Microsoft Security Intelligence Report

Volume 11

An in-depth perspective on software vulnerabilities and exploits, malicious code threats, and potentially unwanted software in the first half of 2011

ZEROING IN ON MALWARE PROPAGATION METHODS

Microsoft

Microsoft Security Intelligence Report

Authors

Joe Faulhaber
Microsoft Malware Protection Center

David Felstead
Bing

Paul Henry
Wadeware LLC

Jeff Jones
Microsoft Trustworthy Computing

Ellen Cram Kowalczyk
Microsoft Trustworthy Computing

Jimmy Kuo
Microsoft Malware Protection Center

John Lambert
Microsoft Security Engineering Center

Marc Lauricella
Microsoft Trustworthy Computing

Aaron Margosis
Microsoft Public Sector Services

Michelle Meyer
Microsoft Trustworthy Computing

Anurag Pandit
Windows Live Safety Platform

Anthony Penta
Windows Live Safety Platform

Dave Probert
Microsoft Security Engineering Center

Tim Rains
Microsoft Trustworthy Computing

Mark E. Russinovich
Microsoft Technical Fellow

Weijuan Shi
Windows Business Group

Adam Shostack
Microsoft Trustworthy Computing

Frank Simorjay
Microsoft Trustworthy Computing

Hemanth Srinivasan
Microsoft Malware Protection Center

Holly Stewart
Microsoft Malware Protection Center

Matt Thomlinson
Microsoft Security Response Center

Jeff Williams
Microsoft Malware Protection Center

Scott Wu
Microsoft Malware Protection Center

Terry Zink
Microsoft Forefront Online Protection for Exchange

Contributors

Roger Capriotti
Windows Live Safety Platform

Doug Cavit
Microsoft Trustworthy Computing

CSS Japan Security Response Team
Microsoft Japan

Dave Forstrom
Microsoft Trustworthy Computing

Eric Foster
Windows Live Safety Platform

Enrique Gonzalez
Microsoft Malware Protection Center

Heather Goudey
Microsoft Malware Protection Center

Vinny Gullotto
Microsoft Trustworthy Computing

Satomi Hayakawa
CSS Japan Security Response Team

Forbes Higman
Windows Live Safety Platform

Yuhui Huang
Microsoft Malware Protection Center

Aaron Hulett
Microsoft Malware Protection Center

Hilda Larina Ragragio
Microsoft Malware Protection Center

Eric Lawrence
Windows Live Safety Platform

Ken Malcolmson
Microsoft Trustworthy Computing

Takumi Onodera
Microsoft Premier Field Engineering, Japan

Daryl Pecelj
Microsoft IT Information Security and Risk Management

Kathy Phillips
Microsoft Legal and Corporate Affairs

Tareq Saade
Microsoft Malware Protection Center

Richard Saunders
Microsoft Trustworthy Computing

Jasmine Sesso
Microsoft Malware Protection Center

Norie Tamura
CSS Japan Security Response Team

Matt Thomlinson
Microsoft Trustworthy Computing

Patrik Vicol
Microsoft Malware Protection Center

Steve Wacker
Wadeware LLC

Table of Contents

Microsoft Security Intelligence Report, Volume 11

Volume 11 of the *Microsoft® Security Intelligence Report* (*SIRv11*) provides in-depth perspectives on software vulnerabilities and exploits, malicious code threats, and potentially unwanted software in Microsoft and third-party software. Microsoft developed these perspectives based on detailed trend analyses over the past several years, with a focus on the first half of 2011.

This document contains a featured intelligence section on malware propagation methods, including data, analysis, and a taxonomy for classifying the methods that threat families use to spread. The full report includes deep analysis of trends found in more than 100 countries/regions around the world and offers ways to manage risks to your organization, software, and people.

The full report, as well as previous volumes and related videos, can be downloaded from www.microsoft.com/sir.

Background

Among the array of technical and non-technical mechanisms that malicious parties have at their disposal for attacking computers and stealing data, the *zero-day vulnerability*—a software vulnerability that is successfully exploited before the software vendor has published a security update to address it—is especially significant for security professionals and attackers alike. Zero-day vulnerabilities—according to conventional wisdom, at least—cannot be effectively defended against, and can arise at any time, leaving even security-conscious IT administrators essentially at their mercy. Although technologies such as Data Execution Prevention (DEP) and Address Space Layout Randomization (ASLR) have been introduced to make it more difficult to reliably exploit software, and processes such as the Secure Development Lifecycle (SDL) have been shown to reduce the incidence of software vulnerabilities, zero-day vulnerabilities continue to capture the imagination.

The zero-day vulnerability is especially alarming for consumers and IT professionals, and for good reason—it combines fear of the unknown and an inability to fix the vulnerability, which leaves users and administrators feeling defenseless. It's no surprise that zero-day vulnerabilities often receive considerable coverage in the press when they arise, and can be treated with the utmost level of urgency by the affected vendor and the vendors' customers.

Despite this level of concern, there has been little measurement of the zero-day threat in the context of the broader threat landscape. This section of the *Microsoft Security Intelligence Report* presents such an analysis, along with details of the methodology used, a discussion of the insights gained from it, and some information about what's been done with those insights.

This analysis approaches its subject in two ways. First, it establishes a method to estimate how malware propagates, including the use of zero-day exploits. Second, it measures the amount of zero-day exploitation in comparison with overall vulnerability exploitation. In other words, what are the relative proportions of exploitation before and after the update?

This analysis was undertaken for a number of reasons. Microsoft is always seeking better statistics about the frequency of zero-day exploitation and the risk

customers face from it. Also, Microsoft frequently fields questions about zero-day vulnerabilities from a variety of interested parties, ranging from journalists to IT security professionals. It is important to provide timely and accurate answers for such questions, but also help put them in perspective relative to other threats in the greater security landscape. In a more general sense, it serves everyone—IT and security professionals as well as consumers—to have realistic models of the way malware spreads in today's world. At a time when effective cooperation and coordination of security efforts across corporate and political borders is as important as it has ever been, it is only through an accurate shared understanding of the threats all users face that IT and security pros can create the most effective defense.

