Types of Malware and its Analysis

Samanvay Gupta

ABSTRACT
The paper explores the still-growing threat of website malware, specifically how hackers compromise websites and how users become infected. The consequences of malware attacks—including Google blacklisting—are also explored with an introduction describing the evolution, history & various types of malware. Types of malware described include Virus, Worms, Trojans, Adware, Spyware, Backdoors and Rootkits that can disastrously affect a Microsoft Windows operating system.

Keywords: Evolution of malware, Malware analysis, types of malware analysis, tools

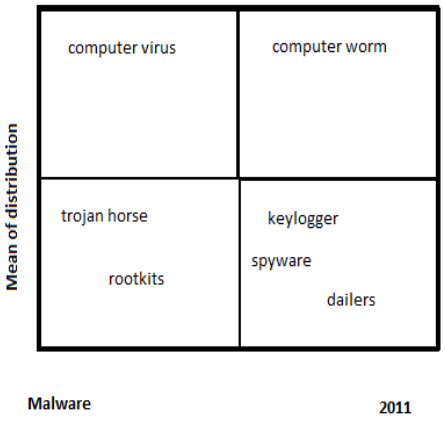
INTRODUCTION
Malware–the increasingly common vehicle by which criminal organizations facilitate online crime–has become an artifact whose use intersects multiple major security threats (e.g., botnets) faced by information security practitioners. Given the financially motivated nature of these threats, methods of recovery now mandate more than just remediation: knowing what occurred after an asset became compromised is as valuable as knowing it was compromised. Concisely, independent of simple detection, there exists a pronounced need to understand the intentions or runtime behavior of modern malware. Recent advances in malware analysis [1, 2, 3,4] show promise in understanding modern malware, but before these and other approaches can be used to determine what a malware instance does or might do, the runtime behavior of that instance and/or an unobstructed view of its code must be obtained. However, malware authors are incentivized to com- plicate attempts at understanding the internal workings of their creations. Therefore, modern malware contain a myriad of anti-debugging, anti-instrumentation, and anti-VM techniques to stymie attempts at runtime observation [5, 6]. Similarly, techniques that use a malware instance's static code model are challenged by runtime-generated code, which often requires execution to discover.

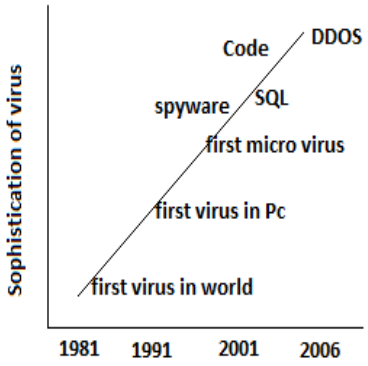
Samanvay Gupta
4th year 1st semester
Computer Science Department Visvesvaraya College of Engineering & Technology
Hyderabad, Andra Pradesh, INDIA
Email: - Samanvaygupta18@gmail.com

In the obfuscation/DE obfuscation game played between attackers and defenders, numerous anti-evasion techniques have been applied in the creation of robust in-guest API call tracers and automated DE obfuscation tools [7, 8, 9, and 10]. More recent frameworks [11, 12, 13] and their discrete components [15, 19] attempt to offer or mimic a level of transparency analogous to that of a non-instrumented OS running on physical hardware. However, given that nearly all of these approaches reside in or emulate part of the guest OS or its underlying hardware, little effort is required by a knowledgeable adversary to detect their existence and evade [14, 15]. It is clearly in the interest of network administrators to detect computers within their networks that are infiltrated by spyware or bots. Such stealthy malware can exhilarate sensitive data to adversaries, or lie in wait for commands from a bot-master to forward spam or launch denial-of-service attacks, for example. Unfortunately it is difficult to detect such malware, since by default it does little to arouse suspicion: e.g., generally its communications neither consume significant bandwidth nor involve a large number of targets. While this changes if the bots are enlisted in aggressive scanning for other vulnerable hosts or in denial-of-service attacks—in which case they can easily be detected using known techniques [16, 17]. It would be better to detect the bots prior to such a disruptive event, in the hopes of averting it. Moreover, such easily detectable behaviors are uncharacteristic of significant classes of malware, notably spyware.

