From Fun to Profit

The Evolution of Malware

Andrew Lee

Pierre-Marc Bureau

Table of Contents

Introduction	2
The Challenge of Artificial Life	2
Threat Consolidation	3
Profits	4
Advanced Threats	4
Click Fraud	5
Information Theft	5
Access Rental	5
Extortion	5
Case Study: Storm Worm	6
Is the Worst Yet To Come?	9
There is Hope	10
Conclusions	11
References	12
Further Reading	12

Introduction

Malware, a word created by the contraction of malicious and software, describes a category of software created for malicious purposes. Examples of malware have been present in the digital world for at least twenty five years.[1] As with any other species, malware had to go through an evolutionary process to survive the changes that have occurred in its lifetime: this evolutionary shift has been all the more dramatic for being compressed into the last quarter of a century. The first "families" of malware were created by computer hobbyists to show off their own programming skills. The virus-writing scene has changed greatly, though, and profit is now the primary driving force behind nearly all malware creation. The huge sums of money that can be made by creating and exploiting malware are a great incentive to malware authors to improve their techniques. From our analysis of past and present trends, we predict that malware programming techniques will continue to increase in sophistication: and that we will have to deal with more malware that exploits software security flaws and uses "defensive programming" aimed at reducing its own susceptibility to detection and removal by security software.

In this article, we will present an overview of the evolution of malicious software*. We will focus on the objectives of this type of program to provide evidence for our predictions as to how they will evolve in the years to come.

The Challenge of Artificial Life

The first computer virus-like programs appeared some twenty-five or more years ago, though the term "virus" only really passed into regular usage with the publication of Dr. Fred Cohen's first papers on the topic.[2] These viruses were generally Proof of Concept (PoC) programs, demonstrating that computer programs could replicate and propagate between files and even between computers, and were often created by computer enthusiasts for the sole purpose of exploring the possibilities of artificial life.[3]

The only criterion determining the "quality" of many of the first viruses when they left their creator's laboratory, was the technical skill displayed by their programmer. At that time, antivirus researchers would often receive a virus sample directly from its creator. The malware authors were passionate about their creations and wanted to share them, and to tell the world about their skills. Much of this malware was never released "into the wild", meaning that they were never seen to infect real-world system outside virus or antivirus laboratories. Malicious code that is never seen in the real world is called "zoo" malware, mainly to distinguish it from "In the Wild" (ItW) malware.[4]

* This paper is a translation from a white paper in French, available at http://www.eset-nod32.fr
The content of this paper was also presented at InfoSec Paris in 2007

In reaction to the malware authors' thirst for notoriety, and with clear knowledge of the potential for misuse and harm that viruses could bring, antivirus researchers resolved never to glorify the creation of malware. They established a convention by which malware is never labeled with the name of its author, and self-glorifying, self-serving information found inside the body of malicious software is not generally publicized. This convention is intended to reduce the attention paid to malware authors as much as possible, and to give them less incentive to write any more nasty programs. A recent example of this scenario was seen in March 2007, when a new Trojan appeared. The body of the Trojan included the text string: "Hello, my name is IrnBot". Antivirus researchers therefore decided to name this malware "RinBot" instead of "IrnBot" [5]

Threat Consolidation

Computer enthusiasts have been meeting and holding discussions on Internet Relay Chat (IRC) servers for a long time. It came as no surprise that this is where some of the first networks of infected computers appeared. The first networks of robots (botnets) were used to gain control of chat rooms and, sometimes, to launch Denial of Service (DoS) attacks on other users when an argument broke out. (The term Denial of Service is usually applied where an attacker tries to disrupt or deny the use of a network connection, rendering it unusable by its victim for a period of time.) The first malicious botnets were simply toys in the hands of malware authors.

