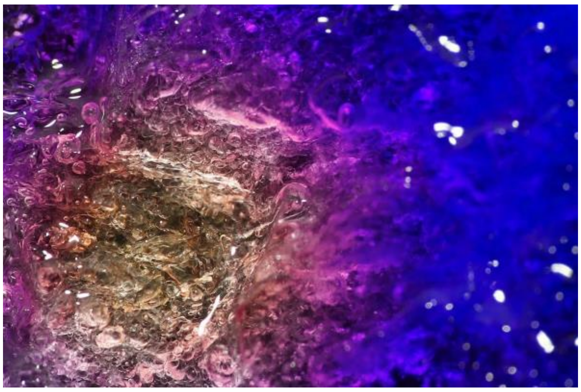


The Achilles Heel of Antivirus

Chia-Wei Wang ,Chong-Kuan Chen, Shiuhpyng Shieh,
Department of Computer Science,
National Chiao Tung University, Hsinchu City, Taiwan
{chiaweiw|ckchen|ssp}@cs.nctu.edu.tw

***Abstract*—Antivirus has become one of the most important security guardians against malware for Internet users. The protection provided by antivirus had great success in detecting malware in the past decade. However, modern malware evolves with mutation and anti-antivirus techniques, thereby effectively hindering the detection. The evidence shows that only 51% of the antivirus software can successfully detect the up-to-date malware. Furthermore, due to the lack of protection, antivirus itself is vulnerable to be disarmed once and for all. In this article, the weaknesses of antivirus software are examined, and the possible solutions are proposed to cope with the problems.**

Keywords—antivirus, malware, anti-AV

I. Introduction

Malware has imposed a noticeable issue of information security ever since the evolution of the computer technology. While computer systems are widely utilized, the sensitive data stored in the storage devices becomes the profit-making target for adversaries. Instead of manually intruding each user's system, adversaries usually spread out the crafted malware in an automatic way as an effective weapon to steal and even sabotage the private data stored in the system.

For instance, the recent CryptoLocker malware, spread in the attachments of social-engineering emails, encrypts all the data in the system storage upon being executed. A victim user is then extorted for the decryption or the data erasure otherwise. In order to prevent users from accessing malware, antivirus software (AV) as the front-line defender is designated to alert users to the malicious file in the real-time paradigm.

In the good old days, AV had great success in the malware detection due to the high barrier to craft new, novel malware, which is undetectable by AVs. However, nowadays there exist various tools that can be utilized to easily mutate an existing malware instance or even customize its functionality for different purposes. Thus, the rapid growth of newly generated malware instances exhausted the resources of AVs to cope with them. Moreover, certain anti-antivirus techniques are also applied to malware for their counterattack against AV. Solely relying on AVs as the only defense against malware is insufficient. In this article, the weaknesses of antivirus are presented, and suggestions as the possible complements to existing AVs are also proposed.

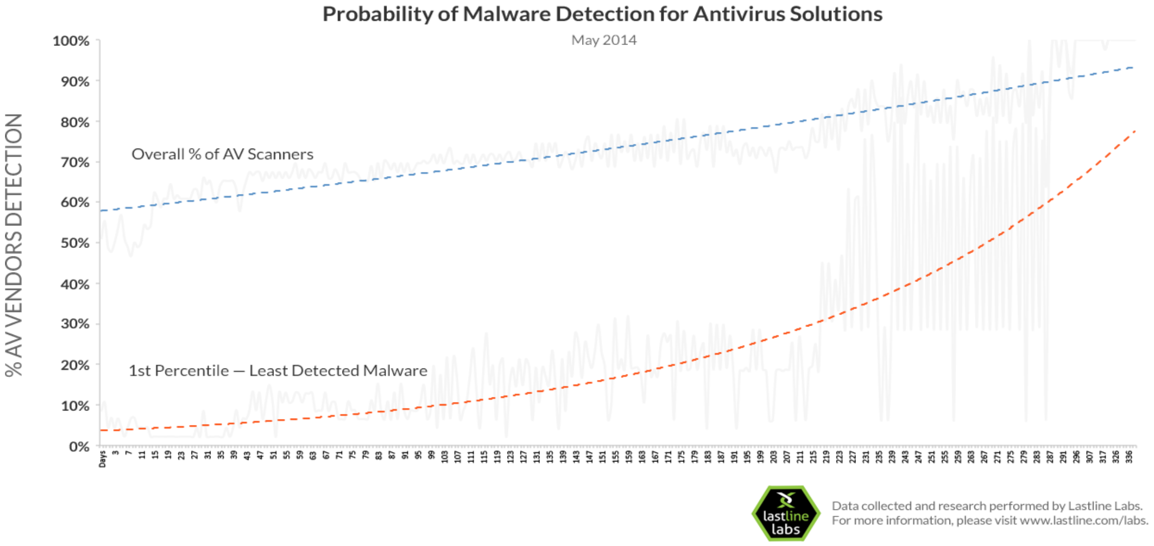
The rest of this paper is organized as follows. Section II overlooks the defects of conventional signature-based detection, which is applied in most AVs against malware. Section III gives the analysis of anti-AV techniques of modern malware. The anticipation of AV vendors is discussed in Section IV. Section V concludes this paper.

