

Windows viruses. Based on submissions from our customers at Symantec, we expect that the number and complexity of native Windows viruses will markedly increase over time. The CIH (DeGroot, 1998) and Marburg viruses are two recent high-profile examples of this new type of threat. While computer users tend not to copy entire applications once they have been installed and infected (which would aid in spreading a Windows virus), a number of other user behaviors could contribute to this problem. First, many users share small Windows programs in e-mail (such as joke programs, graphical demos, etc.), and these can and do spread Windows-based viruses. Second, similar programs anonymously posted to newsgroups are often used to spread these viruses effectively (Gryaznov, 1998). Finally, with the ubiquitous use of local area networks, so-called "fast" Windows viruses (viruses which infect on-access) can quickly spread through the network all over an organization. These factors will cause Windows virus infections to increase in prevalence. For a thorough discussion of Windows viruses, see (Szor, 1998).

ActiveX, Java and script-based Internet-enabled viruses. ActiveX and Java are currently two of the most popular mechanisms for delivering content to users browsing the World Wide Web.

We have identified at least two types of potential future Java/ActiveX viruses. The first type of potential Java or ActiveX virus can be called a *parasitic applet virus*. The basic idea is that an end-user browses a web page that contains an infected applet. As part of the browsing process, the user's browser pulls down a copy of the applet and runs it. The applet then searches for other applets on the

user's computer and parasitically attaches itself to these applets. If this user happened to be running their own web site, another end user could then connect to this user's web site and become infected.

The second type of potential Java or ActiveX virus can be called a c*ompanion HTML virus*. In this scenario, an end user browses a web page that contains an infected applet. As part of the browsing process, the user's browser pulls down a copy of the applet and runs it. The applet then makes a backup copy of itself, and searches for other HTML pages to modify. The applet inserts a tag-reference to the backup copy of itself into these HTML pages. Again, if this user happened to be running their own web site, another end user could connect to the site and view a modified web page. This web page would cause a copy of the companion virus to be pulled over to the end user's computer, and the process would continue.

Recently, we have also seen several new viruses that have been implemented using HTML-based scripting languages such as VBScript. These viruses work in a similar fashion to the Companion HTML virus described above with one exception. Rather than inserting a tag-reference to an external Java or ActiveX component, these viruses directly embed themselves in new HTML pages upon infection. These might be called *parasitic HTML viruses,* since they parasitically infect HTML pages.

Luckily, all of the viruses described above will fail to function when run in a web browser, if implemented in Java (or one of the other common scripting languages) because today's web browsers properly employ security measures (using the Java sandbox model and other security models). These security issues are discussed at greater length in (Morar & Chess, 1998). Consequently, such attacks are not yet likely to pose major threats to current users. Furthermore, today most end users don't run their own web sites, at least from their home computer. So such a virus might find its way onto an end user's computer, but will quickly find that it has nowhere to spread (beyond, perhaps, the browser's cache directory). This will limit the ability for a Java, ActiveX or HTML script-based virus to spread. We call this phenomenon the *browsing-serving asymmetry*.

However this type of threat could well grow in the future as more and more users have continuous connections to the Internet (through cable modems, for example). Any user continuously connected to the Internet can use their machine both to surf the web, as well as to host their own home page. The following scenario could then occur: A user browses a web page infected by a viral ActiveX applet on another computer. This applet is then downloaded and run on user's machine, infecting one of the user's own web pages. Finally, an outside user browses this infected web page which infects this new user's web page, and virus can continue to spread. In a sense, these types of threats are ahead of their time – their ability to do widespread damage is more likely to be

brought on by advances in technology rather than by any further "enhancements" to the viral code by humans.

Future threat from viruses. Currently, macro viruses are easily the most common types of viral threat found in-the-wild. We have suggested above that the growth rate of macro viruses may decrease in the future. Regardless, it seems likely that computer viruses in general will nevertheless remain a real threat to end-users and corporate customers. Since viruses spread on their own, they can be hard to trace and very difficult to fully eradicate (because of infections to backup disks, embedded e-mail attachments, etc…). This lack of accountability and traceability of computer viruses is a likely indication that they will continue to plague all types of computer users in the future. Furthermore, new threats that take advantage of the popularity of Microsoft Windows and the World Wide Web, could grow in the future.

We now discuss a second type of wildlife, namely worms.

Worms

Computer worms are programs that spread themselves from computer to computer over a network. Worms, unlike viruses, do not infect programs, diskettes or files with macro capabilities. Instead, they make copies of themselves and send these copies over a network to other targeted machines. For specific examples of worms which have spread in the wild, see (Gordon, 1998) and (Spafford, 1989).

Luckily, the success of these worms has otherwise been relatively limited, in part for the following reasons:

1. It is more difficult to test worms than it is to test viruses. Few people have an entire network at their disposal for testing. Conversely, a user can test a computer virus with a single computer.
2. There are few constantly connected, script-enabled systems where worms can flourish.

However, as with malicious Java/ActiveX, as more people obtain continuous connections to service providers (such as cable connections), the prevalence of worms may dramatically increase. For example, we have already seen hundreds of worms developed for the Internet Relay Chat (IRC) system (in reality, most of these worms were only slight variations of each other) (Gordon, 1994). IRC is a popular way for people to talk with each other in large groups over the Internet. In order to make IRC more usable, script languages have been developed which allow users to automate certain tasks. Programs written in these script languages have the ability to send themselves over the Internet to other IRC users. As such, it is relatively easy for malicious programmers to use these languages to create worms. Luckily, IRC does provide built-in mechanisms to

disable worms; however, many users are not aware of these options. In general, as more users take advantage of similar two-way services (with script capabilities and accessible indexes/directories of connected users) and configure their computers to continually and automatically subscribe, connect and interact with these services (discussion groups, subscription topic groups, etc.), worms will likely increase in prevalence.

