Vol.2012-OS-120 No.6
2012/2/29

An Analysis of Fake Antivirus Behaviors

Masaki Kasuya[†1] and Kenji Kono[†1,†2]

Fake antivirus (AV) software aims to scam web users, scaring them by showing fake alerts, as if their computers were infected by malware, urging them to purchase commercial versions of the antivirus. Deceived users disclose credit card numbers and other sensitive information. To defend against fake AV, security vendors and researchers provide or develop the countermeasures based on signatures and blacklists of the URLs distributing fake AV. However, these traditional solutions do not fit the current situation. Fake AV is rapidly increasing in number and changing the domain names frequently that are used for fake AV distribution. In this paper, we investigate the scanning behaviors of fake AV and search for an indicator that distinguishes fake AV from genuine AV. Using this indicator, fake AV is expected to be detected without signatures or blacklists. To this end, we collected 38 fake AV samples and 8 genuine AV products and gathered the data of file access tendency, CPU and memory usage. As a result, we found that memory usage indicates the difference between fake AV and genuine AV.

1. Introduction

Fake antivirus (AV) software is rapidly becoming one of the major threats to the security of Internet users. Rajab et al.[1)] and McAfee[2)] have conducted surveys on this issue. The former shows fake AV accounts for 15% of all malware detected by Google's malware detection infrastructure[3)] and the latter shows 23% of malicious web links was fake alert web sites.

The ultimate goal of fake AV is to swindle novice computer users and to acquire their sensitive information such as credit card numbers and contact information. It shows fraudulent alerts pretend to be legitimate security software, but the computer has not been actually infected with malware. To make matters worse, it recommends purchase of a commercial version of AV to novice users. As a consequence, they end up purchasing the useless products because they believe their computers are infected with malware. Recently, rogue defragmentation tools[4)] and fake AVs for mobile device such as Android[5)] are also appearing.

Profits for the scam bring in multi-million dollars to the underground economy. In fact, `Doctor Virus`, an instance of the fake AV, has generated around $9.2 million dollars from fiscal 2005 to 2007[6)]. McAfee's survey discloses that annual revenues of fake AV approached $180 million dollars[2)]. This is because fake AV software has become one of the most lucrative criminal operations on the Internet. In these circumstances, the Department of Justice and the FBI announced Operation Trident Tribunal to disrupt cyber crimes[7)].

To defeat this, security vendors and researchers provide countermeasures such as signatures for virus scanning and blacklists of fake AV distribution servers. However, these only provide limited abilities. The variants of fake AV can be easily and quickly created by polymorphic obfuscation techniques[8)–11)], and new signatures must be developed to detect new variants of fake AV. Blacklists are also useless because a large number of domains for fake AV are rotated among short-lived domains.

Due to the above situation, an alternative indicator is required as a counter plan. Intuitively, there will be a difference between fake AV and commercial AV (genuine AV). If the difference becomes the indicator to distinguish fake AV from genuine AV, some advantages might be able to solve the current issue. For example, even if a novice user installs a fake AV sample, a detection tool can detect it with the indicators. Also, if a fake AV is instantly detected by investigating the behavior, when a security vendor develops in-house system such as honeypots[12)] and aggressive honeyclient system[13)] to collect malware and uses automate clustering systems[14)], it will be able to save them trivial analysis and concentrate on the fake AV they have to analyze.

To this end, we investigate the difference of virus scanning behavior between genuine AV products such as Kaspersky and fake AV samples. Intuitively, fake AV shows the similar behavior, none the less there is malware or not. On the other hand, genuine AV reveals the behavior with a significant difference when it scans malware on a target environment. In particular, we select file access tendency, CPU and memory usage as a candidate of an indicator to distinguish

†1 Department of Information and Computer Science, Keio University
†2 CREST, Japan Science and Technology Agency

fake AV from genuine AV. From our investigations, we found memory usage is a good indicator for recognizing fake AV.

To look into whether there is statistical significance or not about the difference of memory usage, we use Levene's Test[15] for 38 fake AV samples and 8 genuine AV products. Following our instincts, 35 fake AVs exposed almost the same memory usage distribution regardless there is malware or not, and all genuine AV products showed the significant memory usage on scanning for malware. However, 3 fake AVs show genuine AV-like behavior: these fake AVs use memory when malware is putted in a target environment. To investigate whether the reason is due to malware or not, we collected both scanning behaviors about the usage with innocent data which are picture files and without them, so that the three fake AVs disclosed significant differences about memory usage in a case of this, of course, any genuine AV does not show that. Therefore, we found the difference of memory usage is enabled to use as a indicator to distinguish fake AV from genuine AV.

