A Framework for Optimizing Malware Classification by Using Genetic Algorithm

Mohd Najwadi Yusoff and Aman Jantan

School of Computer Science,
Universiti Sains Malaysia,
Penang, Malaysia
mohd.najwadi@gmail.com, aman@cs.usm.my

Abstract. Malware classification is a vital in combating the malware. Malware classification system is important and work together with malware identification to prepare the right and effective antidote for malware. Current techniques in malware classification do not give a good classification result when it deals with the new and unique types of malware. For this reason, we proposed the usage of Genetic Algorithm to optimize the malware classification system as well as help in malware prediction. The new malware classification system is based on malware target and its operation behavior. The result from this study will create a new framework that designed to optimize the classification of malware. This new malware classification system also has an ability to train and learn by itself, so that it can predict the current and upcoming trend of malware attack.

Keywords: Malware Classification, Genetic Algorithm, Unique Malware.

1 Introduction

Malware classification is one of the main systems in malware detection mechanism. It is used to classify the malware into its designated classes. Malware classification system that used machine learning techniques for classifying task was commercially applied in many anti-malware products such as Avira, AVG, Kaspersky, McAfee, Trend Micro, Symantec, Sophos and ESET [1]. Nowadays, malware commonly classified as Virus, Worms, Trojan Horses, Logical Bombs, Backdoor, Spyware, Exploit and Rootkit [2-3].

Malware classification system is very necessary and highly important when combating the malware [4]. In malware classification system, the main appliance or engine is named as classifier and it is used to classify the malware into the appropriate malware class. According to Filiol, current malware classes are mainly based on malware specific objective [3]. As for an example, malware in worm's classes used a network to send the copies of itself to other computer just to spread and do not attempt to alter the system. Therefore, by looking at the malware class, the right and effective antidote can be produced from malware specific objectives and this will help anti-malware products to prevent the malware from affecting the system.

In the recent advent, the widespread of malware has increased dramatically due to the fact that malware writers started to deploy an avoidance technique to avoid detection and analysis by anti-malware products [5]. By using this technique, malware writers can change the malware syntax or also known as malware signature but not its intended behavior, which has to be preserved [6]. The common avoidance technique used by malware writers is the code obfuscation. There are two types of code obfuscation which are polymorphism and metamorphism technique. Many new variants of polymorphic and metamorphic malware can easily be created by encrypting the code, flow transposition, substitution and renaming variable [7]. The other techniques to avoid detection are packing, anti-debugging and anti-virtualization.

The design of malware and its intention has become more sophisticated and significantly improved [8]. Current machine learning classifiers in malware classification system do not give a good classification result when it deals with the new and unique types of malware. It is because malware are becoming increasingly specialized and difficult to analyze [9]. New and unique types of malware are no longer can be classify easily to the current malware classes. These variants of malware have numerous attributes, combination syntax but showing the same behavior [9-10]. Thus, classification of malware has become more complicated and a new classification system is urgently needed.

In this paper, we proposed a framework that optimizes the current machine learning classifier by using Genetic Algorithm (GA). GA is a heuristic search that simulates the process of natural evolution. According to Mehdi, this heuristic algorithm is regularly used to generate useful solutions in optimization and search problems [11]. Malware that was created by an avoidance technique, providing almost the same functionality and showing the same behavior can be classify by using GA. It is because these types of malware basically worked same like crossover and permutation operation in GA [12]. As stated by Edge, GA has an ability to be a learning algorithm. As a result, it will make the new classification system become more reliable than the current classification systems [13].

The rest of the paper is organized as follows. Section 2 provided information about malware and its background. Section 3 is devoted to discuss about malware class and techniques involved. Proposed framework is presented in the section 4. Finally, Section 5 gives a few remarks as the conclusion.

2 Malware Background

2.1 Motivation

In this new millennium era, malicious code or malware has been recognized as the major threats to the computing world [10], [14]. All malware have their specific objective and target, but the main purpose is to create threats to the data network and computer operation [15]. Malware highly utilized the communication tools to spread itself. For examples, worms are being sent out through email and instant messages, Trojan horses attacked from infected web sites and viruses is downloaded from peer-to-peer connections to the user systems. According to Filiol, malware will pursue to abuse existing vulnerabilities on the systems to make their entry silent and easy [3]. In addition, malware works to remain unnoticed, either by actively hiding or simply not making its presence on a system recognized to the user.