HISTORY OF MALWARE

Viruses and the Rise of the Internet, in 1969, there were four hosts on the Internet. In 2005, that number has exceeded 300 million. It is not surprising that the evolution of computer viruses is directly related to the success and evolution of the Internet, and the comparison between the Internet and a living body that is continuously fighting viral infection and disease is both easy to understand and picture. As the Internet has assumed a life of its own, connecting computers, servers, laptops, and mobile phones around the world into a single, evolving web of interconnectivity, so, too, has malicious code quickly evolved and mutated to become a myriad of increasingly more complex malicious software programs. Simply put, anti-virus is the antidote to this infection. As the Internet has evolved, so has the nature of the threat. Viruses have spawned new forms of malicious life that thrive upon the computational technology of Internet connectivity, data, and voice communications. These new threats can rapidly recreate themselves (worms) to attack their hosts, and then spread rapidly from one host to another. Recently, independent threats have combined in the form of *blended threats* that conspire to identify, disable, or destroy any vulnerable carrier hosts. Brain (1986) was one of the earliest viruses. It infected the boot sector of floppy disks, which were the principal method of transmitting fi les of data from one computer to another. This virus was written in machine code, the basic computing language for personal computers (PCs). Virus propagation was slow and depended upon users physically carrying the infection from one machine to another, and then transmitting the infection via the floppy disk when the PC booted up. These viruses became known as boot sector viruses because the upload executed the virus process. By the early 1990s, well-known viruses like Stoned, Jerusalem, and Cascade began to circulate. The first major mutation of viruses took place in July 1995. This was when the first macro virus was developed. It was notably different from boot sector viruses because it was written in a readable format. The use of such macro programming within common office applications resulted in the Concept virus. Viruses written in readable format, combined with the existence of macro programming manuals and the enhanced capabilities of macro viruses relative to boot sector and contemporary fi le viruses, allowed new macro viruses and variants of existing viruses to be rapidly developed and distributed. Furthermore, with computers now being connected to local area networks (LANs) that were slowly being interconnected to each other, the increased importance and feasibility of fi le sharing provided an efficient distribution mechanism for viruses, which further attracted more writers to this new breed of malicious code. The next major mutation of viruses took place in 1999 when a macro-virus author turned his attention to the use of e-mail as a distribution mechanism. Melissa, the first infamous global virus, was born. After Melissa, viruses were no longer solely reliant on fi le sharing by floppy disk, network shared fi les, or e-mail attachments. Viruses had the capability to propagate through e-mail clients such as Outlook and Outlook Express. As of a result of this and new developments in the capabilities of the Windows® Scripting Host, a devastating virus known as Love Letter was spawned on May 4, 2000. The world has never been the same since. Evolving, mutating, and growing in intelligence and its ability to survive and spread its infection, the virus has jumped from the humble floppy disk to distributing itself quickly around the internal network. The virus is presently capable of spreading seemingly unseen, effortlessly and unstoppably across the global Internet, infecting anything and everything it touches. As antidotes to viruses were developed and immunization programs created and deployed to counteract their effect, some viruses were able to adapt and learn to circumnavigate the efforts made to stop them, and new malicious organisms rapidly came into existence. Today we not only have to cope with viruses, but also with worms, Trojan horses, backdoors, rootkits, HTTP exploits, privilege escalation exploits, and buffer overflow exploits. These new threats identify and prey upon vulnerabilities in applications and software programs to transmit and spread attacks. In 2002, these threats began to combine, and the blended threat was born. By utilizing multiple techniques, blended threats can spread far quicker than conventional threats. And the devastation they can wreak can be far more widespread and destructive





The NSA versus Morris: $100 Million in Damage

The most important security incident of the year was triggered by Robert T. Morris, Jr., a graduate student at Cornell University and son of a National Security Agency researcher. He managed to create a piece of software that would automatically self-replicate on all the systems connected to the government's Arpanet. This was the first time when a computer worm triggered a large-scale security incident 49, and according to the U.S. General Accounting Office, the damage ranged in between $10 million and $100 million, as well as thousands of infected government computers. Although Morris claimed that he had written the virus with no malicious intention in mind (it was allegedly an experiment that got out of control), he was convicted of violating the 1986 Computer Fraud and Abuse Act and was sentenced to three years' probation, as well as 400 hours of community service to go along with the $10,050 fine.

The increased number of computer viruses and worms called for the establishment of a new anti-malware organization, called the Computer Emergency Response Team / Coordination Center (CERT/CE).

In order to fight back the increasingly active malware creators, McAfee released its own antivirus tool. The utility was able to detect and disinfect 44 viruses, an important improvement over IBM's virus-search software that was only able to detect 28.

THE MODERN AGES

2001: the Year of the Worm

The malware development in 2001 was mostly driven by the Internet boom. Worm and virus authors have previously made serious attempts at infecting computer users via the web (such as the **Jer** Internet worm), while others tried to use an Internet connection in order to update their creation and avoid simple string scanners.

2003 - Sobig and the Botnet

Although the **Win32.Sobig** worm had been spotted in isolated locations since January, it did not start causing trouble until August, with the advent of its **Sobig.f** variant. Spreading via e-mail, the **Win32.Sobig** worm s thought to be the first organized attempt to create large-scale Botnets (networks of compromised systems that can be remotely controlled by a bot herder). The main reason for writing **Win32.Sobig** is alleged to be an attempt to create a huge network of zombified computers in order to conduct DDoS attacks on corporate servers.

2004 – Google Draws the Curtains

Malware authors continued to focus mostly on worms during 2004, just as they did in the previous year. The successful attacks carried by **Slammer**, **Win32.Sobig** and **Tantalus** were enough reason to keep improving worms rather than viruses. However, the sharp increase in malware and the utter disaster caused by Slammer called for a solution, and antivirus researchers hurried their technological development. Other major industry

players, such as the popular search engine Google have entered the battle against malware.

