Malware Detection

Antiviral shortcomings with respect to 'real' malware

Gary Golomb <gary@proventsure.com>
President, Proventsure LLC, Baltimore, MD
Jonathan Gross <jgross1@gwu.edu>
Senior Forensics Engineer, George Washington University, Washington, DC
Rich Walchuck < rwalchuck@patriot-tech.com>
Executive Director of Professional Services, Patriot Technologies, Frederick, MD

Publish date: 3/15/2007
Updated: 5/5/2007 - Typo in previous *Abstract* and added to discussion on this testing methodology

ABSTRACT

Motivation: Computer forensic investigators and people commonly performing incident response have known for some time that antivirus products are poor detectors of malware currently found on compromised computers. Extensive use of binary executable packers and obfuscators by hackers has rendered analysis of compromised systems into a largely manual process. This paper describes the effectiveness of new and established products on detecting malware retrieved from actual compromised hosts. Many organizations assume they are perfectly safe because they run anti-virus and/or firewalls on user's systems. One purpose of this paper is to evaluate this posture.

Methods: Thirty-five backdoors and related malware binaries were harvested from live compromised computers from December 22, 2006 to February 20, 2007. On March 9, 2007, all 35 binaries were scanned by 32 different up-to-date commercial, freeware, and open-source virus scanners to assess their accuracy in detecting the malware found on these computers.

Results: The average detection rate among the different scanning products was 33%. Asarium had the highest detection rate with 75%. Panda Software had the next highest detection ratio at 50%. Two products tied for the lowest detection rate at 6%. The most shocking and unexpected result was seeing that 30 of the 32 products had a less than 50% detection rate.

Contact: gary@proventsure.com

1 INTRODUCTION

At a macro level, most large organizations believe their enterprise-level antivirus deployments are providing a good, or at least *acceptable* level of protection against the risks posed to the data stored on computers throughout the organization. Indeed, antivirus products are commonly sold to consumers with marketing claims such as the following, "You can expect 100% detection of In-the-Wild viruses (viruses already spreading between users) and excellent detection of Trojan horses." [1] With a perception as pervasive as this, it is no wonder enterprises commonly prioritize resources on protecting their datacenter as opposed to the overabundance of desktops spread throughout an organization.

Although there are a number of claims to being the first antivirus product, the first major products seem to have been "unvirus" and "immune" developed by the Hebrew University in 1988[2]. By 1990, the market had matured to 19 different products, with large vendors entering the fray through acquisitions in 1992[3].

Since then, products have evolved to use three core methods of detection. Most products currently used today implement a combination of all three methods for increased effectiveness.

The first method is known as *dictionary* searching, and involves searching files for binary-level "signatures" of known malware. Signatures are strings of bytes in infected files and not in uninfected files. Signature matching methods are useful in many cases, but also easily evaded using *polymorphic* and *metamorphic* code. Polymorphic code changes the byte-level representation of the work it needs to accomplish while keeping the underlying code-morphing algorithm intact. Metamorphic code, on the other hand, actually reprograms itself in its entirety, including the mutation engine, every time it is executed[4]. This theoretically makes detection more difficult (although detection algorithms can still be developed for the reprogramming functions).

The second method scans for *suspicious behavior*. These behavior detection mechanisms can be implemented a number of different ways, from a shim that watches for certain

[1] (2007). "avast! 4 Home Edition." Retrieved from http://www.avast.com/eng/avast_4_home.html. Retrieved on 3/13/2007

[2] Neumann, P.G. (1988). "*Forum on Risks to the Public in Computers and Related Systems.*" ACM Committee on Computers and Public Policy. Retrieved from http://catless.ncl.ac.uk/Risks/6.6.html. Retrieved on 3/13/2007.