One important goal of this analysis is to provide security professionals with information they can use to prioritize their concerns and effectively manage risks. Like everyone else, IT departments face constraints of time, budget, personnel, and resources when planning and performing their work. Having accurate, up-to-date information about the threat landscape enables security professionals to effectively prioritize their defenses and help keep their networks, software, and people safe.

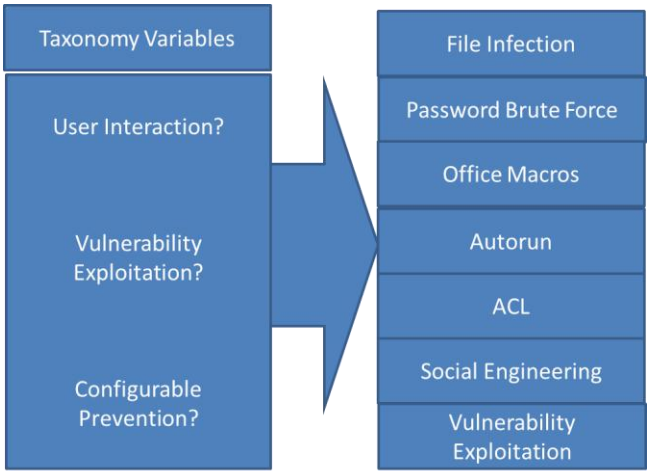
Analysis and Results

To better understand the landscape, Microsoft researchers have drawn on current information about trends and developments in malware creation and distribution to develop a new taxonomy for classifying malware according to the methods it uses to spread. Applying this taxonomy to telemetry data generated by security products has provided insights into the ways attackers distribute malware.

A New Method for Classifying Malware Propagation

The analysis presented here is in part an effort to start a conversation within the industry about the current state of malware analysis and classification. Many of the de facto standards that security professionals use were originally formulated when the threat landscape was very different than it is today. These standards were created when widespread public use of the Internet was nonexistent or very limited, and before malware development and propagation were the domain of professional criminals looking for illegitimate profits. Many of these standards and beliefs evolved chaotically over a period of years, and in some cases terms were never especially well defined. By adding new ways to classify malware and understand how exploitation is measured, security professionals can improve the ways they think and communicate about the threats that modern computer users face. This analysis is not a call to throw away current approaches, but rather a new lens that has been shown to be helpful.

Figure 1. Classifying malware according to propagation methods



The framework sketched in Figure 1 that classifies malware families by the methods—both technical and non-technical—that they use to propagate was developed as part of this analysis. In this context, propagation refers to the crucial moment when the attacker is first running software on a computer. "Insights," beginning on page 16, provides an overview of this taxonomy; an in-depth explanation begins on page 21.

As with any taxonomy, adaptation is a natural progression. As a lesson learned from past malware categorization, this taxonomy should not be considered definitive. On the contrary, the researchers are enthusiastic about presenting its current form and look forward to the community dialogue that is sure to result as it evolves.

Data Used

To apply this taxonomy to infection data, Microsoft researchers analyzed infections reported by the Microsoft Malicious Software Removal Tool (MSRT) during the first half of 2011. The MSRT is a free tool that Microsoft designed to help identify and remove selected prevalent malware families from Windows–based computers. A new version of the MSRT is released each month and distributed through Windows® Update, Microsoft Update, and the Microsoft Download Center.

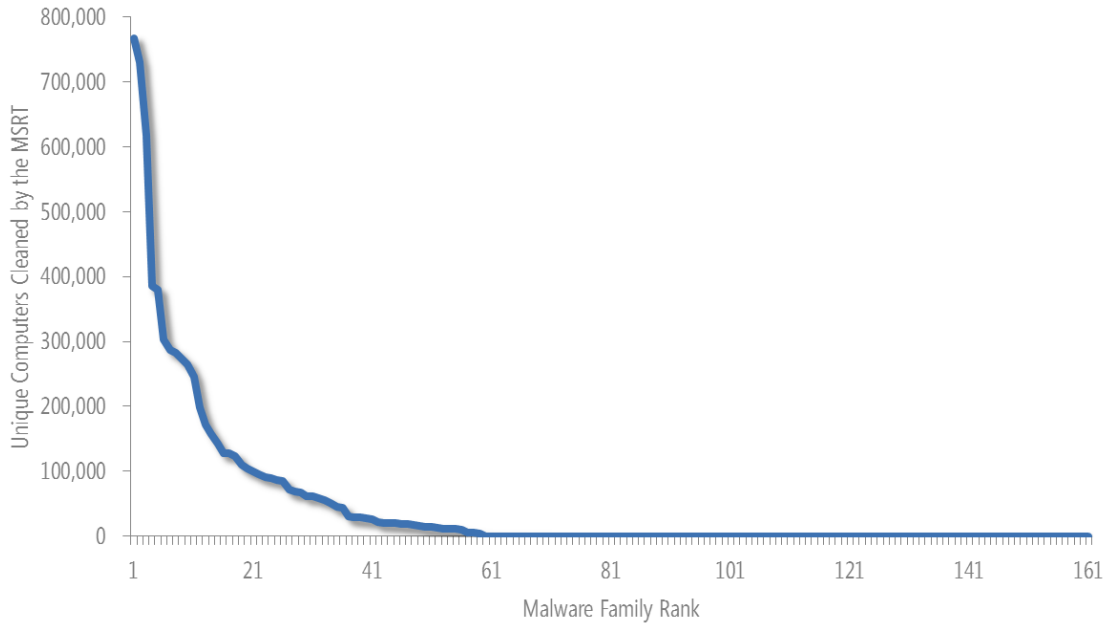
The MSRT was selected as the data source for this exercise for several reasons:

- The MSRT runs on more than 600 million individual computers around the world each month.
- The MSRT specifically targets malware families that present a severe risk to users or are particularly prevalent.
- MSRT data represents infected computers (as opposed to infection attempts that were blocked by real-time protection products).
- Installations of the MSRT are strongly correlated with usage of Windows Update and Microsoft Update, the tool's primary distribution mechanisms, which helps provide a reasonably accurate picture of the risks faced by computers that likely apply regular security updates.

Analytic Methods

Malware infections tend to resemble a power law distribution, as shown in Figure 2, in which a few dozen malware families account for most infections and a "long tail" consisting of a large number of less common families account for the rest.