The lack of security on home-user systems coupled with the huge variety of possible worm-capable services could make worms a real malware threat for the future. Like computer viruses, worms are anonymous in nature. This makes it very difficult to track the origin of such programs and the introduction of a worm into production systems could even happen easily by a curious, completely non-malicious individual. There is no doubt that many such worms will be posted on web sites and newsgroups (as they are now) for any interested person to play with.

There are also newer e-mail systems that provide a rich medium for computer worms. Many of these latest systems have scripting languages that can be launched when a user reads a message. These scripting systems were designed for convenience and not security; consequently, there is a real risk that worms could be developed using such systems. This means that users really could be at risk from infection, just by opening an e-mail. This is in direct contrast, for example, to non-threatening hoaxes such as “Good Times” which falsely claim that opening certain messages can infect users. To combat such hoaxes, the AV community has traditionally made it clear that users were extremely unlikely to be infected simply by reading e-mail. These new systems make this type of threat very real. For a more detailed discussion of virus hoaxes, see (Gordon, Ford, & Wells, 1997).

Traditionally, we have found that most new viruses and malware originate outside the corporate sector since employees are typically not writing viruses or spreading worms in their jobs. Given that these high-powered e-mail systems are primarily used in corporations, we would expect that these new types of worms would be rare. However, as script-enabled e-mail products gain more widespread acceptance with end-users, such worms could potentially become a much larger problem.

Once virus writers or other curious individuals recognize these emerging malware opportunities, it is likely that we will see a much larger number of worms from outside the business sector.

Trojans

A Trojan horse is a program which appears to serve a useful purpose, yet actually performs an unexpected action which is often malicious. Trojan horse programs typically don’t spread like viruses. They typically perform a malicious

action as soon as they are executed. As such, they must be run by an unsuspecting user or manually introduced by a third party.

Their name comes from a story of Greek mythology in which wooden horse was used by the Greeks to reclaim Helen of Troy. The unsuspecting Trojans thought that the horse was a gift from the Greeks, and knowingly let it into their city. In reality, the horse was filled with Greek soldiers. Once safely inside the city, the soldiers quickly exited from the horse, conquered the city of Troy and recaptured Helen for Greece. For a further discussion on the history of Trojans, see (Gordon et. al., 1998) and (Muttick, 1997).

Just like the original wooden Trojan horse, computer Trojan horses always perform some action which is not expected. There are three distinct sources from which a user might receive a Trojan horse:

- Stationary threats come from a *stationary source* such as a web page
- Anonymous threats come from an *anonymous source* such as anonymous e-mail
- Manually introduced threats are introduced by a human attacker

Each of these involves a different delivery mechanism by which the user receives the Trojan on their machine. We now discuss each of these in more detail. Actually, these delivery mechanisms can equally be used to deliver wildlife. However, we cover these ideas in this section because Trojan horses are dependent upon such delivery mechanisms for their survival as they cannot spread on their own.

Delivery mechanisms

Delivery mechanisms can take one of three forms:

- Stationary threats
- Anonymous threats
- Manual introduction

We now discuss each of these in turn.

Stationary threats. *Stationary threats* refer to malicious programs that have been posted to a web site for users to download. These programs are stationary on the web site, easily downloaded and executed by any unsuspecting user.

Stationary threats such as Java and ActiveX have received much attention in the security world, but the actual threat from malicious stationary threats has been negligible to non-existent. We believe that stationary threats been limited up until now for one primary reason: Java and ActiveX, the two most popular types of executable content, must be explicitly placed on a web page which is

registered to a given user or company. If such a user places malicious code on their web site they will be held accountable for the damage done to users of the web-site. Such threats are easily traceable and people are very aware that they will be held legally accountable for anything that they intentionally post.

As we discuss in the next section, these types of malicious threats can also be posted anonymously to newsgroups and other on-line exchange systems. Let's consider a scenario where a once-anonymous applet becomes a stationary applet. A user surfs the web and finds a new applet which has been posted anonymously. The user tries the applet and decides to post it on their own web site because of its apparent entertainment value, not knowing that the applet is truly malicious. Soon other users browse the first user's site and try the applet themselves. They too like its functionality and, in turn, post the applet on their own web pages. Now, we have a malicious applet on numerous web sites. Each of these users has effectively vouched for the safety of such an applet, when in fact the applet is malicious. In this case we have an apparent stationary threat, which has grown out of an initially anonymous posting. This type of threat will likely increase, even though strictly stationary threats (which did not start out as anonymous threats) will likely grow more slowly because of their clear traceability.

The most successful malicious applets distributed in this manner will be those that have a timed "trigger feature" which prevents them from causing harm immediately. Any Trojan without this delay would quickly be caught and removed from on-line services. On the other hand, a Trojan which only started performing malicious actions two months after its initial posting would probably gain widespread acceptance on web sites before it ever did any damage. Such an applet, which would appear to be useful or fun (perhaps display a spinning logo), could attain widespread distribution and wreak havoc on the on-line world. It is likely that given the increasingly high volume of posted applets, fewer and fewer applets will undergo any scrutiny at all. We discuss the notion of a "trigger condition" at more length below in the section on payloads.