[3] Szor, P. (2005). "*Virus Research and Defense*." New Jersey: Symantec

[4] The Mental Driller (2002). "*Metamorphism in practice or 'how I made MetaPHOR and what I've learnt'*" 29A#6 Retrieved from http://vx.netlux.org/lib/vmd01.html Retrieved on 3/13/2007

system calls processes make, all the way to implementing a *sandbox* and running every process inside. Sandbox technology executes binaries in a *virtual machine*, such that the binary process thinks it is running inside the operating system, when in actuality it is running inside of another process. The sandbox process then proxies every system call of the binary process and monitors, logs, and creates alerts about the binaries activities.

The third class is generically referred to as *heuristics*, and encompasses many different ways of determining if a file is malware related. Most heuristic methods perform an analysis of the binary file format or flow control during execution to determine if the binary is exhibiting behavior typical of binaries trying to protect themselves from antivirus products. Heuristics also encompasses identifying suspicious functions related to malware at the assembly level.

Malware authors are aware of how current antiviral applications function, and will generally modify their code accordingly to exclude commonly detected functions. Binary packers and crypters are also frequently used to obfuscate the contents of malicious binaries. These seemingly simple methods are generally enough to bypass most virus scanning tools.

Given all this, it has become common practice for security engineers and incident responders to become accustomed to analyzing hacked computers with fully active backdoors that also have updated antivirus and host-based firewall products, which have detected nothing anomalous. In these situations, since traditional scanning applications are blind to the presence of this malware, the location and analysis of these binaries has become a largely manual process requiring specialized skills in identifying the fingerprints of a compromised computer. With such a specialized skill set required, most organizations without the budget for full-time forensic engineers rely primarily upon antivirus products installed throughout their organization are at a much higher risk for harboring compromised computers.

Described in this paper is a test of 32 commercial, freeware, and open source updated virus scanners to assess the accuracy in detecting malware, and a deeper analysis of the results of those tests.

2 EXPERIMENTAL DESIGN

From a period of December 22, 2006 to February 20, 2007, binaries were collected from compromised hosts in live networks. Because these were real-world systems and not honeypots, low-level details of the compromises cannot be legally disclosed; however, the authors can make the archive of harvested backdoors available upon request. A total of 35 backdoors were collected. In terms of a *representative* dataset, this is a quite limited design, and is examined in the *Discussion* section of this paper. The collection period was limited to three months in order to balance concerns with time-based relevancy of the sample set. Backdoors that are too new would be a greatly unfair test, while those that are too old would be moot for the point of the test.

All of these systems were originally detected as being potentially compromised through network IDS signatures identifying backdoor or bot-related traffic. Each system was manually audited, and the main offending binaries were removed and collected. Only binaries related to the actual running backdoor processes were harvested.

Thirty-two products were used to scan each of the binaries. The products and their revisions are shown in *Table 1* and are sorted in descending order of version number, with the most mature products (by version) at the top of the list. We waited over two weeks to perform the scan, to give products a chance to release signature updates that could have affected the results of scanning the backdoors collected over the previous three months.

Table 1. Malware Detection Products Used

Scanner	Version	Update Date (today: 3/9/2007)
McAfee	4981	Same day as scan
NOD32v2	2105	Same day as scan
eTrust-Vet	30.6.3467	Same day as scan
Symantec	10	Same day as scan
Panda	9.0.0.4	Same day as scan
CAT-QuickHeal	9	Same day as scan
AVG	7.5.0.447	Same day as scan
AntiVir	7.3.1.41	Same day as scan
BitDefender	7.2	Same day as scan
eSafe	7.0.14.0	Same day as scan
F-Secure	6.70.13030.0	Same day as scan
TheHacker	6.1.6.073	Same day as scan
Norman	5.80.02	Same day as scan
Authentium	4.93.8	Same day as scan
Avast	4.7.936.0	Same day as scan
DrWeb	4.33	Same day as scan
VirusBuster	4.3.19:9	Same day as scan
F-Prot	4.3.1.45	Same day as scan
Sophos	4.15.0	Same day as scan
Kaspersky	4.0.2.24	Same day as scan
Ewido	4	Same day as scan
VBA32	3.11.2	Same day as scan
Ikarus	3.1.1.3	Same day as scan
ClamAV	20060426(devel)	Same day as scan
Fortinet	2.85.0.0	Same day as scan
Sunbelt	2.2.907.0	Same day as scan
Prevx1	2	Same day as scan
UNA	1.83	Same day as scan
Microsoft	1.2204	Same day as scan
Proventsure Asarium	1.1	1/22/2007
FileAdvisor	1	Same day as scan
PEiD	0.94	5/10/2006