2005 – The Sony BMG Scandal

One of the most interesting security threats in 2005 were the so-called worms for instant messenger applications. IM services have become so popular, that almost every PC user around the world enjoyed their services. Although a couple of IM worms have been detected long before 2005, their count significantly increased during the year. The first significant outbreak during 2005 took place in August, when the **Win32.Worm.Zotob.A** worm and some of its variants (**Win32.Worm.Zotob.D**) started infecting US-based computers.

2006 – MacOS X Rides On the Trojan Horse

The New Year was relatively calm, with few major security incidents. The smooth Internet experience users could enjoy was partly due to the fact that Microsoft's Windows XP operating system was safer than its predecessors, but partly due to the fact that the antivirus industry was watching. **JS. Blackworm.A** was the first Internet worm to hit in February 2006. The new piece of malware spreads by e-mail using messages with infected attachments, as well as through unprotected network shares.

2007 – Malware Takes the World by Storm

One of the biggest security threats in 2007 was posed by a new and rapidly-evolving email spamming campaign. The central piece of the new campaign is the **Storm Worm**, a mixed-type piece of malware that combines worm features with backdoor and Trojan capabilities. Initially spotted in the wild on January the 17th 2007, the worm is trying to infect computers, and then to add them to the Storm botnet.

2008 – The Emergence of Rogue Antivirus Software

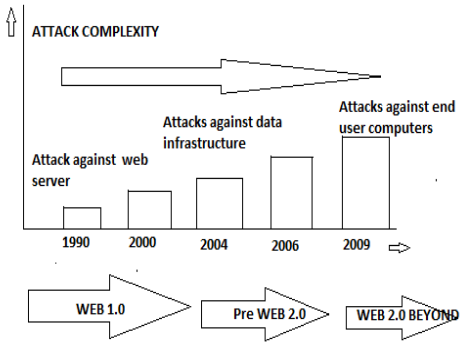
The biggest security threat in 2008 was caused by the discovery in May of the Rustock.C, a backdoor Trojan that allows remote attackers to use the compromised computer as an anonymous proxy server. The new backdoor uses advanced rootkit technologies to conceal its files from both the user and from the operating system itself. This means that malicious users can hijack the system without even the user noticing it.

2009 – The Downadup Invasion

The New Year debuted with the proliferation of the Conficker worm in a logarithmic manner. Although this innovative and highly contagious piece of malware did not inflict any substantial damage to the infected computer, early 2009 estimations confirmed that the total number of compromised machines around the globe during Q1 alone surpassed the population of Belgium or Netherlands. After infection, the worm would attempt to list all the administrative shares on the network and connect to them by performing a dictionary attack. In addition, the worm would also restrict access to a list of antivirus vendor's websites, in order to make disinfection nearly impossible.

2010 – New Security Risks Lurking: Ransom ware and P2P Worms

The experience with the Koobface worm, cyber-criminals brought **Win32.Worm.Prolaco.G,** a new network-aware worm with extremely infectious capabilities. The malware attempts to replicate itself on the local network, and also tries to use a mass-mailer component that harvests e-mail addresses from the local computer and spam its files outside the local network. After successfully compromising the system, the worm would drop a remote access tool that allows an attacker to seize control over the infected machine and dispose of the stored data at will.



TYPES OF MALWARE

Today, PC users and network operators have to fight off and immunize themselves against an ever increasing variety of methods of attack which can infect randomly or target specific networks or

machines in a coordinated attack. **A common name for all software designed to infiltrate or damage a computer system.**

1**. Adware**
Adware is software that enables displaying banner advertisements when the program is running

2. **Backdoor**
A program that installs itself in such a way that the infected computer can be accessed and controlled remotely

3. **Bot / Botnet**
Computers (bots) that are part of a network, which itself is controlled by a robot, is often called a botnet.

4. **Browser hijacker**
This term covers a range of malicious software. The most generally accepted description for browser hijacking software is external code that changes your Internet Explorer settings.

5. **Click jacking**
Click jacking is a malicious technique of tricking Web users into revealing confidential information or taking control of their computer while clicking on seemingly innocuous Web pages.

6. **Dialer**
A dialer is an electronic device that is connected to a telephone line to monitor the dialed numbers and alter them to seamlessly provide services that otherwise requires lengthy access codes to be dialed.

7**. Dropper**
A dropper is a program that has been designed to "install" some sort of malware (virus, backdoor, etc.) to a target system.

8. **Exploit**
The technique used to take advantage of a particular vulnerability.

9. **Grey ware**
Software that may be viewed as useful in some instances, but which also includes component(s) that may be seen as malicious or annoying in other contexts

10. **Hoax**
It alerts about malicious software, which turn out to be false alarms.

11. **Image spam**
Spam using images as all or parts of the text.

12. **Intended**
Due to programming error(s) the malware does not function (as intended).

13**. IP Spoofing**
A technique used to gain unauthorized access to computers, whereby the intruder sends messages to a computer with an IP address indicating that the message is coming from a trusted host.

14. **IRC-bots**
An IRC bot is a set of scripts or an independent program that connects to IRC-Internet Relay Chat as a client, and so appears to other IRC users as another user.

15. **Key logger**
Programs, which log the keys pressed on a keyboard and (usually) send this to a third party.