We do expect that the number of actual theft-motivated attacks using Java or ActiveX will grow in the future. In this scenario, the web operator takes a calculated risk and places malicious Java/ActiveX code on the web site to obtain credit card numbers, passwords etc. As soon as the person has made enough money or obtained enough passwords, they can disappear with their money or information. Each individual scam will probably be short lived and use entirely *new,* specially tailored ActiveX/Java code in order to avoid detection by anti-malware scanners.

ActiveX – In the short term, ActiveX objects will remain the most dangerous active content that web-browsing users will be exposed to. ActiveX objects are basically 32-bit Windows executable files and have the capabilities of any other Windows application. These applets can delete files, attempt to format the hard

drive, alter the registry, export sensitive information over the Internet, and other destructive actions.

Java – Unsigned Java is a fairly secure executable medium and we will probably not see too many new truly malicious Java applets. The Java VM (Virtual Machine) is designed to prevent Java applets from accessing the host system's resources, and has been shown to be fairly robust. To date, most "malicious" Java applets have merely been annoying.

In principle, it could be possible from someone to create a Java applet which penetrated the Java VM – such a threat would be very dangerous since it could potentially compromise a computer's file system through a web browser. However, it seems likely that should such an applet ever be created, it could easily be detected through standard signature-based AV detection schemes – any code to penetrate the Java VM would almost surely have clear characteristics which could easily be encoded into an exact detection for any AV product.

It is currently possible for users to configure their browsers in such a way that a *signed* Java applet has full access to the host system. Such a Java applet could perform most if not all of the same damaging functions that could be performed by an ActiveX object. However, the signing requirement will probably deter most people from producing such malicious applets; obviously, the developer would be quickly held accountable for any damage caused by the applet.

For a further discussion of current malicious Java and ActiveX threats, see (Branigan, 1998).

In summary, we expect the number of truly stationary threats to increase at a slow rate because of the clear accountability involved. However, as we discuss above, the conversion of an anonymously-posted threat to a stationary threat can and will be a growing problem for users.

Anonymous threats. *Anonymous threats* include the gamut of Trojan horse programs that can be posted anonymously to public discussion groups or sent in mass mailings to unsuspecting end-users. Since malicious threats could easily be sent via anonymous e-mail, it can be very difficult to track their source. Such anonymous threats seem far more likely than malicious Java/ActiveX – one is much less likely to get caught through sending anonymous e-mail than they are by posting a malicious program on a public web site. Currently, the two most likely vectors for these anonymous Trojan horses are anonymous (or forged sender) mass e-mailings, and anonymous (or forged sender) USENET newsgroup posting. In addition, there are many other services support such anonymous transmission (such as IRC).

In addition, the popularity of graphical "joke" executables is increasing - we have received many such programs over the past year. These tend to be humorous or fun small programs which friends pass on to each other over the Internet. While many of these programs are innocuous, there is nothing preventing such a program from stealing passwords, exporting the last edited document, etc. Trojan horses don't necessarily always simply format hard drives; -they can have much more subtle payloads, as we'll discuss further below. Since most users don't track usage of Windows sockets (used for Internet connections) while watching a joke program's graphics and listening to its music, it would be very easy for such a program to export sensitive information past a firewall and out to a malicious attacker.

With the increase of third-party web-hosting sites such as *GeoCities*, we may see an increase in what appear to be stationary attacks. These third-party services provide free space for users to put up their own web page. Given that users can provide false information and set up their page anonymously, we may see malicious individuals posting malicious applets on what most end-users would expect to be a fully accountable site. Nothing could be farther from the truth. This is effectively an alternate way to anonymously distribute malicious applets.

We expect anonymous threats (whether they appear to be stationary, or not) to grow markedly over the coming years.

Manual Introduction. Finally, it is always possible for a malicious Trojan to be introduced onto any user's machine through some manual, secret means. In fact, a covert operative could potentially install a Trojan horse in an organization or government. Mechanisms by which this might be done are outside the scope of this paper but it bears mentioning that this sort of threat always exists.

We now discuss another important characteristic of malicious threats – payloads.

Payload

A payload is an intentional side effect (in the case of wildlife) or primary goal (in the case of Trojan horses) of the malicious code. In many cases, the payload has a "trigger condition." When the condition is met, the payload is "delivered."

In the case of wildlife (viruses and worms), the payload is typically only delivered when this "trigger condition" occurs, which may be quite rare. As an example, the well-known Michelangelo boot virus only delivers its payload on March 6 every year. If an infected user starts up their machine on March 6, all of their computer's data will be lost. On any other day of the year, this virus will do nothing destructive. This mechanism is typical of payloads – the more the virus or worm can spread before being noticed, the more widespread the damage when it drops its payload.

By contrast, Trojans typically deliver their payload every time they are executed. Trojans do not spread on their own, so the payload is their only destructive action. This means that an individual user who executes a Trojan is much more likely to experience the payload than an individual user who executes a virus.

If the Trojan has a clear and malicious payload (like formatting the user's hard drive) it is likely to do damage to far fewer machines since it will be quickly noticed and cannot spread on its own. On the other hand, as discussed further below, some Trojan horses have payloads which can elude observation for a considerably longer time or even indefinitely. This makes it conceivable that such threats will be spread to a much larger number of machines by unsuspecting users.