For scanning, the independent third-party services of Virustotal[5] were used. Virustotal maintains real-time automatic updates of virus signatures for each product and returns detailed results from each antivirus engine. Real time global statistics are available from the Virustotal website. There are certain limitations to using Virustotal as the scanning platform, and these limitations are discussed in the *Discussion* section of this paper.

Asarium and PEiD are not maintained by Virustotal, so the sample files were scanned manually by the authors. PEiD was used with its default detection database, and configured to run a deep scan. Asarium was previously developed by one of the authors of this paper. Both Asarium and PEiD were included to illustrate the discussion of generically detecting malware by examining file structures as opposed to more traditional methods. Table 2 contains a file hash listing of the malware files examined.

Table 2. Malware File Hashes

MD5 File Hash
7E-FC-FD-65-77-C4-26-E3-50-2F-35-2C-78-5D-10-E5
C7-10-84-D9-95-8F-82-53-75-CA-57-3C-A7-E2-47-92
E5-A7-05-A9-8D-A4-B9-1E-BD-D9-3D-CC-A5-30-E8-0A
83-86-88-BC-58-FD-1A-B0-11-53-2B-52-7F-DE-A1-14
39-19-E7-AC-D0-3C-53-20-9F-F8-96-EF-3D-6C-5D-7A
5C-9C-1B-2C-46-A5-5F-0E-63-02-BE-63-0F-4F-94-FF
57-B8-23-62-51-91-FA-FC-01-B8-01-96-75-D7-F7-4D
BF-25-44-2B-93-D6-07-78-BC-C7-60-F7-64-95-1F-12
D1-13-3C-32-41-3D-D8-06-A1-F6-C1-56-FE-F1-55-21
BD-56-B8-BD-DF-B9-FC-1E-14-83-45-D9-AB-90-05-12
C8-5D-16-4F-90-10-1D-64-4A-F1-15-77-B1-1F-D3-0A
47-5B-EF-F4-F4-50-44-E7-2F-CD-52-1D-84-46-DB-AC
10-D8-BE-76-16-28-7C-9B-97-5E-C5-21-14-CC-54-C3
C6-FD-E3-A5-7C-40-00-86-05-89-16-B4-33-C6-02-28
02-D2-FD-B2-B3-FD-ED-C1-98-55-C1-15-30-3C-6C-E9
F9-8F-22-97-B9-5C-3F-B1-19-1F-4D-01-0E-C6-DB-24
F4-26-01-D4-AC-18-BB-06-D8-30-B6-F8-E4-50-0A-DF
C5-E2-F8-E2-3E-F4-E1-CD-EE-46-94-6E-07-34-5A-BE
40-4D-25-8E-63-A7-08-32-0B-1B-21-43-DB-25-DA-6A
BA-E2-2C-30-AD-30-0A-88-51-05-06-B3-23-66-34-40
C9-DF-A0-E4-F4-A6-40-9A-AA-49-B8-91-C8-A8-68-F2
A1-5E-B9-51-9F-80-F8-2B-BE-B0-FC-E9-C6-5D-20-CB
03-E5-D9-F3-3C-D8-0D-11-7B-69-20-6C-B7-E7-B3-E2
EC-43-99-88-D5-94-BE-E8-0F-62-BB-3B-5B-02-48-1A
A7-A5-23-2B-CD-EC-3D-69-AC-D8-3E-D4-18-59-F9-A0
2C-DE-A2-29-AE-03-BE-84-5A-AF-85-5E-C4-DD-F4-D6
BC-10-DA-51-1C-B4-87-B1-1B-15-1D-58-89-8B-5E-AA
97-42-B6-E2-10-0C-4E-2C-AD-EE-9F-74-31-73-52-7E
54-1A-CA-15-58-D4-C6-BF-E5-3C-F1-CA-36-BA-11-B1
8B-9A-07-E3-FB-D6-68-10-CE-07-85-E8-E6-18-B2-6F
D6-A5-EE-48-F0-AB-68-55-97-A5-81-51-66-D5-E2-CC
AA-FB-91-10-FD-58-3A-95-AF-0B-3F-09-81-73-86-09
E4-CB-20-46-52-23-B8-BF-2D-D6-85-7B-1B-65-7A-9F
10-36-E3-DD-DC-89-A4-E6-8D-8A-33-F3-82-3A-18-0E
C9-A9-0E-78-A4-8D-3E-41-A8-EA-4C-47-A2-94-9F-73