The remainder of this paper is organized as follows. Section 2 shows detailed fake AV threats. Section 3 explains our intuitive idea to identify fake AV. In section 4, we reveal which behavior is different between commercial AV and fake AV. In section 5, we provide an evaluation of the effectiveness of our intuitive idea. Discussion about our idea and related work in this field presented in section 6 and 7. Finally, section 8 concludes our work.

2. Threat of fake AV

In this section, we present the threat of fake AV as a scam by criminals. We will also discuss the limitations of current countermeasures.

2.1 Profit tool for criminal organization

The cost to hapless purchasers of fake AV is substantial. Cova et al.[16] and Stone-Gross et al.[17] estimate the loss for fake AV, which are made by unwitting people thinking that they are buying a genuine AV product. Stone-Gross et al.[17] have acquired backend servers for real criminal operations of three fake AVs and their investigation reveals the largest revenue was $48.4 million dollars per year. The sum of three revenues was more than $130 million dollars. According to Symantec survey[18], the cost of one fake AV ranges from $30 to $100 dollars.

Fake AV infiltrates victim systems via botnets such as Koobface, Conficker and Bredolab[19)–21]. In addition to this, fake AV distributors often use blackhat search engine optimization and drive-by-download attacks[22)–24].

2.2 Current defensive situation against fake AV

Traditional countermeasures are useless methods for immediate detection and prevention of fake AV. Signature-based tools are useless against the most up-to-date fake AVs and their variants. To keep up with these, we have the need to continuously update latest version of commercial AV, which are manually generated by analyzing existing fake AVs. But signatures can be easily evaded by simple obfuscation techniques[8)–11]. In fact, Rajab et al.[1] show how out-of-date signatures are useless once new one or variants appear. To prevent us from visiting servers distributing fake AV, blacklists are commonly used as a well-known method, which are IP-based, domain-based and DNS-based approaches. The IP-based blacklisting approach introduces many false positives because an IP address includes legitimate and malicious websites. Since the distributor continuously rotates among short-lived domains, domain-based blacklisting is also ineffective. Although Cova et al.[16] mention a DNS-based solution will be a good method, they only say that it seems to be a promising measure.

Someone might say that "fake AV is a complete fabrication, so it can be identified by closely observing the behavior". Although we can watch GUI and surface behavior of fake AV which is different from the stealthy malware such as spyware and rootkit, these naive approaches will be in vain. In practice, the look and feel of recent fake AVs resemble those of genuine products. To make matters worse, some fake AVs display real threat on infected computer such as `AntiVirusElite`[25]. Thus the loose approach[26] might become a complicated method for novice users.

3. Intuitive Idea

In this section, we explain some ideas to seek a new indicator for fake AV. We focus on three scanning behaviors of genuine AV and fake AV: file access tendency, CPU and memory usage. Intuitively, genuine AV will express characteristic behavior different from fake AV when it is scanning malware.

3.1 File access tendency

To inspect whether computer is infected by malware, genuine AV scans a large number of files on the computer. Therefore it accesses each file to check whether the file is infected with malware or not. On the other hand, fake AV will access directory files to get existing directory paths and file names on the computer for scam, but it is not going to inspect each file in detail. Obtaining the difference of this tendency, we can identify whether an unidentified AV is fake or not.

3.2 CPU usage

The difference of CPU usage between kernel and user time might be influenced by scanning behavior of unknown AV. Since the detailed scanning for malware would be executed on the user space in genuine AV, CPU usage will be occupied in the user time. But fake AV does not need to execute detailed scanning. Fake AV will spend most of the CPU time in the kernel because it involves a lot of disk access.

3.3 Memory usage

Scanning to detect malware will lead to significant increase in memory usage, which is true for genuine AV. But memory usage of fake AV will generate almost the same distribution, regardless of the presence of malware. Hence, the difference of memory usage might become an indicator of fake AV.

4. Investigation

In order to verify the correctness of our intuitive ideas presented in the previous section, we collect 38 fake AVs and 8 genuine AVs which are listed in **Table 1** and **Table 2**, and acquire file access tendencies using a system call hooking module, CPU and memory usage by Windows Performance Monitor per one second. Each data is collected in Windows XP SP3 on VMware Fusion 3.1.3. Memory uses 2GB and CPU is Intel Xeon 2.4 GHz. We use the latest signatures for each genuine AV. The names of the genuine AVs have been anonymised to discourage comparisons about this results.