Figure 2. Malware families detected by the MSRT, ranked by the number of computers each family was removed from in the second quarter of 2011 ("2Q11")



To allow for a thorough analysis of infection methods for a significant portion of the malware landscape, this analysis focuses on the 27 malware families detected most often by the MSRT in the first half of 2011, which together accounted for a

majority of total MSRT detections. [1] To classify these malware families for analysis, the researchers investigated the mechanisms by which each of the families has been documented to spread, using information from the MMPC malware encyclopedia as well as other sources. Only mechanisms used actively by each family to spread were considered; The mechanisms used by these families were grouped into nine separate categories. (See "Insights" beginning on page 16 for more information about this classification scheme.)

Many families use multiple mechanisms to propagate. When malware is detected on a computer, the actual method of infection is very difficult to determine without performing forensic work on each computer. Therefore, to analyze infections on hundreds of thousands of computers, some assumptions are necessary.

To compensate for the difficulty in determining the exact propagation mechanism used in each case, an "equal buckets" approach was used in which detections of these families were allocated equally among each category in which they were known to spread. For example, Win32/Conficker spreads by exploiting a vulnerability (CVE-2008-4250, addressed by Security Bulletin MS08-067), by taking advantage of AutoRun on both mapped drives and removable ones, and by using a password dictionary. Using this approach, 100 Conficker infections is translated into 25 vulnerability-related propagations and 75 in feature abuse (25 each for AutoRun USB, AutoRun network, and password brute force activity).

Families that were determined to spread via exploits were classified according to the age of the security update addressing the vulnerability at the time of analysis:

- **Zero-day**. The exploit is known to have existed in the wild before the vendor could publish a security update to address the related vulnerability. If the exploit was zero-day at any time during the month-long period preceding the release of the MSRT version that detected it, it is considered a zero-day exploit for the purposes of this analysis.
- **Update Available**. The security update that addresses the vulnerability was first issued less than a year before the recorded detection.
- **Update Long Available**. The security update that addresses the vulnerability was first issued more than a year before the recorded detection.

[1]The analysis included all malware families detected on computers at least 25,000 times. The families listed here accounted for 83 percent of all MSRT detections for the 6-month period.

For example, security bulletin MS08-067, which addressed the vulnerability exploited by Conficker, was released in October 2008, so Conficker is now listed in the "Update Long Available" category.

Figure 3 lists the malware families included in this analysis and shows how they were classified.

Figure 3. Some of the top malware families detected by the MSRT in 1H11 and their propagation methods

Family	Exploit: Zero-day	Exploit: Update Avail.	Exploit: Update Long Avail.	AutoRun (Net.)	AutoRun (USB)	Office Macro	Passwd. Brute Force	User Inter-action	File Infector
Win32/Alureon		•						•	
Win32/Bancos								•	
Win32/Bredolab			•						
Win32/Brontok					•			•	
Win32/Bubnix								•	
Win32/Conficker			•	•	•		•		
Win32/Cutwail								•	
Win32/Cycbot			•					•	
Win32/FakeRean								•	
Win32/FakeSpypro								•	
Win32/FakeXPA								•	
Win32/Frethog				•				•	
Win32/Hamweq					•				
Win32/Jeefo									•
Win32/Lethic								•	
Win32/Parite									•
Win32/Pushbot			•		•			•	
Win32/Ramnit				•	•	•			•
Win32/Randex							•		
Win32/Renocide				•	•			•	

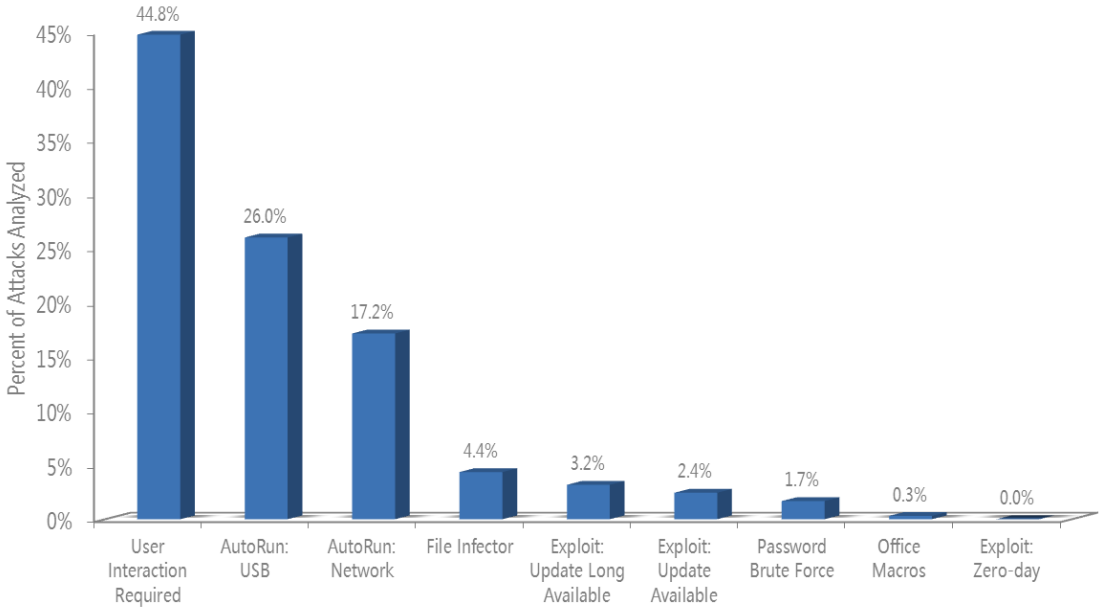
Figure 3 (continued). Some of the top malware families detected by the MSRT in 1H11 and their propagation methods

Family	Exploit: Zero-day	Exploit: Update Avail.	Exploit: Update Long Avail.	AutoRun (Net.)	AutoRun (USB)	Office Macro	Passwd. Brute Force	User Inter-action	File Infector
Win32/Renos								•	
Win32/Rimecud				•	•			•	
Win32/Sality				•					•
Win32/Taterf				•	•				
Win32/Vobfus			•	•	•				
Win32/Yimfoca								•	
Win32/Zbot		•	•					•	

Results

Figure 4 shows the results of this analysis.