II. Pitfalls of Signature-based Detection

Signature-based detection has long been a common approach to the detection of malware. It has high performance and low false positive rate. For each collected malware instance, a security expert analyzes the malware with reverse-engineering techniques. The specific sequence of bytes is then extracted from the malware as its unique signature. A target file matching the signature is recognized as malware. At present, this approach still dominates the detection model of AVs. The effectiveness of signature-based detection is hindered by both malware mutation and the time lag of signature generation. These two factors will be elaborated below, and the solutions will be given.

A. The time race from malware appearence to signature generation

Although the signature-based approach empowers AVs to detect malware, it cannot guarantee the complete coverage of malware detection. One of the main reasons is due to the lack of malware signatures during the time window from the appearance of a new malware instance to its corresponding signature generation. Signature generation is labor intensive and requires significant human effort. For an AV vendor, a malware instance in the wild must be firstly collected so that the corresponding signature for the detection can be extracted. In contrast, malware authors intend to release malware only if the newly generated one does not match any known, recognized signatures of AVs. Consequently, the newly generated malware can evade the detection due to the lack of corresponding signature of AVs. In the undetectable time window, AV's protection is vulnerable and fragile. The active time, which can be referred as the burst of initial infection of malware, is relatively short in the time window. According to the report of FireEye [1], the active time is only two hours. It is crucial for the vendors to equip their AVs with the new signature in such a short period of time. Fig.1 gives the statistical results given by Lastline [2] to illustrate the percentage of existing AVs which successfully detect a new malware instance in terms of days since the first appearance of the malware. The X-axis represents the time window in days whereas the Y-axis denotes the percentage of AVs which



Data collected and research performed by Lastline Labs.
For more information, please visit www.lastline.com/labs.

Fig. 1. Percentage of AVs to detect malware in the respect of the number of days that the malware appeared

successfully detect the malware. As shown in the figure, only 51% of the AV vendors effectively detect new malware encountered in the early stage. Certain least detected, tenacious malware even remains undetected by some AVs for a year.

B. *Write once, mutate everywhere*

To evade the signature-based detection, the mutation techniques are employed by malware authors against AVs. In general, the mutation is realized by replacing the binary code of malware while retaining the equivalent functionality. Therefore, it is not applicable solely using signature with fixed pattern matching to recognize mutated malware. To make detection even harder, malware embedded with a mutation engine is able to self-mutate whenever it propagates. As a result, the “write once, mutate everywhere” strategy adopted by malware authors leads to a large number of undetectable time windows aforementioned, which exhausts AV vendors to develop the signatures for each mutated malware instance [5].

In spite of different appearance of the mutant malware instances, their malicious behavior remains the same. For instance, two mutants from the same origin may both perform the same network transmission. Therefore, the invariant of behavior gives us the opportunity to detect mutated malware. The detection of malware can be significantly improved by using the behavior analysis to complement the conventional signature-based detection.

On the other hand, to shorten the time window before the signature generation, an automatic process should be considered. Heuristic approaches such as machine learning can generate the detection model for known malware automatically. Using the heuristic approaches to detect similar malware, security experts can focus only on the brand new malware. Thereafter, the time windows needed to generate signatures can be diminished.

With runtime behavior as the signature to detect malware, the problem of malware mutation can be mitigated. If a malware instance can be recognized, the effort to further analyze the corresponding mutant can be saved. Meanwhile, the heuristic approach can be conducted to shorten the time windows as aforementioned. With behavior-based signature and the heuristic approach, AVs can be more effective in response to a new malware.