4.1 Difference in file access tendency

Unlike our intuition, fake AV accessed not only directory files but also each file. Therefore, it is difficult to distinguish fake AV from genuine AV. In fact, 20 fake AV retrieved each file as well as all genuine AV.

Table 1 Names of collected fake AVs

XP InternetSecurity 2011	PC PrivacyCleaner	VirusRemover 2008
XP InternetSecurity 2012	AntispySafeguard	VirusRemover 2009
XP HomeSecurity 2011	SecurityAntivirus	PatchupPlus
XP HomeSecurity 2012	MajorDefencekit	PestDetector
XP AntiSpyware 2011	SystemSecurity	ProtectCode
XP AntiSpyware 2012	AntispywareBot	XL Guarder
XP Antivirus 2011	PeakProtection	AdwareBot
XP Antivirus 2012	PrivacyControl	Anti-Spyware
XP Security 2011	SecurityShield	RedCross
XP Security 2012	RegistrySmart	RegClean
XP TotalSecurity 2011	ErrorSweeper	Netcom3
MalwareRemovalBot	AntiVirusElite	Onescan
AntiSpywareExpert	Security Tool	

Table 2 Names of used genuine AVs

Avast	AVG	G Data	Kaspersky	McAfee	NOD32	Norton	Panda

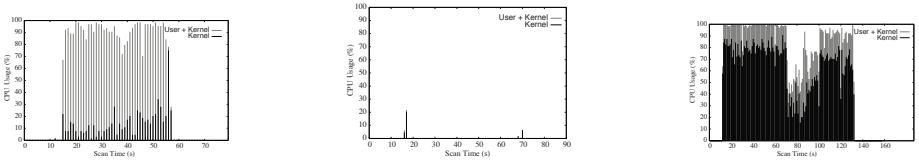
Interestingly, some genuine AVs accessed `Zone.Identifier`[⋆1] attached to malware files which are NTFS alternative data-streams. However, all the genuine AVs did not show this behavior.

4.2 Difference in CPU usage

It is very difficult to use the difference in CPU usage between user time and kernel time as an identifier for fake AV behavior detection. **Fig. 1** and **Fig. 2** show six samples about CPU usage of fake AV and genuine AV, which is a result on the target environment with 500 MB malware. The gray bar indicates the sum of usage of kernel time and user time, the black bar represents only that of kernel time.

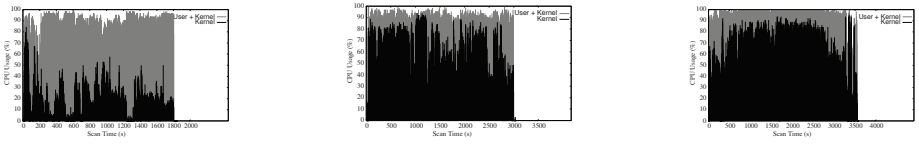
Some fake AVs show intuitive behavior such as in **Fig. 1(c)**. In other words the kernel time dominates the CPU usage. But the others consume the user time for the scanning such as **Fig. 1(a)**. Also, some genuine AVs behavior do not confirm our intuition. Although **Fig. 2(a)** consumes user time for scanning, **Fig. 2(b)** and **Fig. 2(c)** used kernel time to scan a target environment. We can conclude

⋆1 When a file on Internet is downloaded using Internet Explorer, the browser automatically attaches the ID for security alert on Windows.



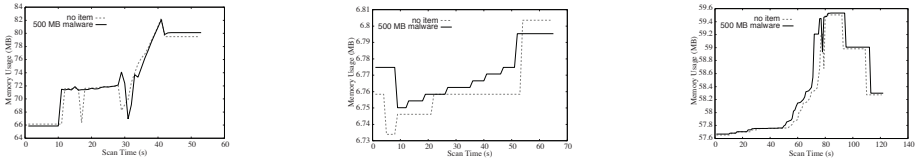
(a) AdwareBot (b) SecurityShield (c) AntiVirusElite

Fig. 1 Fake AV's CPU usage



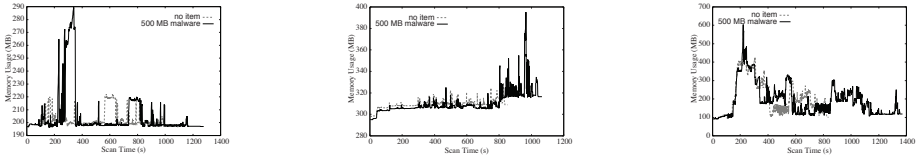
(a) Vendor A (b) Vendor B (c) Vendor C

Fig. 2 Genuine AV's CPU usage



(a) AdwareBot (b) SecurityShield (c) AntiVirusElite

Fig. 3 Fake AV's Memory usage



(a) Vendor A (b) Vendor B (c) Vendor C

Fig. 4 Genuine AV's Memory usage

from this result that we can hardly use CPU time as indicators.

