symantec.

Striking Similarites: Win32/Simile and Metamorphic Virus Code

by Frédéric Perriot
Senior Software Engineer

Péter Ször
Architect

Peter Ferrie
Principal Software Engineer

INSIDE

INSIDE

- Replication Routine
- EPO Mechanism
- Polymorphic Decryptor
- Metamorphism
- Replication
- Payload
- W32/Simile

Contents

Introduction

Win32/Simile is the latest “product” of the developments in metamorphic virus code. The virus was released in the most recent 29A, #6, issue in early March 2002. The virus writer, who calls himself "The Mental Driller," wrote this virus. Some of his previous viruses, such as Win95/Drill (which used the Tuareg polymorphic engine), have proved challenging to detect.

Win32/Simile moves yet another step up the scale of complexity. The source code of the virus is approximately 14,000 lines of assembly code. The metaphoric engine itself takes up approximately 90% of the virus code, which is extremely powerful.

The author named the virus "MetaPHOR," which stands for Metaphoric Permutating High-Obfuscating Reassembler.

The first generation virus code is approximately 32KB, and there are three known variants of the virus in circulation. Certain AntiVirus (AV) companies from some major corporations in Spain received samples of the original variant, which was released in issue 29A, indicating a minor outbreak.

Win32/Simile is highly obfuscated and challenging to understand. The virus attacks the disassembling, debugging, and emulation techniques, as well as the standard evaluation-based techniques for virus analysis. As with many other complex viruses, Simile uses EPO techniques.

Replication Routine

Simile contains a basic, direct action replication mechanism that attacks PE files on the local machine and the network. The emphasis is clearly on the metamorphic engine, which is unusually complex.

EPO Mechanism

The virus searches and replaces all of the possible patterns of certain call instructions (those that reference ExitProcess() API calls) to point to the beginning of the virus code. Thus, the main entry point of the file is not altered.

Sometimes the metamorphic virus body is placed together with a polymorphic decryptor at the same location within the file. In other cases, the polymorphic decryptor is placed at the end of the code section, while the virus body is placed in another section. This is to further confuse the location of the virus body.

Polymorphic Decryptor

During the execution of an infected program, when the instruction flow reaches one of the hooks that the virus has placed in the code section, control is transferred to a polymorphic decryptor, which is responsible for decoding the virus body (or for directly copying it, as the virus body is not always intentionally encrypted).

This decryptor, whose location in the file is variable, allocates a large chunk of memory (about 3.5 megabytes), then proceeds to decipher the encrypted body into it. It does this in a most unusual manner: rather than going through the encrypted data linearly, it processes it in a seemingly random order, thus managing to avoid triggering some of the decryption-loop recognition heuristics.

This “Pseudo-Random Index Decryption,” as the virus writer calls it, relies on the use of a family of functions that have interesting arithmetic properties, modulo 2^n.

While the virus writer discovered this by a process of trial and error, it is possible to produce a mathematical proof that his algorithm works in all cases (provided that the implementation is correct, of course). Such proof is beyond the scope of this article but the proof, by Frédéric Perriot, is available at http://www.peterszor.com/.

The size and appearance of the decryptor varies greatly from one virus sample to the next. To achieve this high level of variability, the virus writer simply generates a code template and then puts his metamorphic engine to work to transform the template into a working decryptor!

In some cases, the decryptor may start with a header whose intent is not immediately obvious upon reading it. Further study reveals that its purpose is to generate anti-emulation code on the fly: the virus constructs a small oligomorphic code snippet containing the instruction “ReaD Time Stamp Counter” (RDTSC). This retrieves the current value of an internal processor ticks counter. Then, based on one random bit of this value, the decryptor either decodes and executes the virus body or bypasses the decryption logic altogether and simply exits.

Besides confusing emulators that would not support the somewhat peculiar RDTSC instruction (one of the Mental Driller's favorites, which he used previously in Win95/Drill), this is also a strong attack against all algorithms that rely on emulation either to decrypt the virus body or to determine viral behavior heuristically. Effectively, it causes some virus samples to completely cease infecting upon a random time condition.