There are four major types of payloads:

- Malicious Payloads
- Data Export
- Data Import
- Client- Server

We now discuss each of these in detail.

Malicious Payloads

We refer to a *malicious payload* to mean some destructive one-time action, which a program uses to incapacitate a user's access to a data or programs on their computer. Some examples of malicious payloads include:

- Formatting the user's hard drive
- Deleting files on the user's computer
- Random corruption of data on the user's machine - as an example, the *Ripper* virus will corrupt random bits of data on an infected user's machine
- Encryption of the user's data
- Rendering the user's machine unbootable

We expect to see about the same level of malicious payloads in future virus and worm-based threats. The more rare the trigger condition, the more likely these threats will be to do potentially widespread damage since they will be able to replicate or be unintentionally spread without being noticed or suspected.

Data Export

Data export is another type of payload that will definitely continue to grow. In this case, the malware exports the user's data to an outside hacker. Some possible examples of data that could be exported include:

- User passwords
- Credit card information
- Documents
- Spreadsheets
- Keystrokes
- Conversations (through a computer-connected microphone)

Since the payoff to the hacker from this type of threat is potentially quite large, we believe the threat will likely increase. Furthermore modern operating systems have built-in communications APIs which are extremely easy to use; these communications services, coupled with the widespread use of the Internet have made it easier than ever to export information to a malicious outside party (for example, using sockets).

These data export threats will likely have two further defining characteristics:

- Visible – visible threats mimic the execution of a legitimate program that gets information from the user. This information is then sent to the attacker who can use it for malicious purposes. For example, there has been a recent upsurge in the number of AOL "password stealer" Trojans (Rosenberger, 1998). These appear to be legitimate AOL programs asking for the user's password, but are actually very cleverly disguised Trojan horses.

- Invisible – threats such as the recently discovered Back Orifice and NetBus are in this category. Once up and running on a victim's machine, this type of threat will silently steal their desired information and export it to the attacker. Invisible threats can be traced by monitoring traffic on the computer's ports, of course, but most individual users are not likely to notice such traffic.

In general, we expect the number of both visible and invisible export attacks to increase dramatically over the coming months and years.

Data Import

It is also possible for a malicious program to import further data/programs in order to do damage. In this scenario, the malicious program is a small "stub" which, once running, connects to the web and downloads additional malicious code. Since HTTP pulling is typically allowed in most corporations, firewalls will not provide much protection from these types of threats. As with data export, our expectation is that this type of threat will increase.

Client-server

Client/server threats, in which a malicious client is contacting the server installed on the victim's machine, will also likely continue to increase. Back Orifice and Sockets de Troie are two examples of such types for threats. The Sockets de

Troie server is more worrisome because it can move itself around on peer to peer networks (giving it worm-like behavior) and allowing a hacker to potentially connect to any number of machines.

As more and more people are continuously connected to the Internet, our expectation is that end-users may see an ever-increasing impact from such threats. As an example, there recently was a report of a new client-server Trojan infecting 15,000 Internet users (Wired, 1998). This Trojan also used data import as a payload – its first action was to download a specific program from a *GeoCities* web site. Once running on a user's machine, the Trojan potentially allowed a hacker to connect to the machine and capture information (in fact it is quite likely that this Trojan was Back Orifice plug-in).

At least some of today's client-server Trojan horses can be completely foiled by firewall software. For instance, assume that a Back Orifice server is running inside an organization and the attacker is outside the organization's firewall. In order to control the server, the attacker must connect to the server from outside the firewall and send commands. However, most firewalls restrict external socket connections into arbitrary machines on the corporate intranet, thereby protecting against such a threat.

While firewalls can protect against threats like Back Orifice, there is no reason why other Trojan horses must a similar client-initiated control scheme. The server could just as easily initiate the communications using a common protocol, bypassing the firewall completely. This is because most firewalls are configured to prevent external access to the intranet, but not to prevent internally initiated connections to the outside world. Such a server-initiated threat could pose much greater danger to a corporation.

The International Computer Security Association has reported that, unfortunately, as many as 80% of all corporate attacks are initiated from inside the company. Here corporate firewalls are of no use regardless of which control scheme is employed.

While client-server Trojan horses can pose a great threat to corporations, they will also pose a greater threat to end-users in the future. Today, most users have dynamically allocated IP addresses, making it more difficult for a casual hacker to consistently locate their machine and attack them. As cable modems, DSL and other constantly connected services become more available, users will increasingly have static IP addresses. Once a hacker succeeds in locating such an address and installing such a Trojan, they will be able to repeatedly attack the user's computer.

In general, we expect growth in this area. It is likely that a large number of proof-of-concept client-server Trojans will be built in the coming months and years. We expect that actual attacks will also increase.

Anti-Malware Technology Options

We now discuss possible mechanisms to combat each of the various malware threats we have described. In particular, we will cover mechanisms to address the following:

- Anti-virus threats
- Worm threats
- Stationary threats
- Anonymous threats

Future Anti-virus Options

Currently, signature (fingerprint) scanning and heuristics are the most commonly used technologies for detecting computer viruses. It is likely that fingerprinting will remain the cornerstone technology of any good AV product. It is one of the only techniques that can catch 100% of all infections of a given virus and also address any potential side effects of the virus. For a further discussion of signature-based virus detection see (Trilling & Nachenberg, 1998).