4.3 Difference in memory usage

According to the result, the gap of memory usage between genuine AV and fake AV is clearly different. **Fig. 3** and **Fig. 4** shows memory usage of fake AV and genuine AV. The break line means memory usage on a target environment without malware, and the plain line shows the same with 500 MB malware.

Fake AVs such as **Fig. 3(a)**, **Fig. 3(b)** and **Fig. 3(c)** show almost the same usage distribution, regardless of the presence of malware in the victim's environment. On the other hand, genuine AVs obviously show the difference in memory usage as shown in **Fig. 4(a)**, **Fig. 4(b)** and **Fig. 4(c)**. Especially, when the genuine AVs find malware in the target environment, the usage significantly increases. Although the maximum value of Vendor A's memory usage is about 220 MB without malware, the opposite case uses 290 MB approximately. Also Vendor B and Vendor C disclose the same behavior. Therefore, the difference in memory usage might become an indicator to distinguish whether it is a fake AV or not.

5. Evaluation

In this section, we use a statistical test to show that the intuitive difference is certainly meaningful. As discussed in the previous sections, fake AV's memory usage is outrightly different from that of a genuine AV when both execute malware scanning. In comparison with fake AV, genuine AV markedly uses memory during scanning malware. Therefore the variance of memory usage with malware will be significantly different from that without malware in the case of genuine AV, whereas it will be almost equal in case of fake AV. Therefore, we use Levene's Test[15] and investigate the equality of variances of memory usage distribution between fake AV and genuine AV.

To this end, we gathered memory usage three times putting 500 MB malware files gathered from malware collection sites[27)–29)] on a target environment and no malware for each fake AV and genuine AV and executed Levene's Test 9 times between memory usage with 500 MB malware and that without malware. If any tests revealed statistical significance, we decide the sample is a genuine AV.

Table 3 The number of statistical significance about genuine AV

Name	with malware	with pictures
Avast	all	none
AVG	all	fraction
McAfee	all	fraction
NOD32	all	fraction
G Data	all	fraction
Norton	all	fraction
Kaspersky	all	fraction
Panda	all	fraction

In case we distinguished the sample as a fake AV, the results should show no statistical significance in each test ideally. But it is difficult. Although many fake AV samples show almost the same memory usage distribution, they show a slightly different distribution. However, genuine AV should always disclose the difference of that in case of scanning malware against no malware. That is the reason why we investigated whether all tests revealed statistical significance. To show that scanning malware causes the increase of malware, we also obtained memory usage in the same way, when fake AV and genuine AV scanned 500 MB pictures, and executed Levene's Test similar to above.

5.1 Levene Test

Levene's Test is described as formula 1.

$$W = \frac{(N-k)}{(k-1)} \frac{\sum_{i=1}^{k} N_i \left(Z_{i.} - Z_{..}\right)^2}{\sum_{i=1}^{k} \sum_{j=1}^{N_i} \left(Z_{ij} - Z_{i.}\right)^2} \tag{1}$$

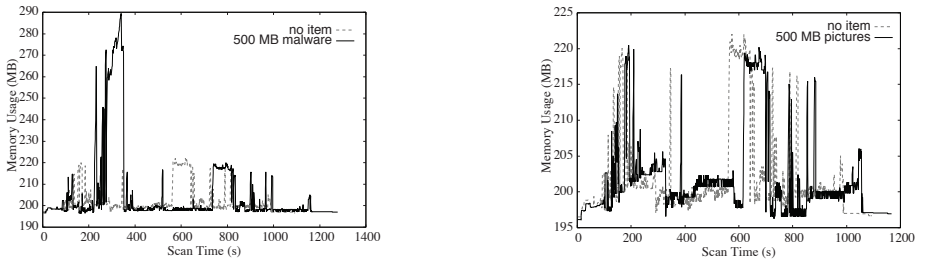
W represents the result of this. N means the number of samples. k expresses the number of different groups, which is 2 in this case. N_i is the number of samples in the i th group. Z_{ij} follows formula 2.

$$Z_{ij} = \left| Y_{ij} - \bar{Y}_i \right| \tag{2}$$

Y_{ij} exhibits the value of the j th sample from the i th group. $\bar{Y}_{i.}$ is a mean of i th group. $Z_{i.}$ and Z_{ij} are the mean of the Z_{ij} for group i and the mean of all Z_{ij}. Significance level is 5%. If W is less than the significance level, the test decide the difference is statistical significance.