III. ANTI-AV: OFFENSE IS THE BEST DEFENSE

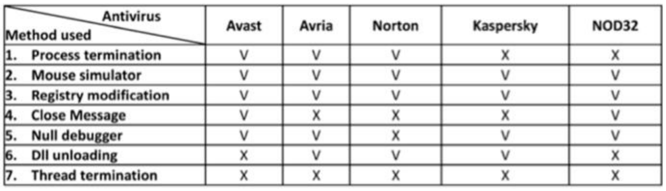
To conceal its activities, malware may attempt to use some anti-AV techniques to attack AVs. We will elaborate these techniques and propose the solutions.

A. *Hook sabotage*

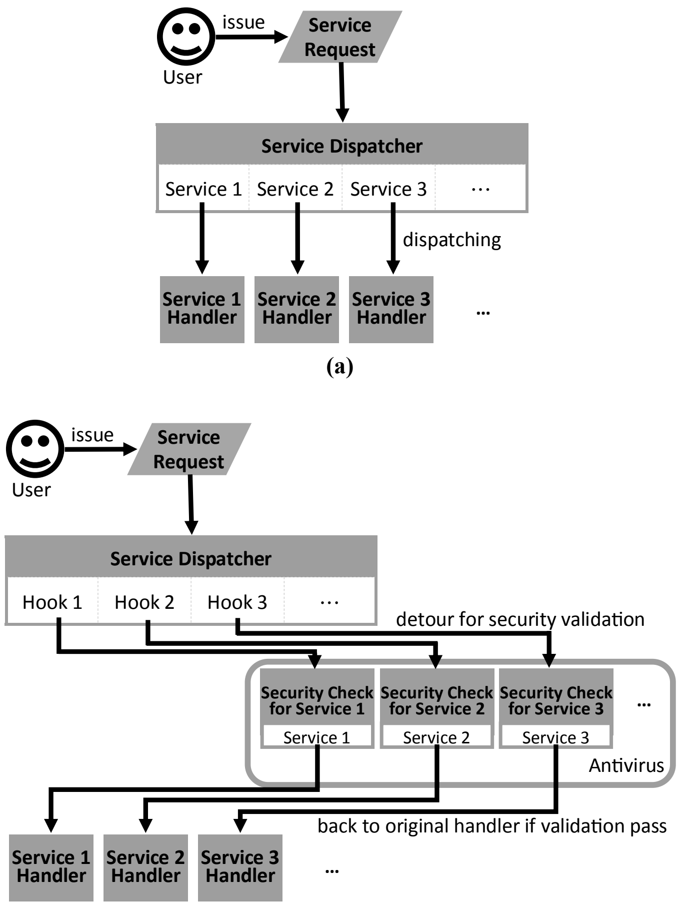
Modern malware is equipped with anti-AV techniques to disarm AVs and thus circumvent their detection. This countermeasure is mostly achieved by sabotaging the checkpoints pre-installed by an AV. Recall that AVs must intercept certain operations issued by users to prevent him/her from mistakenly triggering malware in the real time paradigm. Therefore, an AV usually injects additional checkpoints, also known as hooks, into the system service procedures for the interception.

Figure 2 illustrates the concept of hooks injected by an AV. As shown in Figure 2 (a), the common design of system service procedure of OS consists of service request, service dispatcher, and service handler. Figure 2 (b) gives the system service procedure patched by an AV installed. The hooks injected at the service dispatcher detours a request issued by a user to the security module of the AV before it can be delivered to the service handler. The request for a system service such as the file read/write operation will be validated if any recognized malware is being interacted. The request is forwarded to the original service handler to serve the request only if the validation pass or aborted otherwise.