On initial execution, the virus body will retrieve the addresses of 20 APIs that it requires for replication and for displaying the payload. Then, it will check the system date to determine whether either of its payloads should activate. Both payloads require that the host imports functions from User32.dll. In this case, the virus checks whether it should call the payload routine (which will be explained later).

Metamorphism

After the payload check has completed, a new virus body is generated. This code generation is carried out in a number of steps:

1.) Disassembles the viral code into an intermediate form, which is independent of the CPU upon which the native code will execute. This allows for future extensions, such as producing code for different operating systems or even different CPUs.

2.) Shrinks the intermediate form, by removing the redundant and unused instructions. Earlier replications added these instructions to interfere with the disassembly by virus researchers.

3.) Permutates the intermediate form, for example reordering subroutines, or separating blocks of code and linking them with jump instructions.

4.) Expands the code, by adding redundant and unused instructions.

5.) Reassembles the intermediate form into a final native form that will be added to infected files.

Thus, Simile cannot only expand, as most first generation metamorphic viruses do, but it can also shrink (and shrink to different forms!).

Replication

The replication phase begins next. It starts by searching for *.exe in the current directory, then on all the fixed and mapped network drives. The infection will recursively scan into directories, but only to a depth of three subdirectories, completely avoiding any directory that begins with the letter "W."

For each file that is found, there is a 50% chance that it will be skipped explicitly. Additionally, files will be skipped if they begin with "F-," "PA," "SC," "DR," "NO," or contain the letter "V" anywhere in the name.

Due to the nature of the comparison, other character combinations are skipped unintentionally, such as any directory that begins with the number 7, or any file that begins with “FM,” or any file that contains the number 6 anywhere in its name.

The file infection routine contains many checks to filter files that cannot be infected safely. For example, the file must contain a checksum, it must be an executable for the Intel 386+ platform, and there must exist sections whose names are ".text" or "CODE", and ".data" or "DATA." The virus also checks that the host imports some kernel functions, such as "ExitProcess."

For any file that is considered infectable, random factors and the file structure will determine where the virus places its decryptor and virus body.

If the file contains no relocations, or by small chance, the virus body will be appended to the last section in the file. In this case, the decryptor will be placed either immediately before the virus body, or at the end of the code section.

Otherwise, if the name of the last section is ".reloc," the virus will insert itself at the beginning of the data section and move all of the following data and update all of the offsets in the file.

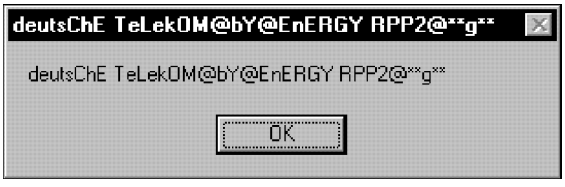
Payload

The first payload activates only during March, June, September, and December. Variants A and B of Win32/Simile display their message on the 17th day of these months. Variant C will display its message on the 18th day of these months.

Variant A will display the message "Metaphor v1 by The Mental Driller/29A":

and variant B will display "Metaphor 1b by The Mental Driller/29A":

Variant C attempts to display "Deutsche Telekom by Energy 2002 **," however, the author of that variant had little understanding of the code, so the message rarely appears correctly.



In all variants, the message appears in randomly mixed letter cases.

The second payload activates on 14 May in variants A and B , and on 14 July in variant C.

In the second payload, variants A and B will display the message “Free Palestine!” on computers that use the Hebrew locale. Variant C attempts to display the text "Heavy Good Code!" although, due to a bug in the virus code, this message is only displayed on systems on which the locale cannot be determined.

Conclusion

As the saying goes: "There are three kinds of lies: lies, damn lies, and statistics." This proved to be the painful truth while attempting to come up with a reliable detection for all the replicants of Simile, which were produced on a variety of systems, and using a wide variety of host programs.

During the extensive and detailed tests executed with the W32/Simile replication on the test systems, we noticed that the virus code either unintentionally generates garbage or accidentally trashes some files as a direct result of its extreme complexity.

It seems that obfuscated code is not only challenging for virus researchers to analyze, but it is very challenging for the author of the code to debug. The complex infection mechanism coupled with the powerful metamorphic engine makes it difficult to reach 100% accuracy, based on evaluation techniques, and thus in-depth analysis of the virus code is essential.