5.2 Result of genuine AV

Table 3 is the result of Levene's Test for genuine AV. In this table, *all* means 9 times tests express significant difference, *fraction* means some tests represents significant difference and *none* means no tests show significant difference.



(a) with malware (b) with picture

Fig. 5 Vendor A's memory usage with 500 MB malware or 500 MB pictures against no item

Disclosing the statistical significance about all cases of the test between 500 MB malware and no malware, it hardly show statistical significance in many cases between 500 MB pictures and no pictures which is same environment of no malware. For example, **Fig. 5** shows Vendor A's memory usage. Certainly, scanning with picture is almost same distribution about no item environment such as **Fig. 5(b)** although the memory usage during scanning malware is different from no items environment such as **Fig. 5(a)**. That is the reason why, this result provides statistically that genuine AV's memory usage follows our intuition.

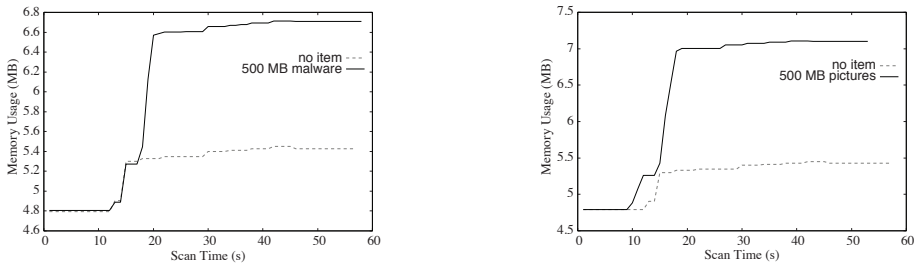
5.3 Result of fake AV

Table 4 is the result of Levene's Test for fake AV. Following our intuition, Levene's Test determines 35 fake AV samples show *fraction* or *none* about the difference of memory usage between 500 MB malware and no additional item. However, 3 fake AVs, `Privacy Control`, `XL Guarder`, `Protect Code`, disclose counterintuitive results. In other words, the 3 samples have statistical significance in any case between 500 MB malware and no additional item as well as genuine AV. **Fig. 6** is the memory usage of `Privacy Control` which is one of the 3 samples. Certainly, **Fig. 6(a)** discloses the difference of the usage between 500 MB malware and no item. But the increasing tendency is different from that of genuine AV, which is genuine AV's memory usage rapidly increase and decrease when it scans malware, but that of fake AV only rapidly increases.

Table 4 The number of statistical significance about fake AV

Name	with malware	with pictures
AdwareBot	none	none
AntiSpywareSafeguard	none	none
Anti-Spyware	none	none
AntiSpywareBot	fraction	fraction
AntiVirusElite	fraction	fraction
AntiSpywareEexpert	fraction	fraction
ErrorSweeper	fraction	fraction
MajorDefenceKit	fraction	fraction
MalwareRemovalBot	none	none
Netcom3	none	fraction
Onescan	fraction	none
PatchupPlus	none	none
PC PrivacyCleaner	fraction	fraction
PeakProtection	fraction	fraction
PestDetector	fraction	none
PrivacyControl	all	all
RedCross	none	none
RegClean	none	none
RegistrySmart	none	none
SecurityAntivirus	none	fraction
ProtectCode	all	all
SecurityShield	fraction	fraction
SecurityTool	fraction	fraction
SystemSecurity	fraction	fraction
VirusRemover 2008	none	none
VirusRemover 2009	none	none
XL Guarder	all	all
XP AntiSpyware 2011	fraction	fraction
XP AntiSpyware 2012	fraction	fraction
XP AntiVirus 2011	fraction	fraction
XP AntiVirus 2012	fraction	fraction
XP HomeSecurity 2011	fraction	fraction
XP HomeSecurity 2012	fraction	fraction
XP InternetSecurity 2011	fraction	fraction
XP InternetSecurity 2012	fraction	none
XP Security 2011	fraction	fraction
XP Security 2012	fraction	fraction
XP TotalSecurity 2011	none	none

Next interest is whether the difference indicates malware. To end this, we investigate the difference of fake AV's memory usage between 500 MB pictures and no item. Interestingly, all cases of Levene's Test about the differences of the 3 fake AV's memory usage are also statistically significant as well as in case that compared with 500 MB malware. Of course, the others do not disclose *all* the statistical significance. In fact, the memory usage of `Privacy Control` was



(a) with malware (b) with picture

Fig. 6 Privacy Control's memory usage with 500 MB malware or 500 MB pictures against no item

certainly different such as **Fig. 6(b)**. Therefore, the 3 fake AV's memory usages are influenced by file size only. As a result, the memory usage of fake AV is different from that of genuine AV. Therefore, the difference of memory usage on scanning is an indicator to distinguish fake AV from genuine AV.