Within a couple of months, people that controlled malicious botnets (also called botmasters or botherders) began to understand that controlling a world-wide network of zombies (computers compromised by a malicious bot) had huge potential for generating income. Botmasters soon started using their botnets to perform click fraud. Click fraud exploits the advertisement model where advertisers pay a couple of cents to a webmaster each time a visitor clicks on an ad. Botmasters started using infected hosts to click on ads placed on websites owned by themselves or their customers so as to make money from the botnets they controlled.[6]

Profits

Having realized the potential for making illicit profits, botmasters started researching ways to infect large numbers of systems, and looked at methods of increasing the persistence of their software on infected systems – that is, to maximize the time and amount of use they could get out of a compromised system. To increase their profit, they made alliances with spammers, the distributors of unsolicited e-mails. Spammers were ready either to pay botmasters for every spam sent on their behalf, or, literally, to rent direct use of their botnets for spam dissemination.[7] To maximize persistence, and reduce detection rates, they began to experiment with rapidly altering and updating the base code of their malware.

Another technique developed by botmasters to increase their profit is to steal personal information from compromised hosts and to sell it on the black market. Botmasters updated their malware to include spyware functionality that would let them monitor keyboard activity and the victim's web browsing activities. For example, a valid credit card number from a well known credit card provider can be sold on the black market for a couple of dollars.[8]

Since the beginning of the twenty-first century an underground economy has developed and matured. We are now facing organized and well trained programmers who create advanced malware, either to use directly for profit or to resell them to other gangs.

Advanced Threats

Modern malware rarely has anything to do with fame or glory. Malicious programs are developed to offer a wide range of functionalities to the bot controller, while remaining hidden from detection by security software, as well as from system users.

Malware has changed and diversified greatly, and can be used to attain many different objectives. For this reason, we can state that malicious programs are purely and simply independent intrusion agents.[9] Functionality implemented in malware can be classified according to their objectives. We can classify some of these objectives and activities according to the following categories.

Click Fraud

This type of activity has been known for years but botmasters still find it profitable to use a large network of compromised computers to click on web advertisements that will bring profit to their owner.[10]

Information Theft

There are numerous types of information worth stealing that are typed into a computer. Among these, we can include usernames and passwords for bank accounts, online gaming accounts, and even credentials for social networking sites like MySpace, FaceBook and LinkedIn. Lists of visited websites are also worth money to advertising and marketing companies. Finally, e-mail addresses stored in address books or any other file type such as Microsoft Excel spreadsheets or Word documents are valuable goods that find a ready underground market.

Access Rental

Once they have control of thousands of infected computers, botmasters can simply rent access to the network to other actors in the world of cyber criminality. Spammers have frequently been seen renting botnets for use in sending out unsolicited e-mails. Rental techniques can vary: in some cases; a botmaster will charge his "sponsor" a certain amount of money for each of their programs that he installs on his botnet. On the other hand, he can simply ask for a sum of money to send a previously agreed amount of spam messages.[6]

Extortion

In the past years, we have witnessed a number of extortion cases where botnets have been used. In a typical case,[10] network administrators return to work after the weekend to discover that a majority of their servers are not reachable from the Internet. After a short investigation, they discover that their Internet link is saturated with packets. After a couple of hours and many calls of complaint, the network administrators receive an e-mail or phone call. This is a request for a "ransom" that should be delivered in an anonymous account within the hour, or else the attacks will continue. For many enterprises, installing and maintaining

Denial-of-Service mitigation infrastructure is too expensive.[11] Furthermore, it is often difficult for them to make contact and work with law-enforcement agencies. For these reasons, they often comply with the ransom request and thus increase the profits made by the botmasters.

The objectives and motivations behind malware creation are not the only elements that have changed in the threat landscape. Development techniques have also evolved. Malware creators have teamed up to use established business models and are now using state-of-the-art group development tools. They have also started to use modular architecture for their programs to reduce development time and facilitate the implementation of new components and upgrades to their malware. A good example of this evolution in programming habits is represented by the big bot families like SDBot and PhatBot.[12] Multiple variants of these bots appear every week and they are quickly modified to include new exploitation vectors when a security flaw is discovered and publicized.[13]

Case Study: Storm Worm

The Storm Worm, also known as Nuwar, Zhelatin, and Peacomm, is a good example of the modern threats that we have to face.[14] Estimates on the number of systems that have been infected by Nuwar vary greatly. However, it is safe to assume that the Nuwar botnet includes at least a couple of hundred thousand infected nodes. Nuwar is not in itself a revolutionary development in the field of malware creation. On the other hand, it is a perfect illustration of what a highly motivated group of skilled programmers can come up with. This malware uses advanced techniques to evade detection from antivirus products, to infect new computers, and to build a reliable network for command and control.