Figure 4. Malware detected by the MSRT in 1H11, by means of propagation ability



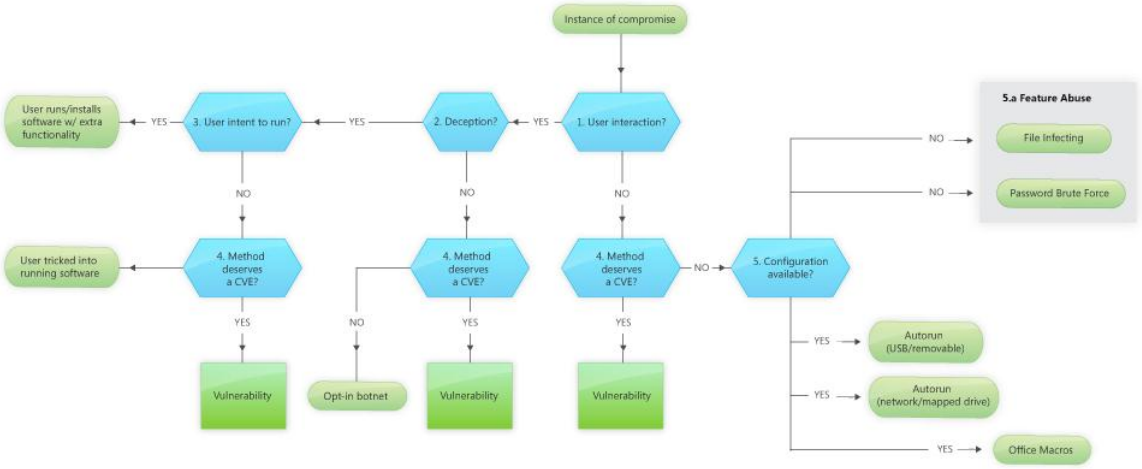
- Threats that are documented as relying on user interaction to spread account for 45 percent of attacks analyzed.

- More than a third of the detections that were analyzed were caused by malicious software that misused the AutoRun feature in Windows. Analyzed threats were split between USB AutoRun threats (26 percent of the total) and network volume AutoRun threats (17 percent).
- About 6 percent of the MSRT detections analyzed were likely caused by exploits. Of these, the majority had had security updates available for more than a year at the time of detection (classified as "Update Long Available"), with the remainder involving exploits for vulnerabilities for which security updates had been released less than a year before detection (classified as "Update Available").
- File infectors, or viruses, accounted for 4 percent of detections.
- The password brute force and Office macro behaviors were each identified in just one of the families examined in this exercise, and accounted for 2 percent and 0.3 percent of the total, respectively.

Insights

The taxonomy introduced on page 9, codenamed "Broad Street," organizes the categories used in this exercise according to propagation behavior, as shown in Figure 5.

Figure 5. The project Broad Street taxonomy, version 2.6



User Interaction

The first distinction shown in Figure 5 is between threats that require user interaction to compromise a computer and threats that do not. Threats that require user interaction can be further subdivided according to whether they require deception, and whether they require the user to make an explicit decision to install software. (An example of a mechanism that requires user interaction but not deception would be an opt-in botnet, such as Java/Loic; see page 22 for more information.)

A typical example of a user interaction that isn't considered an installation decision would be a user following a hyperlink on a webpage or in an email message that leads to a page that attempts to use browser vulnerabilities to install malware.

Feature Abuse

Among threats that don't require user interaction, another fundamental distinction exists between threats that exploit vulnerabilities in software and threats that don't. The latter group includes file infecting viruses and threats that misuse legitimate features or functionality for malicious purposes.

Detections of threats that abuse features—including AutoRun threats, malicious scripts and macros, viruses, and password cracking—are increasing; the project Broad Street analysis attributes almost two-thirds of MSRT detections in 1H11 to a variety of feature abuses. This increase may be caused in part by an increase in the detection of threats that take advantage of the AutoRun feature in Windows. These threats spread by creating or modifying the autorun.inf file on mounted volumes in an effort to cause the computer to execute a malicious program whenever the volume is connected. Some of these threat families display an extra "Open folder to view files" entry in the AutoPlay dialog that appears by default in some versions of Windows when a network or removable volume is connected. Selecting this option would install the malware.

Microsoft introduced a change in the way the AutoRun feature works in Windows 7 and Windows Server® 2008 R2 in an effort to help protect users from such threats. In these versions of Windows, the AutoRun task is disabled for all volumes except optical drives such as CD-ROM and DVD-ROM drives, which have historically not been used to transmit AutoRun malware. In November 2009, Microsoft published a set of updates to the Microsoft Download Center that back-ported this change to Windows XP, Windows Server 2003, Windows Vista®, and

Windows Server 2008. As a result of data obtained through this exercise, these updates have been published as important updates through the Windows Update and Microsoft Update services since February 2011, and have been installed by more than 500 million computers since then.

The publication of these updates on Windows Update has had a significant effect on the ability of malware to use AutoRun to replicate. Between January and May of 2011, the MSRT reported decreases in detections of AutoRun-abusing families of between 62 and 82 percent on supported versions of Windows XP and Windows Vista. For more information, see the entry "Autorun-abusing malware (Where are they now?)" (June 14, 2011) in the Microsoft Malware Protection Center (MMPC) blog at blogs.technet.com/mmpc.

Exploit Age

When compared to the other categories of threats identified for the project Broad Street analysis, exploits are relatively rare, and exploits that target recently disclosed vulnerabilities are rarer still. Of the attacks attributed to exploits in the 1H11 MSRT data, less than half of them targeted vulnerabilities disclosed within the previous year, and none targeted vulnerabilities that were zero-day during the first half of 2011. (Because Microsoft usually releases security updates and the MSRT at the same time, the analysis considers a vulnerability zero-day for the entire month that an update is released. For example, if a malware family only uses a particular exploit in January, and Microsoft releases an update to fix the vulnerability in January, all February cleans of that family are counted as zero-day. This choice was made to avoid under-counting zero-days.)

Zero-Day Exploits: A Supplemental Analysis

However, if one considers exploits that are not associated with families detected by the MSRT, a small number of vulnerabilities did have zero-day exploits in 1H11. To assess the impact of these zero-day exploits compared to exploits of vulnerabilities for which security updates were available, the researchers conducted a supplemental analysis that used data from all Microsoft security products. (See Appendix B in the full report for more information about the products and services that provided data for this report.)