TABLE I. SEVEN TERMINATOR PROGRAMS AND THE ANTIVIRUS SOFTWARE THAT THEY CAN CLOSE



Antivirus Method used	Avast	Avria	Norton	Kaspersky	NOD32
1. Process termination	V	V	V	X	X
2. Mouse simulator	V	V	V	V	V
3. Registry modification	V	V	V	V	V
4. Close Message	V	X	X	X	V
5. Null debugger	V	V	X	V	V
6. Dll unloading	X	V	V	V	X
7. Thread termination	X	X	X	X	X



(a)

(b)

Fig. 2. (a) The vanilla system service procedure (b) The system service procedure with the patch of antivirus for malware detection

The service dispatcher can be referred to as the interface to various system functionalities and is rarely changed for compatibility in different versions of an OS. Considering the applicability, AVs also choose the service dispatcher as the injection points for their hooks. Taking advantage of this convention, malware authors equip their malware with the ability to compare the system service dispatcher with the vanilla one to diagnose the possible presence of an AV. Moreover, by replacing the dispatcher with the vanilla one, the hooks of the AV can be un-installed, leading to the circumvention of the security validation once and for all.

In addition to the hook-sabotaging techniques introduced, other approaches disabling AVs also exist. Table I gives the results when applying 7 common AV terminators against five well-known AVs [6]. The result indicates that even without the in-depth knowledge regarding the system service dispatcher to perform the un-hook patch, it is still possible to disable the AV in the straightforward way.

To prevent AVs from being terminated by anti-AV malware, Hsu et al. proposed the ANSS (ANtivirus Software Shield) [6]. The basic idea is to complement the checkpoints for the security of AVs. In general, the coverage of the checkpoints of AVs is limited to the detection to malicious files being accessed. For malware that survives the detection, the AV is completely exposed without additional protection. In addition to the original checkpoints, ANSS further inserts checkpoints into specific system service procedures, which may be utilized to terminate the AV. The security of AV itself can be therefore enhanced.

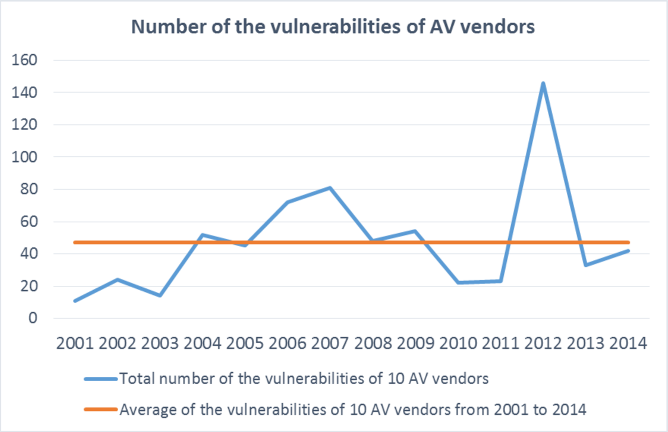


Fig. 3. Number of AVs' vulneraibilties from 2001 to 2014

B. Attacking AVs' Vulnerabilities

To deal with increasing security threat, the functionalities of AVs are further enhanced. In addition to the fundamental malicious file scanning, other features such as spam email and web content filtering are also provided by modern AVs. Along with the growth of implementation complexity of AVs, the security vulnerabilities due to programming bugs are inevitably encountered.

Compared to most malware passively waiting to be triggered by users, the behaviors of AVs are much easier to predict due to the fact that AVs scan every suspicious files. As a metaphor, AV is like a fish, trying to eat the bet. The predictable behavior allows adversaries to focus only on the construction of attacks to vulnerabilities without the consideration of the triggering conditions. AVs' vulnerabilities eventually will be discovered and becomes another threat to users.

According to the top 50 software vendors with the highest number of vulnerabilities given by Common Vulnerabilities and Exposures (CVE) [4], which is a database of vulnerabilities, two AV vendors are included along with other software vendors such as Microsoft, VMware, and Apache. The fact that the two AV vendors are on top of the list implies that AVs are amongst the most vulnerable software systems.

Figure 3 depicted the total number of vulnerabilities each year found in AVs developed by ten well-known AV vendors from 2001 to 2014. On average, 47 vulnerabilities are found annually in the last ten years, shown in Fig. 3. The evidence indicates that the security threats caused by vulnerable AVs deserve more attention.

To secure security tools including the AVs from being compromised, a promising trend is to isolate the tools from the environment where the subject program, which could be a malware, is being executed. The virtual machine technology has the best chance to realize this scenario by dividing the system into the guest OS and the host OS. A security tool operating in the host OS can be cloaked by the boundary between the guest and the host. However, this VM-based implementation encounters great difficulty in addressing the semantic-gap issue. That is, without the assistance of the guest OS, the host-side security tools are required to interpret the raw bytes in the memory of the guest OS into human-readable information before any analysis can be proceeded. In spite of various efforts, may challenges remain unsolved. The operations and functionalities of the security tools hosted in the host OS are still limited. Further enhancement of the VM-based approaches is desirable due to its powerful isolation.