The first interesting characteristic of Nuwar is its command and control (C&C) mechanism. Instead of using the IRC protocol like many other malware families, Nuwar uses Overnet, a peer-to-peer network protocol used in applications like eDonkey.[15] The use of a decentralized network protocol makes tracing the botmaster almost impossible. Furthermore, a compromised system is only in contact with a limited number of other infected hosts. This network segmentation makes it impractical to identify all the infected hosts. Nuwar uses its peer-to-peer network to distribute software updates. Infected hosts receive new and updated components numerous times per week, either to add new functionality or to hinder detection by antivirus products.

Another technique used by Nuwar to evade antivirus detection is to keep changing the binary files that are served by malicious web pages. The binary files are changed every

thirty minutes to make sure that the file downloaded by a victim has not been analyzed previously by security vendors and is therefore less likely to be detected as malicious. These modifications are made by automatically changing certain key values in the packer applied to this malware.

A runtime packer is a software "envelope" used by malware authors to hide the functionality of an executable file. A number of packers are available publicly[16] and are used by legitimate software to reduce the size of their executable files and to protect the intellectual property that is distributed with their code. Packers encode programs to which they are applied.[17] Upon execution, the packer will decode and decompress the original program in memory and execute it. If a key value is modified in the encoding routine, the binary file that is produced looks completely different, especially to security software that relies on detection by known signature. Nuwar authors have automated the process of modifying their packer so to generate thousands of new variants of their program as required.

Once it gets onto a system, a Nuwar variant will try to hide its presence with rootkit-like techniques. Nuwar's rootkit injects code into the operating system (Microsoft Windows) to try to ensure that an application (such as an antivirus scanner) trying to view critical files will only see what the programmer wants it to see. For example, if an application tries to access the list of files present in the Windows folder, the rootkit will intercept this call and hide the entry "peers.ini". This is the list of peers that are contacted on its peer-to-peer network.

In the context of malware, we define persistence as the ability to remain installed and functional on a system over time. To ensure its persistence on a system, Nuwar uses one of the oldest viral techniques: file infection. This threat infects the network driver 'tcpip.sys' and adds a small routine that will start Nuwar's program every time the operating system loads. It is worth mentioning that this small routine is also obfuscated with a packer to slow down analysis of this infection.

Finally, the authors of the Storm Worm have chosen one of the oldest but most effective techniques to get their program loaded onto a victim computer: social engineering. Nuwar's only propagation vector is to include links to malicious web sites in infective e-mails. In one of its message variants, the e-mail invites the recipient to visit a web page where electronic games can be downloaded. Of course, all links on this web page point to a file called "ArcadeWorld.exe" which is the latest variant of Nuwar.