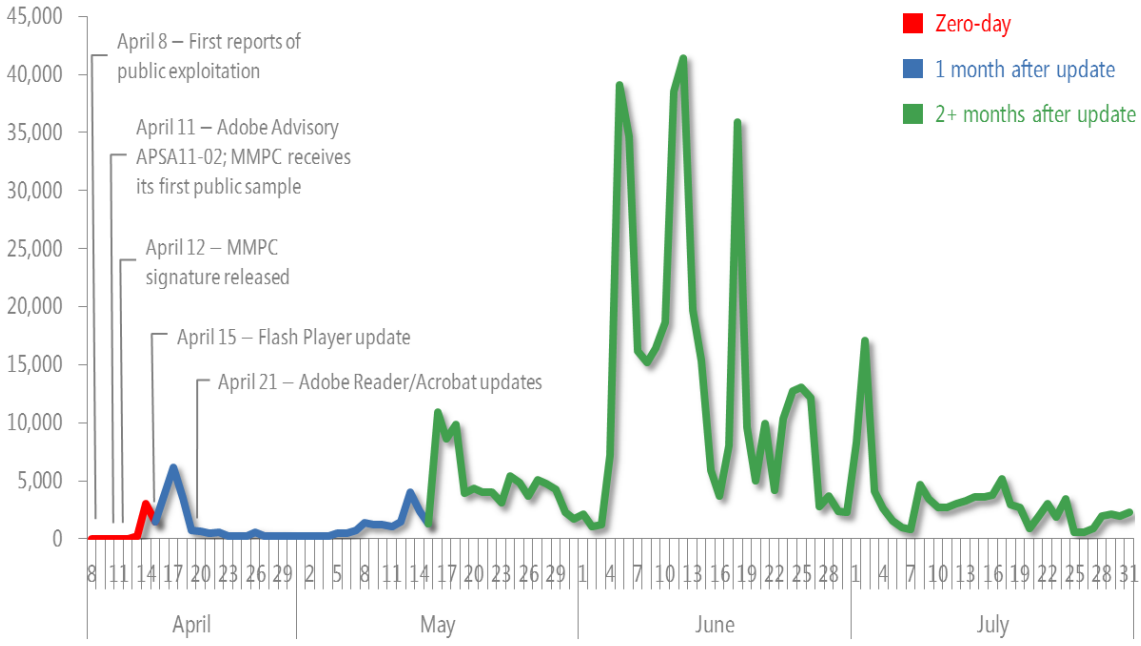
The MMPC tracks vulnerability exploitation attempts using more than 3,000 signatures. Although some generic signatures may detect a zero-day exploit before the vulnerability has been disclosed, in most cases a signature update is required to detect or to single out one vulnerability exploit from another. Given these

constraints, some small-scale, targeted attacks using zero-day exploits may escape detection briefly, and such attacks would not be reflected in the data presented here. In general, though, when attacks involving an undisclosed vulnerability occur in significant volume, they are noticed quickly; security vendors respond by providing detection signatures and protection, and the affected software vendor publishes security updates to address the vulnerability.

In this supplemental analysis, zero-day exploitation accounted for about 0.12 percent of all exploit activity in 1H11, reaching a peak of 0.37 percent in June. Two vulnerabilities accounted for the bulk of zero-day exploit activity: CVE-2011-0611, disclosed in April 2011, and CVE-2011-2110, disclosed in June 2011. Both vulnerabilities affect Adobe Flash Player. (See the full report for more information about these two exploits.)

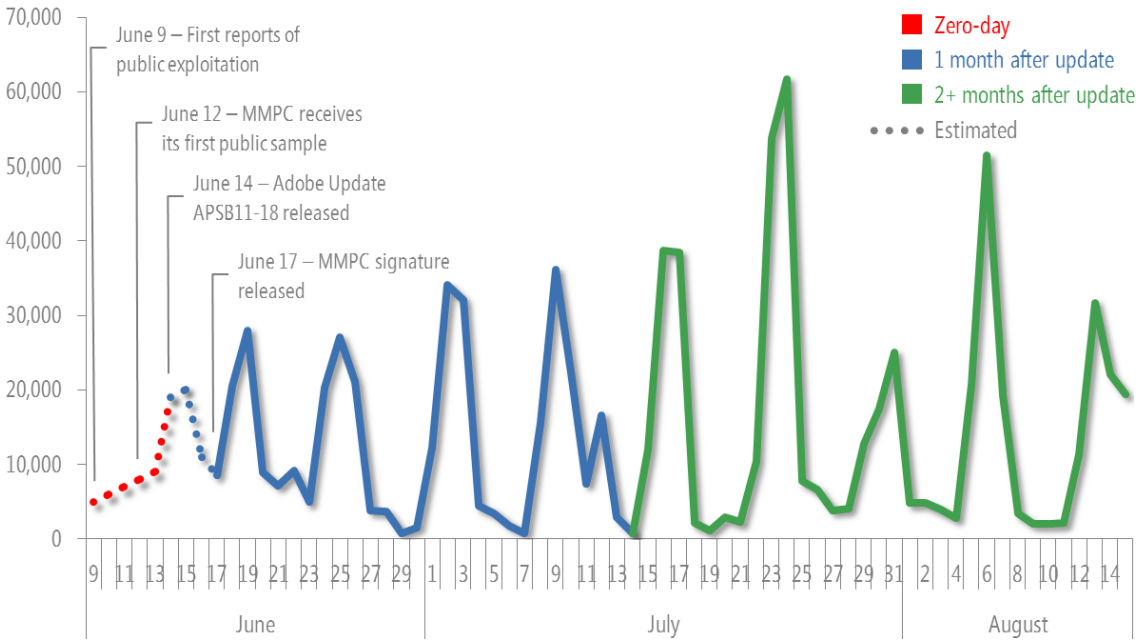
In the case of CVE-2011-0611, Adobe Systems released Security Bulletin APSB11-07 for Adobe Flash Player on April 15, 2011, less than a week after the first reports of public exploitation. Security Bulletin APSB11-08 for Adobe Reader and Adobe Acrobat was released the following week, on April 21, to address exploits involving malicious Flash files embedded in PDF documents. (Exploits using the PDF vector were only detected in a handful of samples before April 21, and the first real surge of activity using PDFs did not occur until May 13, a few weeks after the update had been released.)