16. **Money-mulling**
Money mulling is when a criminal persuades an individual to use their bank accounts to launder the proceeds of crime.

17. **Pharming**
Utilization of the DNS to provide wrong data

18. **Phishing**
Attacks which by utilizing software attempts to get your personal information.

19. **Proof of Concept**
A program written to show that a particular technique is possible to use.

20. **Ransom ware**
Encryption of files on a computer, and leaving a message that a certain ransom has to be paid for the decryption key to be disclosed.

21. **Rogue ware**

Rogue ware consists of any kind of fake software solution that attempts to steal money from PC users by luring them into paying to remove nonexistent threats.

22. **Rootkit**
Rootkits are software used to hide files, running processes, Registry entries, or other kinds of data.

23. **Scare ware**
Scare ware comprises several classes of scam software with malicious payloads, or of limited or no benefit, that are sold to consumers via certain unethical marketing practices.

24. **Smishing**
Smishing is derived from the familiar "phishing." The "sm" comes from SMS, the protocol used to transmit text messages via cellular devices

25. **Spam**
Spam is unsolicited information.

26. **Spyware**
Spyware is programs that collect information about a person or an organization without that entity's consent and awareness.

27. **SQL Injection**
SQL Injection or SQL injection is a code injection technique that exploits security vulnerability in some computer software.

28. **Tracking cookie**
The most commonly used track ware is a Tracking Cookie - a small piece of data that identifies a certain user or a certain computer, with the help of a web browser configured to store cookies.

29. **Trojan**
A program that seems to be genuine and even useful, and thereby tricks the users to install/use it

30. **Virus**
A virus is designed to copy itself and propagate from one computer file to another, usually by attaching itself to program files.

31. **Vishing**
Phishing where Voice over IP (VoIP) is used as the communication channel

32. **Vulnerability**
A flaw in a computer program, which may allow someone to perform action(s) not intended by the author of the program

33. **Worm**
A worm will infect other computers, but do not propagate by infecting other files.

34. **DNS Changer Malware**
DNS (Domain Name System) is an Internet service that converts user-friendly domain names into the numerical Internet protocol (IP) addresses that computers use to talk to each other. When you enter a domain name such as www.fbi.gov, in your web browser address bar, your computer contacts DNS servers to determine the IP address for the website. Your computer then uses this IP address to locate and connect to the website. DNS servers are operated by your Internet service provider (ISP) and are included in your computer's network configuration. DNS and DNS Servers are a critical component of your computer's operating environment—without them, you would not be able to access websites, send e-mail, or use any other Internet services. Criminals have learned that if they can control a user's DNS servers, they can control what sites the user connects to on the Internet. By controlling DNS, a criminal can get an unsuspecting user to connect to a fraudulent website or to interfere with that user's online web browsing. One way criminals do this is by infecting computers with a class of malicious software (malware) called DNS Changer. In this scenario, the criminal uses the malware to change the user's DNS server settings to replace the ISP's good DNS servers with bad DNS servers operated by the criminal. A bad DNS server operated by a criminal is referred to as a rogue DNS server. The FBI has uncovered a network of rogue DNS servers and has taken steps to disable it. The FBI is also undertaking an effort to identify and notify victims who have been impacted by the DNS Changer malware. One consequence of disabling the rogue DNS network is that victims who rely on the rogue DNS network for DNS service could lose access to DNS services. To address this, the FBI has worked with private sector technical experts to develop a plan for a private-sector, non-government entity to

operate and maintain clean DNS servers for the infected victims. The FBI has also provided information to ISPs that can be used to redirect their users from the rogue DNS servers to the ISPs' own legitimate servers. The FBI will support the operation of the clean DNS servers for four months, allowing time for users, businesses, and other entities to identify and fix infected computers. At no time will the FBI have access to any data concerning the Internet activity of the victims. It is quite possible that computers infected with this malware may also be infected with other malware. The establishment of these clean DNS servers does not guarantee that the computers are safe from other malware. The main intent is to ensure users do not lose DNS services.

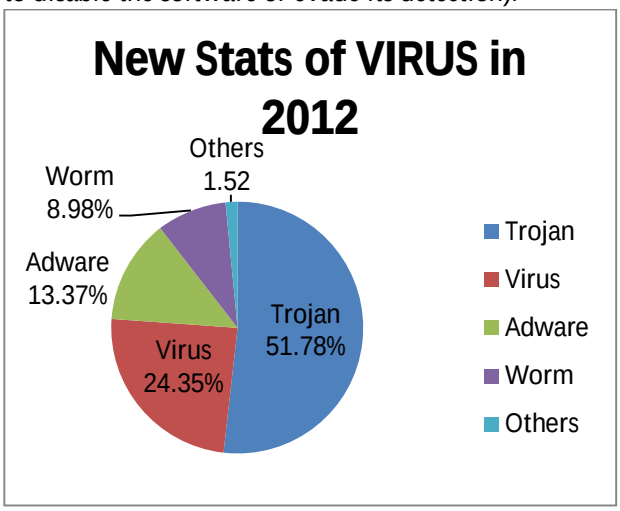
35. Vienna: Actively Fighting Malware Threats

The **Vienna.636.A** virus marked another important milestone in the malware industry. Its appearance in the wild and its highly infectious potential managed to raise users' awareness towards the increasing security threats. Although the originator is still unknown, it is for sure that Franz Swoboda was the first person aware of the **Vienna.636.A** virus. The global IT community was up in arms in order to identify its creator, and according to those days' reports, Swoboda had received a copy of the virus from Ralf Burger. However, given the fact that Burger's allegations would incriminate Swoboda as the author, the later claimed the contrary, and blamed it on Swoboda.