A quick solution delivery for metamorphic virus detection should become a huge team effort at AV companies, as the number of complex metamorphic viruses is growing slowly but surely.

Exact identification becomes a problem even for humans. How long does it take to be sure whether something is really variant A or C, or a new one? Is it modified or is it the same? It is becoming increasingly difficult to know. The need to gain an understanding of metamorphic code more quickly must be the subject of further research.

W32/Simile

Alias: W32.Etap, Metaphor.

Type: Directs action Win32, portable executable infector, complete metamorphic virus.

Removal: Detects and deletes infected files and replaces them from clean backups.

Payload: Displays messages on certain dates.

Unintended payload: Trashes some portable executable files.

SYMANTEC, THE WORLD LEADER IN INTERNET SECURITY TECHNOLOGY, PROVIDES A BROAD RANGE OF CONTENT AND NETWORK SECURITY SOFTWARE AND APPLIANCE SOLUTIONS TO INDIVIDUALS, ENTERPRISES AND SERVICE PROVIDERS. THE COMPANY IS A LEADING PROVIDER OF VIRUS PROTECTION, FIREWALL AND VIRTUAL PRIVATE NETWORK, VULNERABILITY ASSESSMENT, INTRUSION PREVENTION, INTERNET CONTENT AND EMAIL FILTERING, AND REMOTE MANAGEMENT TECHNOLOGIES AND SECURITY SERVICES TO ENTERPRISES AND SERVICE PROVIDERS AROUND THE WORLD. SYMANTEC'S NORTON BRAND OF CONSUMER SECURITY PRODUCTS IS A LEADER IN WORLDWIDE RETAIL SALES AND INDUSTRY AWARDS. HEADQUARTERED IN CUPERTINO, CALIF., SYMANTEC HAS WORLDWIDE OPERATIONS IN 38 COUNTRIES.

FOR MORE INFORMATION, PLEASE VISIT WWW.SYMANTEC.COM

About the Authors

Frédéric Perriot received his degree in Engineering from the Ecole Centrale de Paris, with a major in Real-time Computer Science. Thereafter, he joined the Antivirus department of the IBM Watson Research Center in New York where he worked on the Immune System project from 1997 to 1998.

In 1999, Frédéric was awarded a Diplome d'Etudes Approfondies (DEA, which is equivalent to a Master's degree) from the Université Pierre et Marie Curie in Paris, in the field of Semantics of Programming Languages and Formal Proofs. He worked as a network administrator for the French embassy in Niger as part of his national service in 2000.

Frédéric joined Symantec Corporation in 2001 where he works in the Security Response team as a Blended Threats Analyst. Frédéric specializes in systems programming and reverse engineering.

Peter Ferrie is a Senior Virus Researcher at Symantec Security Response. He specializes in the detection and repair of Win32 malware, reverse-engineering file formats, and the development of engine enhancements for Symantec AntiVirus products.

Peter contributes to Virus Bulletin magazine. He joined the Computer Antivirus Research Organisation (CARO) in 2001.

Péter Ször graduated from the University of Veszprem Hungary in 1991. He is the author of a popular Hungarian virus scanner called Pasteur, which he developed between 1990 and 1995. Szor's interest in computer viruses began in 1990. He worked on various antivirus scanning engines over the last decade including F-PROT, AVP, and Norton AntiVirus. Peter was invited to join the Computer Antivirus Research Organisation (CARO) in 1997.

He is a frequent speaker at Virus Bulletin, EICAR, and ICSA conferences, as well as a regular contributor to Virus Bulletin magazine. In 1999, Szor joined Symantec Corporation where he designs and develops antivirus technologies for the Norton AntiVirus product line. He is the author of several pending U.S. patents.

WORLD HEADQUARTERS

20330 Stevens Creek Blvd.
Cupertino, CA 95014 U.S.A.
408.517.8000
800.721.3934

www.symantec.com

For Product Information
In the U.S., call toll-free
800.745.6054

Symantec has worldwide operations in 38 countries. For specific country offices and contact numbers please visit our Web site.