[5] See http://www.virustotal.com for more information.

Because every product detects malware differently, and triggers alerts on different aspects of malware, testing accounted for *positives* and *false negatives* only. In other words, if a particular product returned any indication the file was malware related, credit was given to the product for identifying the file as malware. If the product generated zero alerts about the file scanned, then it was counted as a missed result.

For the purpose of this test, the terms backdoor and malware are used synonymously, and are used to describe software that is non-commercially supported, maliciously installed, and allows full access to the data, remote management, and administration of the system it is installed on.

3 RESULTS

Of the 35 malware files, three invalid files were removed from the sample set, leaving 32 malware binaries used in the final tests and performance calculations. If the malware binary would not consistently execute on a computer, then it was removed from the test group due to inconsistencies in detection results.

Table 3 (below) and Graph 1 (Appendix) summarize the results of the testing. The average detection rate among all products was 33%. The highest was Asarium with a 75% detection rate, and the lowest was tied between ClamAV and FileAdvisor with a 6% detection rate.

The most shocking and unexpected result was seeing that 94% of the products had a less than 50% detection rate.

Table 3. Scanner Results

Scanner	Results of 32	Percent
Proventsure Asarium	24	75
Panda	16	50
eSafe	15	47
AntiVir	14	44
Fortinet	14	44
BitDefender	14	44
F-Secure	13	41
F-Prot	13	41
VBA32	13	41
Symantec	13	41
AVG	12	38
Ikarus	12	38
Sunbelt	12	38
Norman	11	34
Authentium	11	34
Kaspersky	11	34
NOD32v2	11	34
CAT-QuickHeal	10	31
Prevx1	10	31
UNA	10	31

Avast	9	28
McAfee	9	28
DrWeb	9	28
Ewido	9	28
PEiD	9	28
VirusBuster	7	22
Sophos	7	22
eTrust-Vet	7	22
TheHacker	6	19
Microsoft	4	13
ClamAV	2	6
FileAdvisor	2	6

4 DISCUSSION

The first and foremost point to make about the purpose of the test and paper is not to establish which products detect the most problems in a sample of binaries. The purpose of the paper is to evaluate the performance of market-leading products against malware sampled from production environments in the real world. This is a subtle difference and one of important distinction. Many organizations assume they are perfectly safe because they run antivirus and/or firewalls on end user's systems. One purpose of this paper is to evaluate this posture.

One limitation in this test was the use of Virustotal as a scanning platform. VirusTotal AV engines are command line versions. This can introduce differences between Virustotal results and those seen from the stand-alone versions - especially in products that rely on personal firewall logging as well.

Many published tests are based on "Zoo" collections of hundreds or thousands of binaries. It is commonly known these collections are filled with corrupt executables and false positives. Additionally, AV companies help contribute to these collections, which make them terribly skewed in the first place. The attempt here was to test products against malware coming entirely from live compromises of end-users and not vendors.