With the development of the underground economy, malware is becoming very profitable product as the malware writer used to spam, steal information, perform web frauds, and many other criminal tasks. According to Martignoni, Malware had established during the past decade to become a major industry [9]. The new malware keep on increasing from time to time and this delinquent is seriously concerned by the security group around the world. Panda Security Lab reported that one third of existing computer malwares were created between Jan-Oct 2010 [16].

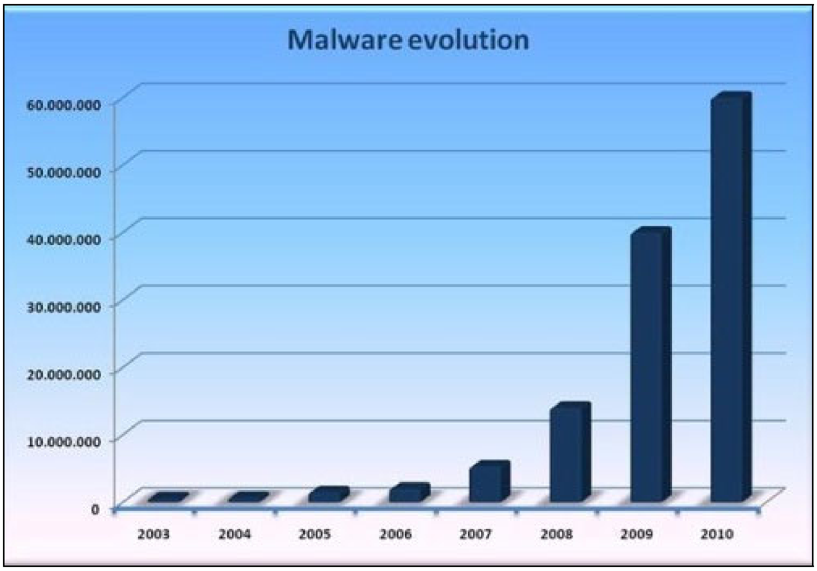

Fig. 1. Malware Evolution: Panda Lab Security [16]

The exponential growth of malware is also reported by many other security groups such as F-Secure, McAfee, Kaspersky, Sophos, Symantec and G-Data [17-22]. The increasing of malware has causes a billion of loss to the computer operation worldwide by breaking down the computer, congest the network, fully utilize the processing power, and many more bad impacts. According to ESET, more than half numbers of the malware samples nowadays are classified as unique. This unique malware are created by assimilating the existing malware with an avoidance technique. Fig. 2 is shows the total number of unique malware samples reported by ESET in 2005-2010.

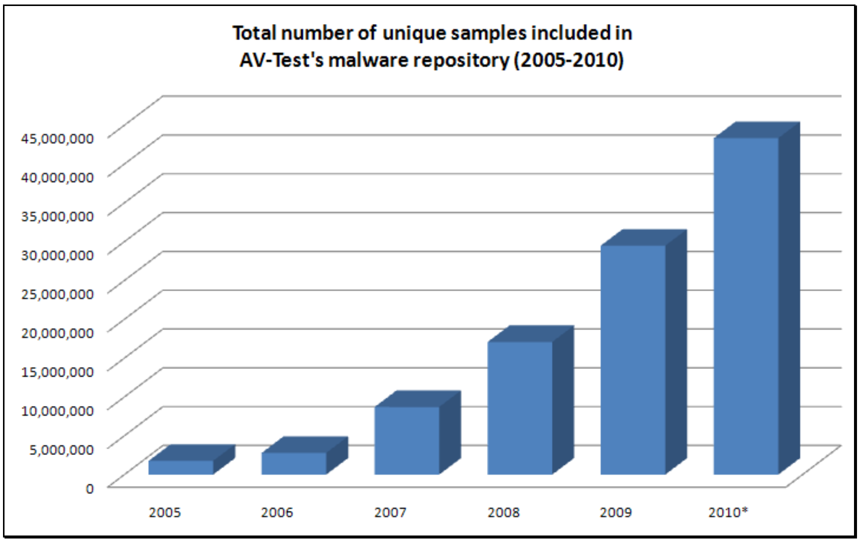

Fig. 2. Unique Malware Sample: ESET Threat Center [23]

2.2 Malware Avoidance Techniques

At the present time, nearly all modern malware has been implemented with a variety of avoidance techniques in order to avoid detection and analysis by anti-malware product [5]. The avoidance techniques that practically used by malware writers are code obfuscation, packing, anti-debugging and anti-virtualization. Code obfuscation technique can be divided into two types which are polymorphism and metamorphism. All these techniques can change the malware syntax but not its intended behavior, which has to be preserved [6]. The main purposed of these techniques is to avoid detection and to make the analysis process became more complicated [8].