Figure 6. Detections of exploits targeting CVE-2011-0611, April–July, 2011



For CVE-2011-2110, Adobe released an update on June 14, 2011 in response to to targeted attacks that were reported to have been occurring since around June 9. The MMPC received its first exploit sample on June 12, two days before the release of the update. Microsoft released a generic signature, Exploit:SWF/ShellCode.A (subsequently redesignated Exploit:SWF/CVE-2011-2110.A), on June 17 to detect and remove the exploit.

Figure 7. Detections of exploits targeting CVE-2011-2110, June–August, 2011



In total, an estimated 0.04 percent of the CVE-2011-0611 attacks and 8.9 percent of the CVE-2011-2110 attacks came before the applicable security updates were released.

Analysis Details

The Project Broad Street Taxonomy

The following analysis uses a new taxonomy that was designed to classify propagation vectors. To create the taxonomy, researchers examined the documented propagation methods used by each of the malware families studied in the analysis. Successful malware propagation reflects a failure of the defensive systems that are in place to prevent attacks; consequently, focusing on means of propagation can help security professionals hone their defenses.[2]

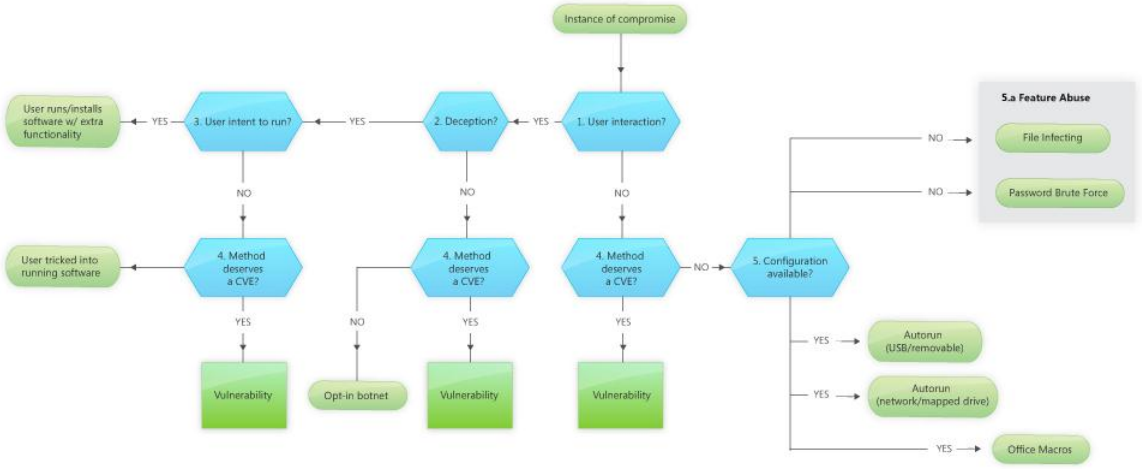
The taxonomy focuses on built-in malware propagation methods. The goal is to assess what percentage of malware succeeds by taking advantage of each vector to provide actionable data to the industry about what can be done to make it harder for malware to succeed using that vector in the future.

Using the Taxonomy

Figure 8 is a reprint of the project Broad Street taxonomy, first shown in Figure 5. The question boxes (diamonds) are numbered to make it easier to reference them in the text.

[2] This analysis intentionally focuses on *propagation* from computer to computer, rather than on malware *distribution*. File infection propagation from computer to computer occurs via shared or removable drives.

Figure 8. The project Broad Street taxonomy



User interaction required? (question 1) The first question the taxonomy poses is whether the user has to perform some action that results in a compromise. If the answer is Yes, the flow proceeds to question 2; if No, question 2 is skipped and the flow proceeds to question 4.

Deception? (2) The second question is one of deception. Deception often entails convincing someone that they will get some benefit from the action, or suffer some penalty if they don't do it, using any of a variety of social engineering techniques. Examples of deception might include a website telling people that they need to install a codec to watch a video, or an email message that claims to be from the tax authorities.

In some cases, users choose to install software that is designed to perform malicious actions. This classification includes scenarios involving **opt-in botnets**, in which the user chooses to give partial control of the computer to another party, who intends to use it to conduct activities such as denial-of-service (DoS) attacks. This category includes Flooder:Java/Loic, an open-source network attack tool designed to perform DoS attacks. Decentralized groups of protesters or vigilantes sometimes distribute software such as Java/Loic to users who wish to participate in DoS attacks on specific political or commercial targets.

If propagation requires deceiving the user, the flow proceeds to question 3. If it doesn't, question 3 is skipped and the flow proceeds to question 4.

User intent to run? (3) If user interaction is required, is the user aware that the action they are taking will involve running or installing software? If the answer is Yes, the flow terminates in an endpoint:

- **User runs/installs software with extra functionality.** The user runs the software, which performs malicious actions in addition to or instead of the software's desired function. A significant overlap exists between this kind of threat and the traditional definitions of "Trojan Horse" software. The analogy with the Trojan Horse from Greek mythology refers to the way many trojans gain access to victims' computers by masquerading as something innocuous: malicious executables represented as installers for legitimate security programs, for example, or disguised as documents for common desktop applications. In modern usage, however, most security vendors define *trojan* simply as a program that is unable to spread of its own accord. To avoid confusion, therefore, this analysis avoids use of the "trojan" or "Trojan Horse" labels.

If the answer is No, the flow proceeds to question 4.

Method deserves a CVE? (4) This question is the same for all three branches of the process flow, and determines whether or not a vulnerability is involved. Because the term "vulnerability" can be open to interpretation, the question asks whether the method used to install the software deserves to be documented in the Common Vulnerabilities and Exposures list (CVE), a standardized repository of vulnerability information maintained at cve.mitre.org. ("Deserves" is used for situations in which the method meets the CVE criteria but has not yet been assigned a CVE number, as with a previously undisclosed vulnerability.)

If the answer is Yes, the flow continues in the vulnerability subprocess, which is documented on page 24.