However, a limitation of this experiment is the size of the sample data, although it was chosen intentionally. After collecting binaries for three months, there were dozens more to use, however files with duplicate hashes were removed, which greatly reduced the sample size. These malware binaries are all binaries taken from recently hacked computers that allow remote access to the computer by outside parties. They are not the types of files virus scanners were originally developed to scan for, such as Office Document macro viruses or mass mailing viruses. Despite this, these types of files are the most regularly found causes of compromise today. If macro viruses had been scanned, it is expected the results of certain products would go down severely, while others would increase dramatically.

Given this constraint, binaries used in the sample set were selected intentionally. This is the malware most organizations are unaware of, since many IDS's and other network auditing products are currently unequipped to reliably detect all of them. IDS is constrained principally to unencrypted traffic, hence custom protocol p2p and traditionally encrypted backdoors will typically evade detection. In addition, these backdoors are one of the most serious threats to organizations, since they all allow some form of remote control and/or administration of the systems they are installed on.

One of the big shifts new vendors are taking in malware detection is examining binary formats as opposed to simply looking for signs of specific viruses. For example, the Portable Executable (PE) is the file format specification used for executables and .dll's on 32-bit and 64-bit Windows operating systems[6]. An executable file consists of a number of headers and sections, which jointly direct how the file is ultimately mapped into memory. These sections are aligned to page boundaries to account for regions which require specific memory protection specifications. These protection specifications include information such as whether or not the area is writable or executable. One particular section of note is the import address table (IAT), which is used as a reference for the application when it is calling a Windows API function. Most packing or compressing programs will destroy the original IAT to free unnecessary space; the Windows loader is extremely forgiving with PE headers. Virus authors can even potentially abuse the Windows loader itself to hide malicious code which is executed before the actual program in addition to obscuring already present malicious content.

Some products work by scanning the structure of executable files and comparing the header structure to profiles of normal compilers like Visual Studio and Borland. Others work by analyzing information contained in the IAT (described above) and Export Address Tables.

These methods are extremely useful in identifying malware because virus and malware writers typically use binary obfuscators and packers. UPX is the most common example of these, although by no means the most stealthy[7].

These methods are also useful since they look for more "universal" signs of binary integrity. Since these facets of binaries do not change frequently, these methods require little updating over time, as opposed to antivirus products which typically need to be updated daily, or even hourly.

The potential drawback of these methods is that they produce alerts that are more general in nature. For example, where Asarium returned an alert with the name " HIGH:ENIGMA-PACKED-BIN" other scanners might have returned a more specific event name indicating the actual backdoor. On the other hand, it was seen that almost all of the products identified each backdoor differently from one another. This lack of consistency between products brings the value of this point into question in the first place. Additionally, with the proliferation of backdoor source code and people with the ability to customize them, it seems to be most useful to simply identify a file as malware regardless of the ability to identify the exact flavor of it. This gives administrators the ability to do something they are currently lacking - the ability to identify the problem and clean the system.

[6] May 21, 2006. "Microsoft Portable Executable and Common Object File Format Specification." Microsoft Corporation. Redmond, Washington.
[7] http://upx.sourceforge.net/

5 SUMMARY

Many organizations assume they are perfectly safe because they run anti-virus and/or firewalls on user's systems.

The average detection rate among 32 different scanning products was 33% when scanning 32 different backdoors harvested from live compromised computers in the wild. Proventsure's Asarium had the overall highest detection rate with 75%. Panda Software had the next highest at 50%, and two products tied for the lowest detection rate at 6%. Perhaps the most disappointing result observed was that 94% of all the products detected less than half of the malware scanned.

The results should serve as a significant revelation to people serving in operational security and executive roles in enterprise information technology groups. Most people are well aware of industry standards and best practices for security by using multiple layers of technology, however many technologies developed years ago are not always best suited for the breadth of problems existing in the wild today. Additionally, as shown in the beginning of this paper, vendors will make claims of performance that are difficult for organizations to validate. The vendor that claimed 100% detection in the quote at the beginning of this paper actually had a 28% detection rate in this tests.

Additionally, as threats continue to evolve, vendors need to step forward with new ideas and solutions to problems, as opposed to consistently being reactionary to new problems after organizations have already fallen victim to them.