A polymorphic technique can change the malware binary representation as part of the replication process. This technique consists of encrypted malicious code along with the decryption module [24]. It also has the polymorphic engine to decrypt and generate new mutants for the code before running it. When the polymorphic malware infecting the computer systems, it will encrypt itself by using new encryption key and a new code is generated. It is also has the polymorphic engine to decrypt and generate new mutants for the code before running it.

As for metamorphic technique, malware will transform the representation of programs by changing the code into different ways when it replicates but it still performs the same behaviors [25]. This technique can include control flow transposition, substitution of equivalent instructions and variable renaming [7]. Metamorphic malware can reproduced itself into different ways to rapidly created new malware variants and never look like the old malware.

Malware writers used packing technique to compress the Win32 portable execution file (PE file) and the actual file is unpacked as it is executed [26]. The main purposed of this technique is to protect the commercial software code from crack. A packed program contains with a program that is used for decompressing the compressed data during execution in the objective of making the task of static analysis become more difficult [27]. Packers will compress and sometimes it will encrypt the program. However, the program is transparently decompressed and decrypted at runtime when the program is loaded into memory. Some malware even packed its code several times in order to make it harder to be unpacked and used up so many resources until the computer hang or crash.

Debugger is a useful tool for reverse engineering of the malware code. A debugger normally can step through each instruction in a program code before it is executed. This tool performs their monitoring by either inserting breakpoints in the program or by tracing the execution using a special system calls [28]. Malware writers applied an anti-debugging technique to detect and avoid their malware from run under a debugger. Anti-debugging is an active technique where the malware writers embed code aimed to check process list for debugger process [29]. This technique will change the functionalities of the program when it interpreted by a debugger and make that program to discontinue it malicious intent or jump to end it.

The other technique is an anti-virtualization. Virtual environment is a place commonly used to do an analysis and extract the features of the malware. To avoid the malware from being analyze, malware writers used anti-virtualization to create malware code that has a capability to detect virtualization environment [30]. By using this technique, malware can check whether they are running in a virtual or normal environment before the execution. When the virtual environment is detected, malware might simply act like a genuine program or commonly refuse to run inside the virtual environment [31].

As an effect, it is not surprising that malware writer often use all these technique to automatically avoid detection. Even if anti-malware can detect it, that malware is unique and does not represent any existing malware class. Classification of malware become much harder and a new classification system is urgently needed.

3 Classification Part

3.1 Malware Class

Nowadays, malware are classified based on malware specific objective. Malware can be divided into several classes and there are diversities among the researchers in classifying the malware. In 2006, Aycock defined that malware consist of ten classes which are Logic Bomb, Trojan Horse, Back Door, Virus, Worm, Rabbit, Spyware, Adware, Hybrids and Zombies [2]. In 2009, Apel reported that malware consist of seventeen classes, seven more classes from Aycock [5]. Recent study by Filiol in 2010 stated that malware classes are divided into two groups which are “Simple” and “Self-Reproducing” [3]. However, the common class of malware nowadays is likely represented in Table 1.

Table 1. Common malware classes

Types	Specific Objective
Virus	Computer program that can copy itself and infect a computer.
Worms	Self-replicating malware which uses a computer network to send copies of itself to other nodes without any user intervention.
Trojan Horse	Software that appears to perform a desirable function for the user prior to run or install, but harms the system
Logical Bombs	Simple type of malware which wait for significant event such as date or action to be activated and launch its criminal activity
Backdoor	Method of bypassing normal authentication, securing remote access to a computer, while attempting to remain undetected.
Spyware	Malware that can be installed on computers, and collects small pieces of information about users without their knowledge
Exploit	A piece of software that attacks particular security vulnerability. Exploits are not necessarily malicious in intent
Rootkit	Software inserted onto a computer system after an attacker has gained control of the system.

We did an analysis about the malware operation behavior. For this analysis, we focus on worms and Trojan horses only. It is because these two are the common types of malware that attack host-based system [12]. We are observing the executed malware by focusing it into the specific target and its operation behavior in windows environment systems. The malware samples that we are using for this experiment comprises of 300 unique samples and the result that we get is shows in Fig. 3.