6. Discussion

So far, we have demonstrated that memory usage becomes a good indicator to distinguish fake AV from genuine AV. However, adversaries might be able to evade our intuitive idea using some methods. In this section, we discuss counter approaches of adversaries and our idea's limitations.

First, some fake AVs might aggressively use memory such as genuine AV when it finds malware on environments of victims. But, it is not realistic. If fake AV reflects genuine AV's memory usage, fake AV has to prepare substantial malware lists such as signatures. Of course, genuine AV vendors continuously generate signatures against numerous up-to-date malware and variants, which is a boring task because security vendors mostly have to deal with tens of thousands of new malware samples day after day[30]. Therefore, it is too costly to have things such as signatures for each malware in advance.

Although we have mentioned `AntiVirusElite` detects real malware on a victim's environment in Section 2, it has not detected all malware on the environment. Furthermore, it has not disclosed statistical significance about memory

usage when we gathered the data. And since adversaries cannot know in advance what kind of malware is used to gather memory usage, they have to prepare a lot of signatures to evade our approach.

Second, an adversary might reconfigure open source anti-virus software such as ClamAV[31]. In this case, an adversary might be able to evade our approach. Therefore, it is needed to specifically analyze how these fake AVs work. However, any fake AV samples we collected are not reconfigured instances of open source AV. Since the target of fake AV is a scam against novice users, it does not satisfy the goal to modify open source AVs. In fact, any collected fake AVs took less than three minutes to scan. Even if fake AV changes an open source AV for a scam, it will be analogous with to scanning time tendency. Therefore, we should also consider scanning time to defeat this.

7. Related work

Explained in the past section, there are some studies about fake AV. Google researchers investigated dynamics of fake AV, and they revealed some interesting characteristics such as relying on advertisement, funneling user traffic by landing web pages and containing trend key words[1]. Cova et al. characterized the behaviors of the campaigns and the economics of fake AV[16]. Also, they discuss the limitations of current techniques and the next research direction for the counter approaches. Stone-Gross et al. disclose the relationship between the number of chargebacks and refunds[17]. According to them, the numbers of both show similar distributions in the case of fake AV, however the tendency is uncommon in regular chargeback processes. Therefore, they mention the similar distribution can be an indicator to find fake AV distributors for a credit card company. In this paper, we uncovered essential difference of memory usage between fake AV and genuine AV, which can expect to be a new indicator of a countermeasure of fake AV for users and security vendors.

As another scamming method, there is phishing. Although several server-side and client-side approaches have been proposed to defeat phishing[32)–36)], Yue and Wang[37] mentioned these approaches are insufficient to prevent novice users from being deceived. Therefore, they developed BogusBiter. The concept is *the best place to hide a leaf is in a forest*. In other words, it hides real sensitive information such as login name and password by simultaneously sending a lot of fake information to phishing sites. However, the method does not fit into fake AV prevention. Because fake AV usually uses credit card numbers as sensitive information for purchase in case of fake AV, it will cause invalid transaction to input the wrong numbers.

For countermeasure of common malware, there is taint analysis as a well-known technique, which tracks the focused information flow. Taint analysis is useful for several kinds of counter approaches. In fact, many researchers proposed malware detection systems[38),39)], malware analysis systems[40),41)] and intrusion detection systems[42),43)]. Although our technique use information of memory usage about fake AV and genuine AV, it does not use data flow information.

8. Conclusion

Fake AV is becoming one of the major security threats. The idea is very lucid; it shows phony security alerts and swindles novice users to obtain sensitive information such as credit card details. The countermeasure using traditional tools cannot immediately follow the latest AVs and polymorphic variants. Therefore an alternative approach to fill this gap is desired.

In this paper, we investigated some behaviors of genuine AV and fake AV: file access tendency, CPU usage and memory usage. Intuitively, genuine AV aggressively consumes computer resources when it scans malware, however fake AV hardly uses that. In our research, memory usage is suited as a indicator to build behavior detection tools for fake AV, although file access tendency and CPU usage does not fit for the indicator.