Heuristics will continue to improve but will never reach the complete reliability of exact fingerprinting techniques - there are simply too many ways for a virus to fool even the most clever heuristic scanner. Our in-house tests have shown that the heuristic technology in current anti-virus products can detect roughly 90-95% of all new macro viruses, between 50 to 80% of all new dos viruses and approximately 80% of all new boot viruses. These percentages will only go down with time as virus writers figure out ways to elude heuristic scanners.

Access control is a viable option for enterprises that have homogeneous configurations on all computers (network computers also fall into this category.) Access control software works by only allowing only pre-approved programs/macros to be used by the user. All other programs/macros are prohibited and alerts can be sent to the administrator. Such a solution works well in corporations with homogeneous desktops, but is too restrictive for many businesses.

Content-filtering personal and corporate firewalls will also become increasingly important for examining Java and other network-borne executable content. Currently, it is possible to download and use Java applets without these applets ever being copied to your hard drive. In this case, it is impossible for current on-access anti-virus products to detect these viruses (since they rely upon file creation in order to scan). However, a firewall that scans all content as it flows over the network connection could provide protection from these threats.

Future Anti-worm Options

As with computer viruses, fingerprinting and heuristic technologies can be used to detect computer worms. Since worms are self-replicating, fingerprinting will provide the most robust solution for customers. On the other hand, there are worms which can run successfully without being copied to the user's hard drive and detection of such threats is much more difficult using traditional anti-virus technology. It is an open problem how anti-worm software would go about scanning RAM and individual process-spaces. While signatures are great at detecting existing threats, it is possible that malicious individuals will create worm generators. In this case, we may see many new worms every day (hour/minute?) for which signature scanning will provide little if any protection. In this case, heuristics may be of help. However, even heuristics are powerless against a determined attacker.

As with computer viruses, personal and corporate firewalls will be essential in combating the threat from worms. Anti-virus on-access components may also be able to detect these worms as long as they are saved to the hard drive of the host system. Given the wide range of products and protocols which can host computer worms, a solution provider will have to build a very general content-filtering, proxying firewall product that can be easily programmed (via data additions) to support new threats. Unfortunately, such a firewall product may still have difficulty detecting worms in encrypted transmissions. Here, it will be up to the application vendor to implement appropriate security in the scripting software.

Perhaps the most effective way to prevent e-mail and other script-based worms would be for the producers of the targeted software to build in security and make it readily configurable. While anti-virus software can help to plug the holes, only system-wide security, directly integrated into the e-mail or script-driven system can provide 100% protection.

Stationary Threat Options

Java and ActiveX threats can be trivially detected using fingerprinting techniques. Some companies are also investigating heuristic detection algorithms, but it is unclear whether this type of technology could be effectively implemented in the future.

For the time being, there are three choices for combating stationary threats:

- certification and signing of "known good" Java/ActiveX content
- maintain a fingerprint database for "known bad" Java/ActiveX applets
- behavior blockers can be used to detect malicious code as it runs

The problem with "known good" certification is that it is only effective if a large number of Java/ActiveX developers sign up for it. Unless the great majority of stationary threats are properly certified, this method will not provide much protection to the user.

While virus-writer-generated malicious applets will probably never affect mainstream customers, it seems likely that professional scam-artists will use these technologies to steal money/information from consumers. Unfortunately, such threats will likely be impervious to signature scanning. If a scam-artist wants to steal someone's money, they will probably do so with a brand-new, home-brewed malicious applet rather than an old known applet found on a virus web site. Consequently, fingerprint scanning for these real threats, while effective in mal-ware detection reviews; will provide little utility to actual customers.

One way to protect against all types of stationary threats is through software, which allows a user's files to be accessed only by designated "trusted applications". Using such software, a user could, for example, designate that all .DOC files only be accessed by *Microsoft Word97*. If an ActiveX program running in a browser tried to open a .DOC file, the user would get an alert.

Heuristics could potentially be developed to detect all types of stationary threats. This would likely require further research since we have seen few commonalties among existing stationary threats.

Java specific solutions. As with computer viruses and worms, personal and corporate firewalls will be essential in combating the threat from malicious Java applets. Anti-virus on-access components may also be able to detect these applets if they are saved to the hard drive of the host system in an uncompressed form. However, there is no guarantee that any such content will be saved to the hard drive and subsequently scanned by the anti-virus software. This characteristic of Java makes personal firewalls or proxies a very attractive mechanism (or perhaps the only mechanism) for intercepting these applets and scanning or certifying them.

Anonymous Threat Options

Destructive people will continue to post malicious Trojan horses to USENET newsgroups (or equivalent forums), or send them to users via anonymous e-mail or other commonly used protocols such as the DCC protocol on IRC. If known threats are posted or e-mailed, traditional anti-virus, on-access scanners can detect these threats trivially as long as the anti-virus vendor keeps up with and monitors these threats. However, if an entirely new Trojan horse is posted or e-mailed to users, fingerprint-based scanning will prove wholly useless.

Heuristics can be used very effectively to detect certain specific classes of Trojan horses (such as AOL "password stealers", malicious programs which attempt to capture an AOL user's login password); unfortunately, one cannot rely upon such technology to detect anywhere near 100% of these threats.