If the answer is No and user interaction is required to install or run the software, the flow terminates in one of two endpoints, depending on whether deception is involved:

- **User tricked into running software.** This result indicates a "false badging," such as a malicious executable named "document.pdf.exe" with an icon similar or identical to the one used for PDF files in Adobe Reader. The user launches the executable, believing it to be an ordinary PDF file, and it installs malware or takes other malicious actions.
- **Opt-in botnet**. This result indicates that the user has voluntarily installed botnet software.

If the answer is No and user interaction is not required to install or run the software, the flow proceeds to question 5.

Configuration available? (5) Can the attack vector be eliminated through configuration changes, or does it involve intrinsic product features that cannot be disabled through configuration? Configuration options would include things like turning the firewall off, and using a registry change to disable the AutoRun feature.

If the answer is Yes—in other words, if the attack vector can be eliminated through configuration changes—the flow terminates in one of three endpoints:

- **AutoRun (USB/removable).** The threat takes advantage of the AutoRun feature in Windows to propagate on USB storage devices and other removable volumes, as described on page 17.
- **AutoRun (network/mapped drive).** The threat takes advantage of the AutoRun feature to propagate via network volumes mapped to drive letters.
- **Office Macros.** The threat propagates on new computers when users open Microsoft Office documents with malicious Visual Basic® for Applications (VBA) macros.

Feature abuse: (5a) If the answer is No—in other words, if the attack vector uses product features that cannot be turned off via a configuration option—it is considered feature abuse, and the flow terminates in one of three endpoints:

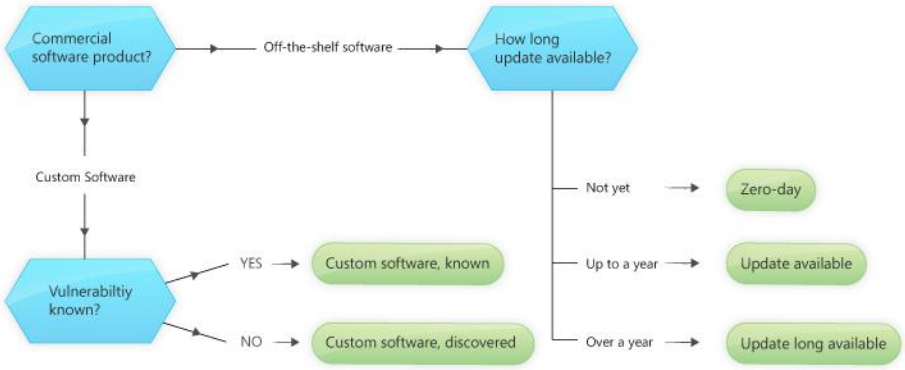
- **File infecting viruses.** The threat spreads by modifying files, often with .exe or .scr extensions, by rewriting or overwriting some code segments. To spread between computers, the virus writes to network drives or removable drives.
- **Password brute force.** The threat spreads by attempting brute force password attacks on available volumes to obtain Write or Execute permissions, as with the `net use` command.

A note on "other": All taxonomies include either implied or explicit "other" or "unclassified" elements. For simplicity, these are not shown, but one could imagine classifying a threat as "other feature abuse," "other configuration issue," or "other ways a user is deceived."

Vulnerability Subprocess

If the answer to question 4 is Yes—if the method used to install the software has or deserves a CVE entry—the attack is considered an exploit, and the process flow continues in a subprocess, shown in extended form in Figure 9.

Figure 9. The extended vulnerability subprocess of the project Broad Street taxonomy



The first question in the subprocess asks whether the vulnerability affects commercial software or custom software. Vulnerabilities are not unique to commercial software, and other exploit analyses have found that vulnerabilities in custom software, such as website code, account for a significant percentage of exploitation. Exploits of custom software are classified according to whether the vulnerability involved was known to the developers before the attack, or was discovered by the attacker.[3]

If the vulnerability affects commercial software, the flow terminates in one of three endpoints, according to the amount of time that has elapsed since the release of a security update addressing the vulnerability:

- **Zero-day.** The vendor had not released a security update to address the vulnerability at the time of the attack.
- **Update available.** The vendor released a security update that addressed the vulnerability less than a year before the attack.
- **Update long available.** The vendor released a security update that addressed the vulnerability more than a year before the attack.

Methodology Details

The project Broad Street analysis focuses on successful malware installs. Many other analyses are focused on attacks. Sometimes, attacks that are seen more often will seem more successful, but that may or may not be accurate.

[3] The researchers would like to thank the Verizon RISK team for pointing out this extension to the approach.

One might object that only examining computers that are regularly updated would naturally tend to reduce exploit detections of all kinds. In fact, that is a key point: Regularly installing security updates is one of the most fundamental steps that IT departments and individual users can take to reduce their risk from malicious software. IT departments and individual users who are concerned about security—a group that is presumed to include most of those reading this report—are likely to regularly install security updates from Microsoft and other vendors, and to face less risk from older exploits as a result. The project Broad Street analysis, therefore, examines the residual risk faced by hundreds of millions of computers that are already being kept up to date.

Although the MSRT only detects a subset of the malware families recognized by Microsoft antimalware solutions, malware that propagates via exploits, such as "traditional" worms, do not seem to be underrepresented in this subset. Most of the prevalent malware families not detected by the MSRT are adware and other potentially unwanted software families, as shown in Figure 10.

Figure 10. The most commonly detected malware families not detected by the MSRT in 2Q11

	Family	*Security Intelligence Report* Category
1	Win32/Hotbar	Adware
2	JS/Pornpop	Adware
3	Win32/Autorun	Worms
4	Win32/OpenCandy	Adware
5	Win32/ShopperReports	Adware
6	Win32/Keygen	Miscellaneous Potentially Unwanted Software
7	Win32/ClickPotato	Adware
8	Win32/Zwangi	Miscellaneous Potentially Unwanted Software
9	Win32/Obfuscator	Miscellaneous Potentially Unwanted Software
10	Win32/OfferBox	Adware

Although malware can be distributed by vectors that are extrinsic to the malware, this analysis focuses on the documented ways in which specific forms of malware are installed.