Vulnerabilities commonly exploited by Malware

Based on an analysis of malware-related vulnerabilities in the National Vulnerability Database, [18] the following types of vulnerabilities are typically exploited by malware to disseminate, propagate, and install themselves. Most worms exploit vulnerabilities in the victim computer's software or network to affect their own propagation. When a software vendor issues a patch, or a "researcher" announces vulnerability, the worm author can use the information in the announcement to understand and craft a worm to exploit the vulnerability.

1. Buffer overflows (the real vulnerability is a design flaw, *i.e.,* the lack or failure of input validation to prevent the submission of overlong data strings);
2. Weak access control (due to poorly designed or configured access controls);
3. Poor or incorrect handling of malformed data (due to lack or failure of input validation to filter out malformed data);
4. Decoding errors (*e.g.,* browser or Web server Uniform Resource Locator [URL] decoding errors);
5. Sabotaged configurations (*e.g.,* through tampering with the configuration script)
6. Vulnerabilities in anti-virus software (exploited to disable the software or evade its detection).



HOW WEBSITES AND THEIR USERS GET INFECTED

It is alarming just how easily it can infect websites and their users. While many different attack methods exist, injection and cross-site scripting are the most popular. With these types of strikes, users can become infected with malware just by visiting a site. Often called "drive-by downloads," these attacks do not require the user to actively download an infected file. The malware will download itself to users' computers without their knowledge.

Top Drive-By Downloads

No.	Threat	Description
1.	Trojan.Clicker.CM	Displays pop-up ads that lure users to click; when clicked, the pop-ups lead to sites that contain malicious adware.
2.	Trojan.Wimad.Gen.1	Poses as a common Windows Media audio file; if run, this threat allows attackers to load malicious software onto a user's computer.
3.	Trojan.AutorunINF.Gen	Malware that autoruns and executes the Conficker virus that has the potential to turn computers into hosts in a botnet, and lock users out of accounts, among many other symptoms.
4.	Trojan.Downloader.JLPK	A malware that decrypts functions and downloads more malware files.
5.	Trojan.Exploit.SSX	Usually appears on sites through SQL injection attacks that insert an invisible IFrame into clean code; can steal user information.
6.	Trojan.Downloader.Js.Agent.F	A JavaScript file which inserts a links to malicious JavaScript and IFrames into clean code; can steal user information.
7.	Trojan.Exploit.ANPI	A Visual Basic script that exploits a vulnerability in Internet Explorer to download, save, and execute infected files; can steal user information.
8.	Trojan.IFrame.GA	A JavaScript file which gets injected into compromised websites and sends browsers to a collection of exploits such as Trojan.Exploit.ANPI; can steal user information.
9.	Trojan.Downloader.JS.Psyme.SR	Uses scripts to download other malware onto the user's computer by the names GameeeEeee.pif and Gameeeeeee.vbs; can steal user information.
10.	Trojan.Downloader.WMA.Wimad.S	A disguised application which is commonly in a media file extension; once run, it prompts the user to download a file named, "PLAY_MP3.exe" which can steal information.

Typically, these stealth attacks take advantage of compromised web servers and website developer desktops that are not secure, affecting web server PHP, HTML, and JavaScript files. Malware commonly targets unpatched browsers, vulnerable operating systems, and popular applications such as ActiveX, Microsoft Office, and RealPlayer.
Online content is dynamic. Websites are updated constantly. And with each update, malware can find a new opportunity for infiltration. The average online shopper may not have the latest security patches installed, or may be using outdated browsers and plug-ins that may not be completely secure. Malware evolves just as rapidly as the rest of the Internet, so even up-to-date systems with the latest patches may still be vulnerable to attack. With no guarantee that a site has been recently scanned for malware, even the most tech-savvy online shopper may end up infected with malware.
To compound the problem, it is now easier for hackers to attack sites with malware than ever before. With the proliferation of "packaged" attack software—also called exploit or command-and-control (C&C) toolkits—hackers can develop malware much faster. For example, the Zeus toolkit has accounted for more than 90,000 unique malicious code variants alone.8 Toolkits with C&C servers create botnets, or a collection of infected computers. When malware is installed on a new computer, the malicious code reports back to its C&C server, adding the latest compromised computer to its botnet.
Website malware spans the gamut from keystroke loggers to password harvesters to screen scrapers, along with other tools designed to infect a website visitor's computer. Once compromised, the attacker has a backdoor to that computer to transfer stolen data or perhaps send thousands of spam messages.

WHAT MAKES A WEBSITE VULNERABLE TO MALWARE?