There are several potential solutions to this problem:

1. Have e-mail programs filter all e-mail from non-verifiable (anonymous) accounts. Some Internet infrastructure must be set up to make this work well. Some companies might even want to completely forbid receipt of any anonymous e-mail inside their corporate mail system.
2. Have USENET newsreaders filter all file attachments from non-verifiable (anonymous) accounts.
3. Require the e-mail system to authenticate mail from all senders. This could offer protection against malicious attachments by making the sender completely accountable for their mail.
4. A program, which allows the designation of "trusted applications" (like *Norton Safe on the Web*), could help protect systems from some threats.
5. Service providers can perform this filtering for their users. For instance, AOL or MCI could filter all USENET newsgroup postings before they are visible by users. However, this would be an extremely expensive proposition.

While technology-based solutions can help to shield users from these anonymous threats, a good dose of common sense can also provide significant protection. Users need to be educated to exercise extreme care when executing any programs or using any documents from an unknown source.

Conclusions

In this paper we have provided a series of predictions concerning the future of various types of computer malware. We have also suggested a number of possible anti-malware technologies for combating each of these threats.

Current Threats

In the medium to near future, our expectation is that we will see a decrease in wild encounters of DOS file and boot record viruses. To support this hypothesis, we have provided a number of recent statistics on virus growth (in Appendix B).

We do expect to see continued growth of macro viruses (although this growth may slow) and we will likely see more macro-capable platforms attacked by virus writers. Again, this hypothesis has been supported by recent new viruses attacking *Microsoft Access* databases and *Microsoft PowerPoint* files.

We will also likely see more Windows viruses as virus writers become more proficient in this area. Again, we have seen a recent upsurge in the number of Windows viruses both in zoo collections and in the wild. The CIH and Marburg viruses are two high profile examples of such Windows threats.

Growth of Future Threats

In the future, the addition of new hardware and software platforms are likely to change the nature of the malicious code problem. Historically, new generations of hardware and software have provided opportunities for new threats. As examples, the emergence of DOS led to the creation of DOS viruses, adoption of Windows 95 has led to new Windows 95 viruses, and the adoption of macro languages into *Microsoft Office* products has in turn led to completely new types of wildlife.

Moving forward, as more home users have continuous (cable or DSL based) connections to the Internet, there will be a much greater opportunity for the spread of computer worms. Luckily, most of our current anti-virus technology can be adapted to provide protection against such threats, as long as they are copied to the user's hard drive. To protect against worms which reside only in memory (and don't get copied to the actual hard drive), personal firewalls could become much more of a necessity. In particular, as people become more and more dependent on the World Wide Web and new macro/script-enabled, Internet-enabled applications, the opportunity for worms to spread will only increase. Generalized content-filtering personal/corporate firewalls will be required to provide robust protection against these threats.

In the end, the best way to prevent e-mail and other script-based worms is for the developers of the vulnerable software to build in security and make it easily configurable. This would provide a clear protection benefit to customers.

The increased dependence on the Internet will also potentially change the landscape with respect to malicious ActiveX and Java. Two scenarios will likely play out. On the one hand, we expect to see little growth in malware posted to truly-stationary web-sites. Such web sites have a great deal of incentive to make sure such threats don't get posted because of their clear traceability. On the other hand, we do expect the number of attacks to increase on seemingly stationary, yet anonymous web sites such as GeoCities – since such sites are effectively anonymous, the risk of getting caught is much smaller. Unfortunately, common end users may not always realize that a given web site is actually anonymous. Greater education of the mainstream computer user will be the best mechanism for prevention in these cases. Web site certification programs and software-based-filtering may also provide appropriate security.

Regardless of exactly how these scenarios play out, we will probably see the growth of ActiveX/Java zoo viruses. Current anti-virus technologies

(fingerprinting) can be used to detect these zoo threats, but they will be largely useless against an adversary who wants to cause real damage or steal money/information from users. Java/ActiveX-filtering personal/corporate firewalls will be required to provide robust protection against these threats. Also, behavior blockers will become increasingly important; these products will likely evolve and become more usable in the coming months and years.

Finally, the threat from traditional Trojan horses will continue to grow. Since users can anonymously post Trojans on public communications channels (USENET newsgroups) or e-mail them to private not-so-savvy users, it is likely that we will see an increase in this type of threat. Current fingerprinting technology could help detect such malicious applications in some instances, however, new Trojan horses will be largely impervious to fingerprinting or even heuristic protection. Once again, behavior blockers will help to detect and prevent some attacks by Trojan horses. Content filtering (preventing anonymous users' posts) may also be one of the most effective measures against this type of threat. Any time we can make a user feel accountable (and locatable) for his/her actions, we will expect to see the associated malicious code threat diminish.

Possible Alternative Future Scenarios

Using history as a guide, we could also come up with an alternative scenario in which the threat of malware actually decreases. Just as some advances in hardware and software have caused an increase in new types of threats, other advances have actually slowed the growth of certain types of threats. For example, as mentioned above, the emergence of Windows has led to an increase in new Windows viruses. On the other hand, as Windows has been adopted more and more, the threat from the original DOS viruses has decreased markedly. As another example, *Microsoft Office97* appears to have contributed to slowing the growth of macro viruses.

It is entirely possible that future advances in technology will greatly lessen the broad threat of computer malware. We tend to support the theory that the general threat will not decrease but simply change in nature. The true answer remains to be seen.