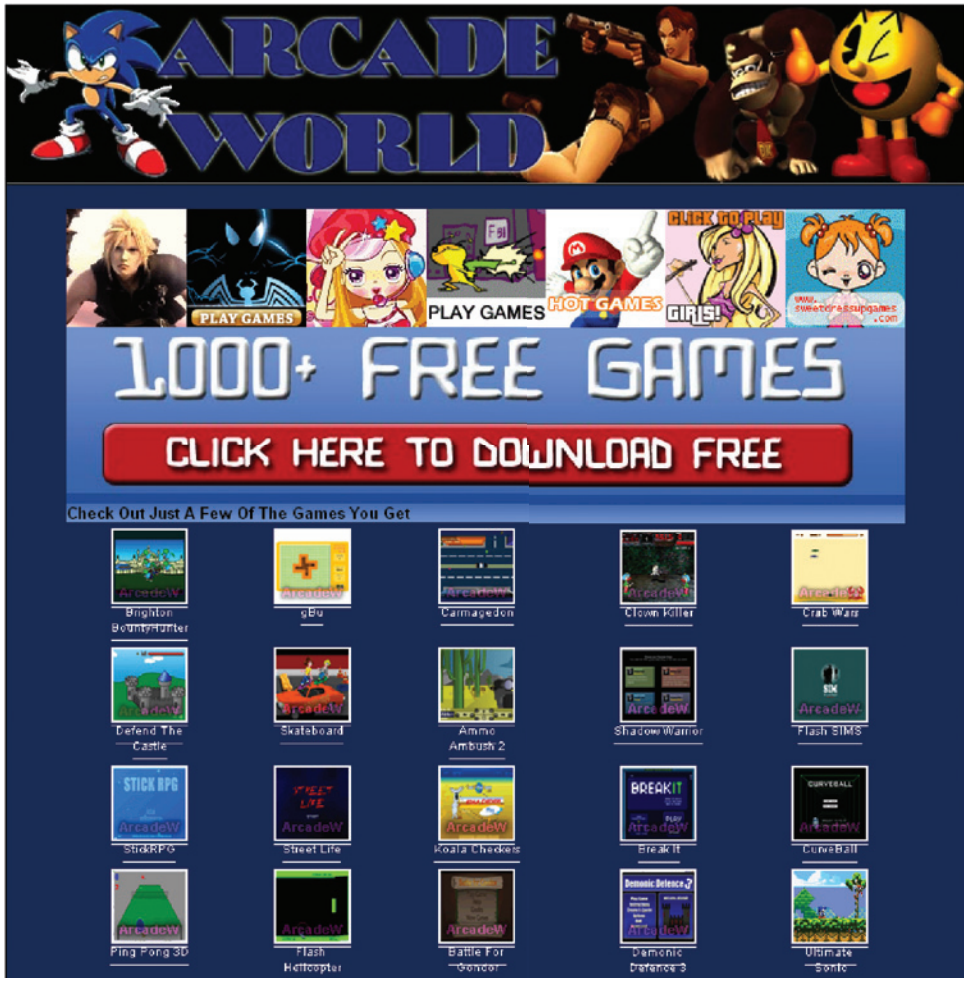

Figure 1: Deceptive Web Page Dispensing Nuwar, Disguised as a Computer Game

Profit is the driver for the development and support of Nuwar. This botnet has been used numerous times to conduct pump and dump and other stock fraud operations. In a pump and dump scam, the infected computers are used to send hundreds of thousands of e-mails intended to trick users into buying cheap stock on the stock market. The price of the stock is artificially inflated and when it reaches a threshold, the Nuwar controllers (or their customers) sell all their shares and make enormous profits.[18]

Nuwar is a good example of the type of threats we will have to face on a daily basis in the near future. Its code is well developed and optimized. It uses external libraries to compress its data and to establish network connections. Furthermore, it is actively maintained and upgraded daily by its authors. This threat does not use a lot of system resources, and tries to remain installed on a system for as long as possible to maximize the potential profit for its controller.

Is the Worst Yet To Come?

In general, we need to be prepared for the increased professionalization of malware authors. These programmers are strongly motivated not to repeat the errors they have made in the past, not least by their aspiration to large financial profits. They will take their time to test their malware, in laboratories they now have no difficulties in resourcing, before releasing it on the Internet.

Malware creation is becoming more political, which is why we predict that malware authors will invest more money and time into research and development. They will invest significant resources into finding new security flaws in operating systems and popular client software, so as to find new infection vectors by which they can install their malware on more computers. The exploitation of numerous security flaws in the Microsoft Office Suite to install malware is one sign of this trend.[6]

The use of packers to hide malware is a trend that is here to stay. We expect malware creators to continue using this type of obfuscating envelope in an attempt to evade antivirus detection. They will continue to enhance their packers by adding new tricks in an attempt to fool reverse engineering and analysis, and by incorporating checks for when the malware is being executed under emulation or in a sandbox. (If the malware "realizes" that it's running in an isolated environment, it will behave differently to the way in which it behaves on an infected system.) We predict that serious malware authors will continue to turn away from publicly available packers, in order to develop and maintain their own private packers, designed to be harder to detect and circumvent.