To support our intuitive idea, we collected 38 fake AV samples and 8 genuine AV products, and gathered these memory usage in case of putting on malware or not on the target environment, so that genuine AV revealed statistical significance about the usage on malware scanning using Levene's Test, but some fake AVs showed almost same distribution about the memory usage, regardless there is malware or not. However 3 fake AVs was different from our intuition. To look into whether malware causes the behavior against our intuition, we obtained memory usage when fake AV and genuine AV scanned 500 MB pictures and took Levene's Test. As a result, The reason for increasing of memory usage about the

3 fake AVs was file size. Therefore, we think memory usage can be a indicator to distinguish fake AV from genuine AV. This indicator will help us to fill gap of current limitations for now fake AV solutions.

Future work will be to necessary develop an automatically detection tool based on memory usage distribution.

References

1) Rajab, M.A., Ballard, L., Mavrommatis, P., Provos, N. and Zhao, X.: The Nocebo Effect on the Web: An Analysis of Fake Anti-Virus Distribution, *Proceedings of the 3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '10)* (2010).
2) Paget, F.: Running Scared: Fake Security Software Rakes in Money Around the World, http://www.mcafee.com/us/resources/white-papers/wp-running-scared-fake-security-software.pdf (2010).
3) Provos, N., Mavrommatis, P., Rajab, M.A. and Monrose, F.: All Your iFRAMEs Point to Us, *Proceedings of the 17th USENIX Security Symposium*, pp.1–16 (2008).
4) Ward, E.: Trojan Feigns Failures to Increase Rogue Defragger Sales, http://www.symantec.com/connect/blogs/trojan-feigns-failures-increase-rogue-defragger-sales (2011).
5) Maslennikov, D.: Android malware: new traps for users, http://www.securelist.com/en/blog/208193306/Android_malware_new_traps_for_users (2011).
6) SPAMfighter News: South Korea - Former Antivirus Chief Charged for Cheating, http://www.spamfighter.com/News-9994-South-Korea-Former-Antivirus-Chief-Charged-for-Cheating.htm (2008).
7) FBI: 'Scareware' Distributors Targeted, http://www.fbi.gov/news/stories/2011/june/cyber_062211/cyber_062211 (2011).
8) Christodorescu, M. and Jha, S.: Testing Malware Detectors, *Proceedings of the ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA '04)*, pp.34–44 (2004).
9) Yetiser, T.: Polymorphic Viruses - Implementation, Detection, and Protection, http://vx.netlux.org/lib/ayt01.html (1993).
10) Linn, C. and Debray, S.: Obfuscation of Executable Code to Improve Resistance to Static Disassembly, *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, pp.290–299 (2003).
11) Moser, A., Kruegel, C. and Kirda, E.: Limits of Static Analysis for Malware Detection, *Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC '07)*, pp.421–430 (2007).
12) Spitzner, L.: *Honeypots: Tracking Hackers*, Addison-Wesley Longman Publishing Co., Inc. (2002).
13) Wang, Y.-M., Beck, D., Jiang, X., Roussev, R., Verbowski, C., Chen, S. and King, S.: Automated Web Patrol With Strider HoneyMonkeys: Finding Web Sites That Exploit Browser Vulnerabilities, *Proceedings of the 13th Annual Network and Distributed System Security Symposium (NDSS '06)* (2006).
14) Bayer, U., Comparetti, P.M., Hlauschek, C., Kruegel, C. and Kirda, E.: Scalable, Behavior-Based Malware Clustering, *Proceedings of the 16th Annual Network and Distributed System Security Symposium (NDSS '09)* (2009).
15) Levene, H.: *Robust Tests for Equality of Variances*, Stanford University Press (1960).
16) Cova, M., Leita, C., Thonnard, O., Keromytis, A.D. and Dacier, M.: An Analysis of Rogue AV Campaigns, *Proceedings of the 13th International Symposium on Recent Advances in Intrusion Detection (RAID '10)*, pp.442–463 (2010).
17) Stone-Gross, B., Abman, R., Kemmerer, R.A., Kruegel, C., Steigerwald, D.G. and Vigna, G.: The Underground Economy of Fake Antivirus Software, *Proceedings of the Tenth Workshop on Economics of Information Security (online) (WEIS '11)* (2011).
18) Symantec: Symantec Report on Rogue Security Software, http://www4.symantec.com/Vrt/wl?tu_id=TeCm125590003756772344 (2009).
19) Villeneuve, N., Deibert, R. and Rohozinski, R.: KOOBFACE: Inside a Crimeware Network, http://www.infowar-monitor.net/reports/iwm-koobface.pdf (2010).
20) Poulsen, K.: Conficker Doomsday Worm Sells Out For $49.95, http://www.wired.com/threatlevel/2009/04/conficker-dooms (2009).
21) Kirk, J.: Bredolab-infected PCs Downloading Fake Antivirus Software, http://www.pcworld.com/businesscenter/article/209031/bredolabinfected_pcs_downloading_fake_antivirus_software.html (2010).
22) Wang, D.Y., Savage, S. and Voelker, G.M.: Cloak and Dagger: Dynamics of Web Search Cloaking, *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS '11)*, pp.477–489 (2011).
23) Lu, L., Perdisci, R. and Lee, W.: SURF: Detecting and Measuring Search Poisoning, *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS '11)*, pp.467–476 (2011).
24) Doshi, N.: Misleading Applications - Show Me The Money!, http://www.symantec.com/connect/blogs/misleading-applications-show-me-money (2009).
25) Dedicated 2 Spyware: Remove Anti-Virus Elite. Removal instructions, http://www.2-spyware.com/remove-antivirus-elite.html.
26) Karnik, A., Avelino C.Rico, J., Prakash, A. and Honjo, S.: Identifying Fake Security Products, http://www.mcafee.com/us/resources/white-papers/wp-identifying-fake-security-products.pdf (2009).
27) Offensive Computing: Offensive Computing — Community Malicious code researh and analysis, http://offensivecomputing.net.
28) MDL: Malware Domain List, http://www.malwaredomainlist.com/.