Recommendations for Researchers

In addition to investigating the various anti-malware technologies suggested in this paper, we believe it would be of benefit for interested researchers to investigate the current sociological impacts of computers they relate to malware. In large part, computer malware now spreads because of changes in the way people use their computers, rather than for technological reasons. Some possible areas for research include:

- It would be of some interest to study how the various ways in which people exchange data contributes to the increase or decrease of malware threats. For example, when DOS was the predominant operating system, people often exchanged data through floppy disks. This helped increase the spreading capabilities of both DOS and boot viruses, through completely non-technical means (Nachenberg, 1997). More recently, people are much more likely to exchange *Word* and *Excel* documents – this sociological change has likely contributed to the spread of viruses which infect these types of documents. On the other hand, the fact that users seem less likely to exchange *Access* databases will potentially contribute to lowering the chances of such a virus ever spreading in the wild.

- It would also be interesting to investigate how the various ways in which people use the World Wide Web affects malware threats. Studying typical user habits in browsing the Web, communicating over IRC, posting to newsgroups, and many other Internet activities would provide an interesting viewpoint about how we might see malware spread in the future.

- Studying typical e-mail habits of users would provide interesting ideas on how threats might spread in the future through this medium.

There are likely numerous other interesting ideas to consider here. All of this work will contribute to our understanding of how the malware threat may evolve in the future.

Acknowledgments

The authors gratefully thank Sarah Gordon of IBM Research for lending her great insights and suggestions in numerous areas of this paper.

Appendix A – Table of Definitions

Anonymous threats: An anonymous threat is malware that is received via a vector such that the malware cannot be traced to its source. Examples of anonymous threats are any malware delivered via anonymous or forged posts to USENET newsgroups, via anonymous or forged e-mail, or on WWW sites on hosts such as GEOCITIES, where the owner of the site may be untraceable.

Behavior blocker: A behavior blocker is a type of anti-malware technology which installs itself into the operating environment targeted by the malware. Once integrated, the behavior blocker monitors all actions, as they occur (as opposed to in emulation) and reports suspicious or malicious actions to the user for proper resolution.

Browsing-serving asymmetry: This term describes the current state of Internet usage. Most users who browse the Internet and who are susceptible to an ActiveX/Java virus attack do not have their own Web server on the same machine which could be attacked (i.e. which has its own ActiveX/Java applets that could be infected by a viral applet). Likewise, Web servers are rarely used to surf the web, and consequently, will not encounter such ActiveX/Java viruses unless they are manually introduced by an attacker. In the future, if home users both surf and host WWW sites from their home computer, we expect the browsing-serving asymmetry to diminish.

Client-server threats: This describes a class of malware which allows an attacker to control the malicious code using a client-server model. Examples of such threats are the Back Orifice and NetBus Trojan horses.

Companion HTML virus: This describes a type of virus which adds applet-reference tags to HTML-based web pages in order to propagate viral infection. When such a virus (in the form of an applet) runs on a machine, it makes a copy of itself and then locates new HTML pages. It inserts references to the newly created copy of the applet into these HTML pages. Later, if another user browses the infected HTML pages, his browser will pull the companion malicious applets over and run them, where they will proceed to infect that user's HTML pages. Such a virus would be constrained by the browsing-serving asymmetry.

Data Import threats: This describes a type of malware which imports additional information as part of its payload. For instance, a Trojan horse could import additional modules from an external web site in order to do more damage to a victim network.

Fast viruses: Fast viruses spread quickly by installing themselves in the operating system. These viruses monitor various system services and infect files as they are accessed by the operating system, application programs, or the user.

Invisible Data Export: This describes a type of payload where the malware silently extracts and exports information from the victim computer. Contrast with visible data export.

Malware: A general term describing all types of malicious code, including viruses, worms, Trojan horses, etc.

Parasitic applet virus: This term describes an applet virus which spreads by attaching itself to other applets on the victim system in such a way that both the host applet and the virus can still function. Such a virus would be constrained by the browsing-serving asymmetry. Contrast with the companion HTML virus.

Parasitic HTML virus: This term describes a virus which spreads by inserting its entire viral body (usually composed of a script-based language, such as VB-script) into the body of an HTML page. Each time the HTML page is accessed on a new system, the virus activates and copies itself into other HTML pages found on the system.

Payload: This describes a malicious side-effect of a virus or or primary effect other malware. Common payloads are file deletion, data diddling and data export.

Stationary threats: Stationary threats are those which are obtained from a source which can be authenticated in some way (not necessarily strict authentication). Examples of stationary threats are any type of malware obtained from a typical web site, or from a CD or other distribution medium from a known entity such as a company or individual. The source of these threats is said to be stationary and cannot move or avoid tracing.

Visible Data Export: This term describes a type of payload where the malware prompts the user for information and then exports information from the victim computer. Contrast with invisible data export, where the attacking malware exports data without any interaction with the user.

Appendix B – Statistics On Virus Growth

These are statistics for the number of *incidents* per quarter per 1000 PCs, in an environment typical of well-protected Fortune 100 companies. An *incident* is a virus coming in from outside the organization. It might infect zero machines or many, but it counts as one incident. We believe these incident rates are roughly proportional to the number of infected systems worldwide. Statistics provided by (White, 1998). For some further statistics on virus growth, see (White, Kephart, & Chess, 1996).