Some antivirus systems now use emulation as a major component of their detection process. A file is launched inside the emulator and analyzed in an isolated environment before it is allowed to run on the protected system. If an executable file performs a set of actions that is associated with malware, it will be identified by the heuristic engine and its execution will be blocked. Malware authors are aware of the fact that emulators are effective at defeating packers and can identify malicious files quickly. They have the financial resources to buy any antivirus product and study its components. This is why we expect to see more malware that includes routines to hamper detection under emulation. These routines will either try to detect the presence of an emulator and stop executing or will simply try to crash the emulator.

Since they have linked all their infected hosts into botnets, malware creators can easily maintain their creations. They can add new functions and components to thousands of infected hosts in minutes. They can also react quickly to antivirus signature updates and

modify their binary files to evade detection. One aspect of the command and control communication between malware agents and their controller that is often overlooked, is that the controllers can now monitor the performance and health of their botnets in real time. We believe that performance evaluation of malware as a means of maintaining a botnet will gain in popularity. This constant feedback will help malware authors to identify quickly which characteristics of their malware perform well, and which should be improved. In short, feedback from malware to its creators will help malware to evolve faster.

Most malware detection mechanisms work by monitoring modifications that are made to a system or files that are written to disk. It is more than conceivable that we'll see more malware that will never touch the hard drive of its victim and will never alter the operating system. One way in which such malware might work is to execute in memory for as long as it takes to attain its objectives, and erase itself. For example, malware could execute on a computer, steal all e-mail addresses and stored passwords, send them to the botnet controller and then completely erase itself. This behavior would make detection harder, and would also create situations where a victim never finds out that their systems have been infected and that data has been stolen. Indeed, a similar technique was used by the "Code Red" worm, which only ever existed in the memory of the system. To a large extent, it was in reaction to this earlier threat that many anti-malware systems started to apply greater attention to the examination of network traffic streams.

There is Hope

We have made it clear that malware techniques are evolving quickly, even more so since there are now huge profits to be made. Defensive strategies and techniques are also evolving to counter these threats. Numerous actions can be performed to slow down or stop malware infections. We think that user education is a key step toward better defense. By educating users on the dangers of web browsing and social engineering, improved popular awareness of the threats will reduce the number of infected computers. This, however, is not something that can simply fall into place without effort, and will probably require the introduction of computer security awareness training during a child's earliest days of computer use.

Antivirus/anti-malware solutions are also part of the defensive equation. They have evolved greatly in the last twenty years. Antivirus software is now equipped with emulators to counter obfuscation by packers. They also often include firewall modules that will block any unwanted or suspicious external connections. Rootkit detection has been implemented to identify hidden files and processes. Furthermore, signature based detection is only a small

part of modern detection. Generic signatures and heuristic detection are frequently used to detect new malware variants that have not yet been analyzed by antivirus companies.[15]

Finally, frequent application of software patches for all installed components on a system is essential to maintain security. By keeping software patched and up-to-date, system administrators reduce the available attack surface that malware is able to exploit in order to compromise a system.

Conclusions

Digital threats have evolved from a bizarre hobby for computer geeks into a huge source of profit for organized cyber-crime. From a technical point of view, malware has diversified to enable pursuit of a wide variety of criminal objectives and has increased its capabilities for penetration into victim systems. Recognizing the potential profit, we predict that malware will continue to be a threat to IT infrastructures for the foreseeable future. Malware will continue to evolve, and malware authors will try to improve the stealth and persistence of their creations. On the other hand, defensive measures are also constantly improving. User education, regular software updates and patches and sophisticated defensive software are keys to the solution.