29) VX Heavens: Welcome to VX Heavens! (VX Heavens), http://vx.netlux.org.
30) Bayer, U., Habibi, I., Balzarotti, D., Kirda, E. and Kruegel, C.: A View on Current Malware Behaviors, *Proceedings of the 2nd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '09)* (2009).
31) Clam AV: Clam AntiVirus, http://www.clamav.net/lang/en/.
32) Downs, J.S., Holbrook, M.B. and Cranor, L.F.: Decision Strategies and Susceptibility to Phishing, *Proceedings of the 2nd Symposium on Usable Privacy and Security (SOUP '06)*, pp.79–90 (2006).
33) Schechter, S.E., Dhamija, R., Ozment, A. and Fischer, I.: The Emperor's New Security Indicators: An evaluation of website authentication and the effect of role playing on usability studies, *Proceedings of the 28th IEEE Symposium on Security and Privacy (S&P '07)*, pp.51–65 (2007).
34) Wu, M., Miller, R.C. and Garfinkel, S.L.: Do Security Toolbars Actually Prevent Phishing Attacks?, *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pp.601–610 (2006).
35) Whalen, T. and Inkpen, K.M.: Gathering Evidence: Use of Visual Security Cues in Web Browsers, *Proceedings of the conference on Graphics Interface*, pp.137–144 (2005).
36) Dhamija, R., Tygar, J.D. and Hearst, M.: Why Phishing Works, *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pp.581–590 (2006).
37) Yue, C. and Wang, H.: Anti-Phishing in Offense and Defense, *Proceedings of the 24th Annual Computer Security Applications Conference (ACSAC '08)*, pp.345–354 (2008).
38) Egele, M., Kruegel, C., Kirda, E., Yin, H. and Song, D.: Dynamic Spyware Analysis, *Proceedings of USENIX Annual Technical Conference*, pp.233–246 (2007).
39) Yin, H., Song, D., Egele, M., Kruegel, C. and Kirda, E.: Panorama: Capturing System-wide Information Flow for Malware Detection and Analysis, *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, pp.116–127 (2007).
40) Moser, A., Kruegel, C. and Kirda, E.: Exploring Multiple Execution Paths for Malware Analysis, *Proceedings of the 28th IEEE Symposium on Security and Privacy (S&P '07)*, pp.231–245 (2007).
41) Sharif, M., Lanzi, A., Giffin, J. and Lee, W.: Automatic Reverse Engineering of Malware Emulators, *Proceedings of the 30th IEEE Symposium on Security and Privacy (S&P '09)*, pp.94–109 (2009).
42) Newsome, J. and Song, D.: Dynamic Taint Analysis for Automatic Detection, Analysis, and Signature Generation of Exploits on Commodity Software, *Proceedings of the 12th Annual Network and Distributed System Security Symposium (NDSS '05)* (2005).
43) Ho, A., Fetterman, M., Clark, C., Warfield, A. and Hand, S.: Practical Taint-Based Protection using Demand Emulation, *Proceedings of the 1st ACM European Conference on Computer Systems (EuroSys '06)*, pp.29–41 (2006).