Website owners continue to look for ways to improve the customer experience and to increase their website's popularity, not to mention their profits. Supporting the latest mobile devices, social networking, location-awareness, user customization, and user interactivity are key enhancements. Unfortunately, advances in website capabilities greatly increase a website's risk of inadvertently hosting malware. By August 2009, Google had indexed over 350,000 malware-hosting websites and distribution of malware via websites almost doubled in 2010, with over 286 million unique variants of malware identified in 2011 alone.

Malware Analysis

Most of the pieces of malware that are currently in the wild are designed in such a manner that they won't reveal their presence in order to keep generating profit or to cause damage for as long as possible. However, while some pieces of malware do not reveal any visible symptoms, you can still find out if and when you got infected.
Computer malware usually tampers with users' data in such a way that there is always a side effect. For instance, no matter how well concealed a piece of malware is, it will still affect your computer's performance or delete programs and system files. This could instantly render some of your programs useless, as they won't be able to find one or more critical files (usually DLLs). However, programs or the operating system itself would sometimes crash because of some critical files being accidentally deleted or even because of wrong settings applied by mistake, so not any crash should be regarded as a viral infection. Still,

if you haven't done anything wrong and your system starts behaving abnormally, chances are that you have been infected by a computer virus or a malicious browser add-on. Computer viruses usually infect multiple files on a system, in order to prevent the user from deleting the "suspicious" file. If a file is deleted, either by accident, or voluntarily (as the user detects that something is wrong with the file), the other infected files could carry on with destroying data100.
Unlike computer viruses, Trojan Horses are more difficult to detect and eliminate. This is because Trojan Horses do not leave too much evidence about its presence on the host computer. On the contrary, it tries to cause as little damage to the computer as possible, in order not to draw the computer user's attention. This way, it delays the mean time to detection and elimination, which means that it would be able to parasite the system for longer periods of time. However, there are some signs that could "tip" you of the presence of an unwanted "guest" on your computer. For instance, if you notice that your files appear to be moving from a location to another or change their file size (harder to detect, yet not impossible), then you might be infected with a Trojan or you might be the victim of remote-control software. System instability could also trigger the first suspicions that you have been infected. Apart from miscellaneous hardware flaws or driver conflicts, computer viruses are the most common sources of performance loss. Messages popping up upon the OS boot or after it has been completely loaded are also a sign that something is not all right with the computer. Lock-ups, freezes and computer restarts out of the blue or visible decreases in performance should also alert you about the presence of an intruder.

TYPES OF MALWARE ANALYSIS

There are two types of malware analysis performed by the security experts: Code (static) Analysis and Behavioral (dynamic) Analysis. Although both the above analysis will give you a very clear picture about the working of the malware, but tools, time and skills required to perform these are very different.

Behavioral Analysis

Behavioral Analysis is a method where a behavior of malware is monitored upon its execution in a sandbox environment. The behavior is monitored, such as, creation or deletion of a process, adding or deleting entries in the register, whether malware is connecting to a remote server, added itself in auto-run, monitoring network traffic, etc. This technique is easier compared to Code Analysis, where the source code of malware is obtained or analyzed using a technique called reverse engineering.

Code Analysis

Code Analysis is a method in which the actual code of the malware is examined by reverse engineering the malicious executable. The approach gives us a better understanding of the malware functions.

TRENDS IN MALWARE INCIDENTS

From 2005 to 2006 the total number of new malicious programs increased 41 percent, according to Kaspersky Lab's "Security Bulletin 2006: Malware Evolution," and a staggering 172 percent, according to L. Corrons in "Panda Labs' Annual Report 2007." Corrons also predicted a 60 percent increase in unique novel malware starting in 2007.
Malware appears in any given environment in which the following criteria are satisfied
1. The targeted platform runs an OS that is widely used.
2. Reasonably high-quality documentation is available to the malware writer.
3. The targeted system is not securely configured, or has a number of documented vulnerabilities.
Potentially vulnerable OSs and applications include
1. All popular desktop OSs (*e.g.,* Windows, Macintosh OS X, Linux)
2. Most general purpose Web, office, graphics, and project management applications;
3. Most graphical editors;
4. Applications with built-in scripting languages.

It is common for hundreds or even thousands of types of malware to exploit the same handful of vulnerabilities. This happens because the vulnerabilities are not addressed by virus definitions produced by anti-virus software

vendors, and patches are not always issued in timely manner (if at all) by the supplier of the target machine's OS or application if they are issued, they may not be installed in a timely manner, if at all, by the target machine's administrator or operator. The following are the most prevalent vectors for malware propagation—

1. E–mail (includes spam and other phishing emails),
2. Web sites,
3. Instant messages,
4. Removable media (*e.g.,* "thumb" drives, compact discs [CD]),
5. IRC,
6. Bluetooth (emerging),
7. Wireless local area network (theoretical).