References

1. Ford, R. (2007), http://www.npr.org/templates/story/story.php?storyId=11954260
2. Cohen, F. (1984), Computer Viruses-Theory and Experiments
3. Szor , P. (2005), The Art of Computer Virus Research and Defense. Addison-Wesley
4. Retrieved from http://www.wildlist.org/faq.htm
5. Nazario, J. (2007, 3 7). Nirbot - Even Botters Need Attention. Retrieved from Arbor Networks: http://asert.arbornetworks.com/2007/03/nirbot-even-botters-need-attention/
6. Harley, D. et al. (2007). AVIEN Malware Defense Guide. Syngress
7. Schiller, C. et al. (2007) Botnets: the Killer Web App. Syngress
8. Lovet, G. (2006). Dirty Money on the Wires: the business models of cyber criminals. Proceedings of the 16th Virus Bulletin International Conference.
9. Nazario, J. (2001). The Future of Internet Worms. Crimelabs Research
10. Neil Daswani, M. S. (2007). The Anatomy of Clickbot.A.
11. Mirkovic J. & Reiher P. (2004) A taxonomy of DDoS attack and DDoS defense mechanisms. SIGCOMM Compututer Communication.
12. Stewart, J. (2004, 3 15). PhatBot Trojan Analysis. Retrieved from Secure Works: http://www.secureworks.com/research/threats/phatbot/
13. Canavan, J. (2005). The Evolution of Malicious IRC Bots. Proceedings of the 15th Virus Bulletin International Conference
14. Kerbs, B. (2007, 09 28). Just How Bad is the Storm Worm. Retrieved from Washington Post: http://blog.washingtonpost.com/securityfix/2007/10/the_storm_worm_maelstrom_or_te.html
15. Stewart, J. (2007, 2 8). Storm Worm DDoS Attack. Retrieved from Secure Works: http://www.secureworks.com/research/threats/storm-worm/?threat=storm-worm
16. Markus F.X.J. Oberhumer, L. M. (2007, 07 31). UPX, the Ultimate Packer for eXecutable. Retrieved from UPX, the Ultimate Packer for eXecutable: http://upx.sourceforge.net
17. Harley, D. & Lee, A. (2007, 3). ESET Heuristic Analysis Report. Retrieved from ESET ThreatCenter: http://www.eset.com/download/whitepapers/HeurAnalysis(Mar2007)Online.pdf
18. Harley, D. & Lee, A. (2007, 6). A Pretty Kettle of Phish. Retrieved from ESET ThreatCenter: http://www.eset.com/download/whitepapers/Phishing(June2007)Online.pdf

Further Reading

Brosch, M. M. (2006). Runtime Packers: The Hidden Problem? Las Vegas: BlackHat. Retrieved from http://www.blackhat.com/presentations/bh-usa-06/BH-US-06-Morgenstern.pdf

Hoglund, G. and Butler, J. (2005). Rootkits: Subverting the Kernel Rootkit. Addison Wesley.

Nicolas Ianelli, A. H. (2005, 12 1). Retrieved from Botnets as a Vehicle for Online Crime: http://www.cert.org/archive/pdf/Botnets.pdf

Corporate Headquarters

ESET, spol. s r.o.
Aupark Tower
16th Floor
Einsteinova 24
851 01 Bratislava
Slovak Republic
Tel. +421 (2) 59305311
www.eset.sk

Americas & Global Distribution

ESET, LLC.
610 West Ash Street
Suite 1900
San Diego, CA 92101
U.S.A.
Toll Free: +1 (866) 343-3738
Tel. +1 (619) 876-5400
Fax. +1 (619) 876-5845
www.eset.com

© 2009 ESET, LLC. All rights reserved. ESET, the ESET Logo, ESET SMART SECURITY, ESET.COM, ESET.EU, NOD32, VIRUS RADAR, THREATSENSE, THREAT RADAR, and THREATSENSE.NET are trademarks, service marks and/or registered trademarks of ESET, LLC and/or ESET, spol. s r.o. in the United States and certain other jurisdictions. All other trademarks and service marks that appear in these pages are the property of their respective owners and are used solely to refer to those companies' goods and services.

EFTPWP220081111