Year	Quarter	DOS File	Boot	Macro	Total
1988	Q1	0.008	0.0	0.0	0.08
	Q2	0.012	0.008	0.0	0.02
	Q3	0.004	0.004	0.0	0.008
	Q4	0.004	0.004	0.0	0.008
1989	Q1	0.012	0.008	0.0	0.02
	Q2	0.004	0.008	0.0	0.012
	Q3	0.056	0.036	0.0	0.092
	Q4	0.092	0.14	0.0	0.232
1990	Q1	0.084	0.072	0.0	0.156
	Q2	0.116	0.108	0.0	0.224
	Q3	0.248	0.184	0.0	0.432
	Q4	0.22	0.172	0.0	0.392
1991	Q1	0.368	0.252	0.0	0.62
	Q2	0.42	0.38	0.0	0.8
	Q3	0.508	0.384	0.0	0.892
	Q4	0.58	0.536	0.0	1.116
1992	Q1	0.756	0.892	0.0	1.648
	Q2	0.288	0.284	0.0	0.572
	Q3	0.268	0.424	0.0	0.692
	Q4	0.328	0.472	0.0	0.8
1993	Q1	0.384	0.456	0.0	0.84
	Q2	0.244	0.408	0.0	0.652
	Q3	0.176	0.408	0.0	0.584
	Q4	0.228	0.6	0.0	0.828
1994	Q1	0.168	0.772	0.0	0.94
	Q2	0.068	0.596	0.0	0.664
	Q3	0.112	0.76	0.0	0.872
	Q4	0.124	0.856	0.0	0.98
1995	Q1	0.16	1.208	0.0	1.368
	Q2	0.152	1.284	0.0	1.436
	Q3	0.164	1.456	0.0	1.62
	Q4	0.164	1.288	0.096	1.548
1996	Q1	0.136	1.2	0.26	1.596
	Q2	0.132	0.892	0.368	1.392
	Q3	0.084	0.776	0.42	1.28
	Q4	0.124	0.552	0.468	1.144
1997	Q1	0.1	0.372	0.536	1.008
	Q2	0.076	0.260	0.548	0.884
	Q3	0.12	0.270	0.792	1.182

References

Bassham, L., & Polk, W.T. (1992). Threat assessment of malicious code and human computer threats, National Institute of Standards and Technology Report 4939 [On-line]. Available: http://bilbo.isu.edu/security/isl/threat.html.

Bontchev, V. (1996). Possible macro virus attacks and how to prevent them, Proceedings of the Sixth International Virus Bulletin Conference, 97-127.

Branigan, S. (1998). Risks with web programming technologies. Proceedings of the EICAR '98 Conference on Web-Safety.

Cohen, F. (1994). A short course on computer viruses (2nd edition). Wiley & Sons.

Chess, D. (1997). Mushy mutating macros, antivirus online, Volume 2, Issue 3 [On-line]. Available: http://www.av.ibm.com/2-3/Feature2.

DeGroot, P. (1998). New BIOS virus puts PC hardware at risk. Computing Canada, Aug 17, 1998, Willowdale.

Denning, P. (1990). Computers under attack: Intruders, worms, and viruses. Addison-Wesley.

Gordon, S. (1994). IRC and security -- can the two co-exist? Network Security Magazine, October, 1994, Elsevier Advanced Technology, Oxford, UK.

Gordon, S. (1995). Technologically enabled crime: shifting paradigms for the year 2000, Computers and Security Journal, October 1995.

Gordon S. (1998). The worm has turned. Virus Bulletin, August 1998, 10-12.

Gordon S., & Chess D. (1998). Where there's smoke, there's mirrors: the truth about Trojans on the internet. Proceedings of the Eighth International Virus Bulletin Conference, 183-204.

Gordon S., Ford R., & Wells J. (1997). Hoaxes and hype. Proceedings of the Seventh International Virus Bulletin Conference, 49-66.

Gryaznov D. (1998). Viruses in the Usenet. Proceedings of IVPC '98: Protecting the Workplace of the Future.

Muttick, I. (1998). Trojans – The new threat? Proceedings of IVPC '98: Protecting the Workplace of the Future.

Microsoft (1997). Office 97 delivers business value and reduced cost of ownership. [On-line]. Available: http://www.windows.com/office/office97/documents/tco/default.htm.

Morar J., & Chess D. (1998). Web browsers – threat or menace? Proceedings of the Eighth International Virus Bulletin Conference, 99-122.

Nachenberg, C. (1997). Anti-virus technology in the 21st century. Proceedings of the Seventh International Virus Bulletin Conference, 253-272.

Newhouse, E. (1988). The dirty dozen - an uploaded program alert list, Issue #8 [On-line]. Available: http://www.spiritweb.org/KeelyNet/PD/virus.asc.html.

Rosenberger, R. (1998). Misconceptions: Viruses on America Online. [On-line]. Available: http://kumite.com/myths/myths/aol.htm.

Spafford, E.H. (1989). The internet worm program: an analysis. ACM Computer Communication Review, 19(1), 17-57.

Szor, P. (1998). Attacks on Win32. Proceedings of the Eighth International Virus Bulletin Conference, 57-84.

Trilling, S., & Nachenberg, C. (1998). Incremental virus signature updates. Proceedings of the Eighth International Virus Bulletin Conference, 301-309.

White, S. (1997). Private communication.

White, S., Kephart J., & Chess, D. (1996). The changing ecology of computer viruses. Proceedings of the Sixth International Virus Bulletin Conference, 189-202.

Wired (1998). [On-line]. Available: http://www.wired.com/news/news/technology/story/15432.html.