Virus-writers are using increasingly complex techniques to prevent their virus code from being detected by signature-matching anti-virus scanners.
The techniques they use include

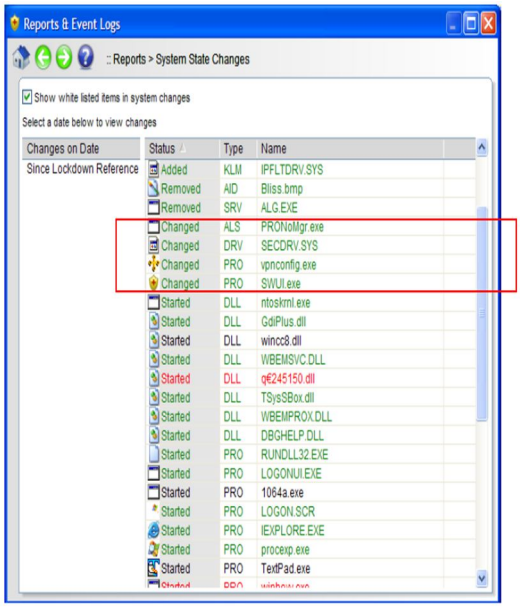
1. Polymorphic encryption—the virus is encrypted to escape detection.

2. **Metamorphic obfuscation**—the virus code is "morphed" by adding non-virus-related logic to obscure the presence of virus logic. As the code changes, so does the virus signature generated from that code, thereby rendering the virus undetectable by matching the signature of the pre-morphed virus.

3. **Code integration**—Virus code is mixed into valid program code using a tool such as the Mist fall Virus Engine.

DANGEROUS MALWARE DEFEATS ANTI-VIRUS

New malware/viruses can make computer impossible to clean using anti-virus! Anti-virus products use signature and heuristics to find malware and remove it. That approach is being undermined by new malware, and Cut wail is one such malware. Cut wail malware uses rootkit technology to make it very difficult to detect and remove. Once it infects the computer, it starts sending out large amount of SPAM from that computer. The most recent variant of the Cut wail family of rootkits goes a step further and makes any computer impossible to clean by infecting large number of installed programs with polymorphic code. Once a large number of files are infected, starting any of the infected files restores the rootkit on the computer. Because the infected files are impossible to clean, only option is to reinstall operating system, applications, and drivers.



Rootkit malware modifies files to become regenerative and cannot be removed by anti-virus software.

ANTI-MALWARE COUNTERMEASURES

Technology-Based Countermeasures
Countermeasures to malware fall into three general categories—

1. Detection—the ability to recognize and locate malware on a system, in a file on that system, and/or in software, hardware, or media not yet installed on the system;

2. **Prevention**—keeping malware from entering, installing, and/or executing on a system. Also, keeping malware from propagating itself to other areas of a system or to other systems. Also, deterring malicious actors from embedding or implanting malware in software before it is installed on the system.

3. **Eradication**—Removing malware and all of its associated traces (files, processes, system changes), and restoring the system to its pre-infected state. Each of these categories of countermeasures is discussed below

Detection
The ability to detect the presence of malware is the first step toward its isolation and eradication. Traditionally, virus detection has been performed by matching "signatures" generated from virus code captured by researchers in a laboratory environment (*e.g.,* an anti-virus tool vendor's lab) against virus code captured in the wild. However, signature-matching has inherent inaccuracies, and is not effective for detecting more sophisticated, complex malware such as rootkits and logic bombs. More advanced behavior-based detection techniques are emerging to address the need to find such malware in both systems under development and systems in operation.
1. Signature-Matching
2. Behavior-Based Detection
3. Anomaly-Based Detection
4. Detection of Indirect Malware Indicators

Prevention
Approaches throughout the system life cycle are needed to prevent malware from being embedded or implanted in systems under development, being delivered to and installed on an operational systems, and being executed on operational systems and then propagating to other systems on the same network. These approaches include—
1. Quarantine
2. Constrained Execution Environments

Eradication
Removal of malware and recovery from its effects on operational systems/environments focuses on sanitizing all systems and devices suspected of harboring the malicious code to eliminate all traces of that code, and to restoring the affected systems to their pre-malware state. The technologies for accomplishing eradication are often built into the same tools that are used to detect malware. However, in the majority of cases, at least some operational restoration measures will also be needed, such as restoration of the system from backup, reinstallation of clean versions of affected software, and also strengthening of detection and prevention countermeasures to reduce the likelihood of future malware infections.

ANTI-MALWARE TOOLS

Classification of Tools
The tools described in the IATAC IA Tools Database can be classified using two different classification methodologies. The first set of classification determines how the tool is used within an organization—

1. Host- or endpoint-based—used to protect individual computer systems.

2. **Network-based**—Monitor an organization's networks for signs of malicious code activity, by actively recording network traffic, analyzing firewall, router and application logs, or performing scans of systems over the network. They may also operate at the network boundary to detect and block malware from entering the network.

3. **Tools-as-a-service**—Access to tools in the form of a malicious code detection service accessible over the Internet. Such services are most useful when they augment the use of host/endpoint-based and network-based tools, in order to gain better coverage. Often tools-as-a-service are provided for detection only, with the user required to purchase a license or service in order to eradicate threats found by those tools The second set of classification is based on the mechanisms the tools use to perform their anti-malware detection and response activities—

4. **Malware detection and removal**—Tools that primarily perform detection and removal of malware; subcategories include: virus detection and removal, Trojan detection and removal, spyware detection and removal, and rootkit detection and removal.