Other classifications of malware

Other malware classification systems use some terms that this malware taxonomy does not, including:

- **Drive-by download**. This term refers to exploits that target vulnerabilities in web browsers, which can lead to computers becoming compromised if users simply browse to the malicious site. The project Broad Street taxonomy presented here does not use this term; it classifies all exploits according to whether a security update that addresses the vulnerability is available and how long ago it was released.
- **Exploit kit**. Exploit kits are collections of exploits that usually target web browsers and plugins in the form of packages that can be deployed on a web server. Project Broad Street sees exploit kits as collections of attacks that exploit vulnerabilities.
- **Pay per install**. This term is used to identify malware that is distributed by other malware as part of an affiliate scheme. This taxonomy is focused on the initial compromise, and does not take economic arrangements into consideration.
- **Bluetooth**. Some security software vendors highlight malware that uses Bluetooth wireless connections to propagate. Analysis of Bluetooth as a propagation mechanism is out of scope for this project, but it seems likely that use of this vector would be classified as either social engineering or exploits, or potentially a new part of the taxonomy.

Conclusion

The intent of this analysis is not to downplay the risks posed by zero-day vulnerabilities, or to encourage software vendors and others to "let their guard down" against them. Rather, it is to provide security professionals with information they can use to prioritize their concerns and respond effectively to threats. Like everyone else, IT departments face constraints of resources such as time, budget, and personnel when planning and performing their work. Having accurate, up-to-date information about the threat landscape is vitally important to security professionals who seek to effectively prioritize their defenses and keep their organizations safe.

Call to Action

- Security professionals, including antivirus/antimalware vendors, penetration testers, incident response analysts, and others can use the project Broad Street taxonomy to talk more clearly about how computers are compromised.

- Test and deploy security updates from all software vendors as quickly as possible. See the Microsoft Security Update Guide, available from the Microsoft Download Center, for guidance and recommendations.

- Ensure that your development team is using the Security Development Lifecycle (SDL) (www.microsoft.com/sdl) or a similar software security assurance process. Using such a methodology can help reduce the number of vulnerabilities in software and help manage vulnerabilities that might be found after deployment.

- Build your defenses against social engineering.

Advice to IT Professionals on Social Engineering

IT professionals are accustomed to thinking about the technical aspects of security. However, as this report has shown, the human element—the techniques that attackers use to trick typical users into helping them—has become just as important for attackers as the technical element, if not more so. By implementing effective technical safeguards, programs, and processes designed to defend against social engineering, you can help your users avoid being taken advantage of by attackers. You can even enlist them as some of your most valuable assets in the fight against security threats.

Organizations

Your network provides the underlying infrastructure in which your applications are deployed. It is important to secure your network as a vital component of your defense-in-depth strategy.

Minimize and Monitor Your Attack Surface

- Limit the number of powerful user accounts in your organization and the level of access they have, because this will help limit the harm a successful social engineering attack can cause.
- Regularly audit your powerful user accounts. Provide them only to those who must have access, and to the specific resources to which they need access.
- Ensure these user accounts have strong authentication (strong passwords and/or two-factor authentication).
- Regularly audit attempts to access sensitive company information—both failed and successful attempts.

Create a Social Engineering Incident Response Plan

- Put in place systems to detect and investigate potential social engineering attacks.
- Create a virtual team to respond to attacks, and consider the following areas:
  - What was or is being attacked, and how.
  - Which resources are threatened or compromised.
  - How to shut down an ongoing attack with the least amount of disruption to the business.
  - How to recover from the attack.
  - How to implement protections against similar attacks.

Create a Plan For Addressing Social Engineering In Your Organization

- Determine which threats have the greatest potential:
  - Determine the resources attackers are most likely to pursue and those most critical to the business.
  - Analyze attacks that have occurred against your organization and those like it.
  - Determine where technology, policies, or company culture creates "soft spots" that are especially vulnerable to social engineering attacks.
- Determine how to address these vulnerable areas:
  - Determine where technology or processes can be altered to reduce or eliminate the threats.
  - Create policies that make it easy for people to perform secure actions without feeling rude.
  - Create awareness training for those vulnerable areas that are most critical, and where technology, process, and policy may not address

the problem sufficiently. Ensure that your guidance fits well within your organizational culture; it should be:

- **Realistic.** Guidance should enable typical people to accomplish their goals without inconveniencing them.
- **Durable.** Guidance should remain true and relevant, and not be easy for an attacker to use against your people.
- **Memorable.** Guidance should stick with people, and should be easy to recall when necessary.
- **Proven Effective.** Guidance should be tested and shown to actually help prevent social engineering attacks.
- **Concise and Consistent.** The amount of guidance you provide should be minimal, be stated simply, and be consistent within all the contexts in which you provide it.

- More details on how to create a process around social engineering prevention and response can be found in "How to Protect Insiders from Social Engineering Threats" on Microsoft TechNet.

Software

Many social engineering attacks involve tricking the user into opening a malicious file or browsing to a malicious website that takes advantage of a code vulnerability. As the data presented in this report shows, in many cases these attacks use vulnerabilities for which a security update has already been made available—sometimes quite a while ago. One of the most important things you can do to blunt social engineering attacks is to keep software as up-to-date as possible. The Microsoft Security Update Guide, Second Edition, available from the Microsoft Download Center, provides guidance on how to deliver updates to your users in a timely and effective manner, in consideration of all of the other challenges in your IT environment.

People

Information security awareness and training are critical for any organization's information security strategy and for supporting security operations.

In many scenarios, people are an organization's last line of defense against threats such as malicious code, disgruntled employees, and malicious third parties. It is therefore important to educate workers on what your organization considers appropriate security-conscious behavior, and on the security best practices they need to incorporate in their daily business activities.

Drive Awareness and Train Your Organization

- Use creative ways to help your people understand the threat that social engineering imposes, the skill with which attacks are carried out, their role in protecting the organization, and the advice that will enable them to resist these attacks.

- Provide a regular rhythm of updated information and refresher courses to keep employees aware of the risks involved in relaxing security.

- Keep the message fresh so people don't lose sight of its meaning and importance.

Encourage the Behavior You Want and Enforce Where Necessary

- Many social engineering attacks take advantage of the positive qualities of people and social norms. Find ways to encourage behavior that allows for questioning of why someone needs information or access, such that it becomes socially acceptable to push back or say "No."

- When enforcement is necessary, set policies to require realistic safe behavior. Ensure that users understand why such measures are necessary to protect the organization as well as the consequences of not following the policy.

One Microsoft Way
Redmond, WA 98052-6399
microsoft.com/security