IV. Scattered Malware Information

For a long time, AV vendors work in the standalone paradigm. Each vendor has its own definition to tag an analyzed malware [7]. This information is hardly synced among different AV vendors. Consequently a malware sample detected by an AV can still escape from another. The cooperation of the AV vendors is expected to construct a more comprehensive defense against numerous malware.

VirusTotal [3], which is the web service organizing the malware analysis result of various AVs, shares the collected samples with their cooperated AV vendors. Through the information sharing, the time windows from malware appearance to signature generation can be shortened. This provides better protection to the end users. Similarly, further collaboration of AV vendors such as sharing malware signatures and naming rules can reduce the response time to a new threat.

V. Conclusion

In this article, three issues including security threats, detection strategy, and collaboration of anti-virus software are discussed. The heuristic approaches such as dynamic behavior analysis can be applied along with the signature-based detection for malware detection for both efficiency and accuracy. By setting up additional checkpoints at the early stage of the installation of a clean OS, the anti-AV attack can be effectively eliminated. Moreover, through the virtual machine technology, the AVs have the opportunity to serve its security purposes while being cloaked by the isolation features provided. While the large amount of malware exhausting individual AV vendors, cross-vendor collaboration such as sharing of malware information among different AVs will greatly raise the barrier for malware to circumvent.

Acknowledgments

This work is supported in part by the Ministry of Science of Technology of Taiwan, Taiwan Information Security Center, Industrial Technology Research Institute of Taiwan, Institute for Information Industry of Taiwan, the International Collaboration for Advancing Security Technology, HTC Corporation, D-Link, Trend Micro, Promise Inc., Chungshan Institute of Science and Technology, Bureau of Investigation, and Chunghwa Telecomm.

References

[1] Ghost-Hunting With Anti-Virus , http://www.fireeye.com/blog/corporate/2014/05/ghost-hunting-with-anti-virus.html

[2] Antivirus Isn’t Dead, It Just Can’t Keep Up http://labs.lastline.com/lastline-labs-av-isnt-dead-it-just-cant-keep-up.

[3] VirusTotal, https://www.virustotal.com/en/about/

[4] Common Vulnerabilities and Exposures List , https://cve.mitre.org/

[5] P. O’Kane, S. Sezer, and K. McLaughlin. Obfuscation: Thehidden malware. Security & Privacy, IEEE, 9(5):41–47, 2011.

[6] Fu-Hau Hsu, Min-Hao Wu, Chang-Kuo Tso, and Chieh-Wen Chen. “Antivirus Software Shield Against Antivirus Terminators”, IEEE transaction on information forensics and security, vol. 7, no. 5, October , 2012.

[7] A. Mohaisen and O. Alrawi. “Av-meter: An evaluation of antivirus scans and labels.” In DIMVA, 2014.

__Chia-Wei Wang__ is a PhD student in the Laboratory for Distributed System and Network Security at the National Chiao-Tung University, Taiwan. His research interests include the virtual machine security and the malware research, Win32 especially. Chiawei has a bachelor degree in computer science from National Sun-Yat Sen University, Taiwan. Contact him at cwwang.cs98g@g2.nctu.edu.tw

__Chong-Kuan Chen__ is a PhD student in the Department of Computer Science at National Chiao Tung University, Taiwan. His research interests include network security, system security, and malware analysis. Contact him at ckchen@cs.nctu.edu.tw.

__Shiuhpyng Winston Shieh__ is a distinguished professor and the past Chair of the Department of Computer Science, National Chiao Tung University (NCTU), and the Director of Taiwan Information Security Center at NCTU. His research interests include reliability and security hybrid mechanisms, network and system security, and malware behavior analysis. He is actively involved in IEEE and has served as the Reliability Society (RS) VP Tech, and Chair of RS Taipei/Tainan Chapter. Shieh received his PhD in electrical and computer engineering from the University of Maryland, College Park. He (along with Virgil Gligor of CMU) invented the first US patent in the intrusion detection field. He is an IEEE Fellow and ACM Distinguished Scientist. Contact him at ssp@cs.nctu.edu.tw.