5. **Detection of malware indicators**—Tools that rely primarily on behavioral and heuristic anomaly detection and analysis to identify system or program behaviors that are indicative of infection by malicious code.

6. **Trace detection**—Tools that scan systems for API hooks and other common traces of the presence of malicious code on the system.

7. **Malicious code analysis**—Malware research tools that focus on analyzing malicious code to determine how it is structured and how it operates, usually in support of generating new malware signatures or removal techniques.

8. **Malware honeypot**—Tool that captures code originating from a suspicious source or code of an unknown type, and monitors the behavior of that code to determine whether it is, in fact, malicious.

9. **Hidden process detection**—Tools that detect hidden processes running in the OS or kernel, as such processes often indicate the presence of malicious code.

Tool Selection Criteria

The tools selected for inclusion in this Report satisfy the following three criteria—

1. Definition—these tools satisfy the objective, approach, and methodology of an anti-malware tool based on the definition of malware.

2. **Specificity to malware**—The primary and explicit function of these tools is to reduce the risks and adverse impacts associated with malware, either operationally, by detecting, blocking, isolating and constraining, or removing and recovering from malware attacks, or by enabling analysis and better understanding of malware structure and behavior.

3. **Current availability**—the tools that are included in this report are currently available from the Government, academia, or commercial sources, or as freeware on the Internet.

CONCLUSION

Effective malware defense is a difficult and increasingly important issue for computer networks; however, current defenses are often unable to manage these threats. Current solutions rely on malware fingerprints (signature) to be known a priori, which is not always possible. Computing has become part of the fabric of our everyday lives, and the foundations of modern society are becoming more digital every day. Information and communications technology (ICT) has transformed for the better how we live, but society still confronts some long-standing and evolving challenges. As the number of people, computers, and devices that connect to the Internet continues to increase, cyber threats are becoming more sophisticated in their ability to gather sensitive data, disrupt critical operations, and conduct fraud. Cyber threats today are often characterized as technically advanced, persistent, well-funded, and motivated by profit or strategic advantage. Security intelligence is a valuable asset to all Internet users, organizations, governments, and consumers alike, who face a myriad of threats that are anything but static. Because we live in a world that is so dependent on IT, dedication to security, privacy, and reliability might be more important today than it was than when Trustworthy Computing was established in 2002. Computing continues to contribute to the computing ecosystem as we face a new world of devices, services, and communications technologies that continue to evolve.

REFERENCES

[1]. M. Bailey, J. Oberheide, J. Andersen, Z. M. Mao, F. Jahanian, and J. Nazario. Automated Classification and Analysis of Internet Malware. In RAID, 2007.
[2]. M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant. Semantics-Aware Malware Detection. In S&P (Oakland), pages 32–46, 2005.

[3]. M. Christodorescu, C. Kruegel, and S. Jha. Mining Specifications of Malicious Behavior. In ESEC/FSE, pages 5–14, 2007.

[4]. C. Kruegel, W. Robertson, and G. Vigna. Detecting Kernel-Level Rootkits through Binary Analysis. In ACSAC, pages 91–100, 2004.

[5]. P. Bacher, T. Holz, M. Kotter, and G. Wicherski. Know your enemy: Tracking botnets. http://www.honeynet.org/papers/bots, 2005.

[6]. P. Szor. The Art of Computer Virus Research and Defense. Addison-Wesley Professional, 2005.

[7]. L. Martignoni, M. Christodorescu, and S. Jha. OmniUnpack: Fast, Generic, and Safe Unpacking of Malware. In ACSAC, pages 431–441, 2007.

[8]. D. Quist and Valsmith. Covert Debugging: Circumventing Software Armoring. In Black Hat USA, 2007.

[9]. A. Vasudevan and R. Yerraballi. Stealth Breakpoints. In ACSAC, pages 381–392, 2005.

[10]. C. Willems, T. Holz, and F. Freiling. Toward Automated Dynamic Malware Analysis Using CWSandbox. IEEE Security and Privacy, 5(2), 2007.

[11]. Anubis: Analyzing Unknown Binaries. http://anubis.seclab.tuwien.ac.at.

[12]. BitBlaze Binary Analysis Platform. http://bitblaze.cs.berkeley.edu.

[13]. Norman Sandbox Whitepaper. http://www.norman .com/documents/wp sandbox.pdf, 2003.

[14]. P. Ferrie. Attacks on Virtual Machine Emulators. Symantec Advanced Threat Research, 2006.

[15]. TEMU: The BitBlaze Dynamic Analysis Component. http://bitblaze.cs.berkeley.edu/temu.html, 2007.

[16]. Singh, S., Estan, C., Varghese, G., Savage, S.: Automated worm fingerprinting. In: Proceedings of the Symposium on Operating Systems Design and Implementation (2004)

[17]. Moore, D., Voelker, G.M., Savage, S.: Inferring internet denial-of-service activity. In: Proceedings of the USENIX Security Symposium (2001)

[18]. "Net threats: Why going online remains risky," in *Consumer Reports,* September 2007.