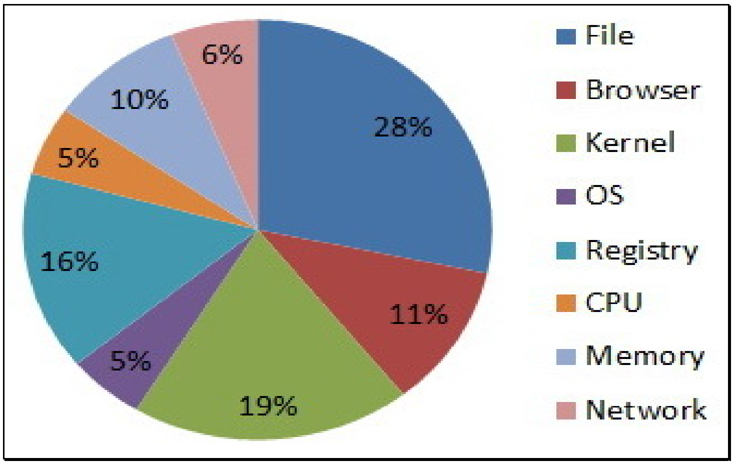

Fig. 3. Malware Specific Operation

From this result, we can classify the malware specific operation into 4 main classes which are Data, Application, System and Dos. Malware that are attack File will group under Data class and malware that are attack Browser will be under Application class. Malware that are attack Kernel, Operating System and Registry will be group under System class. Lastly, malware that are attack CPU, Memory and Network will be group under Dos class.

3.2 Machine Learning Classifier

Classifier is the main engine in the malware classification system. In this paper, we had studied several current machines learning classifier that has been used in malware classification. Table 2 shows the summary of Naïve Bayes (NB), Support Vector Machine (SVM), Decision Tree (DT) and K-Nearest Neighbor (KNN).

Table 2. Machine learning classifier in malware classification

Classifier	Speed	Accuracy	Strength	Weakness
Naïve Bayes [32-33]	Very Fast	High	Fast, easier to maintain and consistence result	Sensitive to the correlated attributes
Support Vector Machine (SVM) [4], [34]	Fast	High	Regression and density estimation results. Better performance in text classification, pattern segment and spam classification	Expensive and problem lies on the prohibitive training time.
Decision Tree [33], [35]	Very Fast	High	Easy to understand, easy to generate rules and reduce problem complexity	Mistake on higher level will cause all wrong result in sub tree
K-Nearest Neighbor [36]	Slow	Moderate	Useful when the dependent variable takes more than two values and effective if the training data is large	Very computationally intensive. O(n^2)

Based on the summary above, we decided to select DT classifier to be used and work together with the GA in the proposed framework. DT is selected because this algorithm is the most suitable algorithm for our proposed system that is based on malware specific target and its operation behavior. Malware target is referring to the Windows file system which is in the tree format. As for example, *TrojanDropper:Win32* malware has targeted two directories to execute and perform its malware operation behavior [37];

1. *C:\Documents and Settings\Users\Local Settings\Temp\*
2. *C:\Windows\System32*

3.3 Genetic Algorithm

GA is a heuristic search that simulators the process of natural evolution. The standard GA can be seen as the combination of bit-string representation, with bit-exchange crossover and bit-flip mutation, roulette-wheel selection plus generational replacement. GA is belongs to the larger class of Evolutionary Algorithm (EA). GA include the survival of the fittest idea into a search algorithm which provides a method of searching which does not need to explore every possible solution in the feasible region to obtain a good result.

GA also commonly used on a learning approach to solve computational research problem. According to Mehdi, this heuristic algorithm is regularly used to generate useful solutions in optimization and search problems [11]. In a GA, a population of strings which encode candidate solutions to an optimization problem evolves toward better answers. By tradition, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible [38].

A simple presentation of GA is shows as follows;

```
generate initial population, G(0);
evaluate G(0);
t:=0;
repeat
      t:=t+1;
      generate G(t) using G(t-1);
      evaluate G(t);
until find a solution
```

GA technique is implemented in this research in order to solve and classify the unique malware that was created by an avoidance technique. A framework is proposed by combining GA with the current machine learning classifier. New and unique malware can be detected and classify through this technique. It is because, unique malware work similar like crossover and permutation operation in GA [12]. An avoidance techniques that normally used by malware writers can be detected and classified using this new malware classification system because it not only filter the content but also train and learn by itself, so it can predict the current and upcoming trend of malware attacks.

4 Proposed Framework

We proposed the usage of the GA to enhance and optimize performance of the new classifier. GA will work together with the DT and there will be the training phase for GA that is explained in the sub section. Our proposed framework consists of several elements which are Malware Sample, Virtual Environment, Knowledge Storage, Classifier and Class Operation. Fig. 4 shows our proposed framework for optimizing malware classification using GA. The next sub-section is explained about each element in this proposed framework.

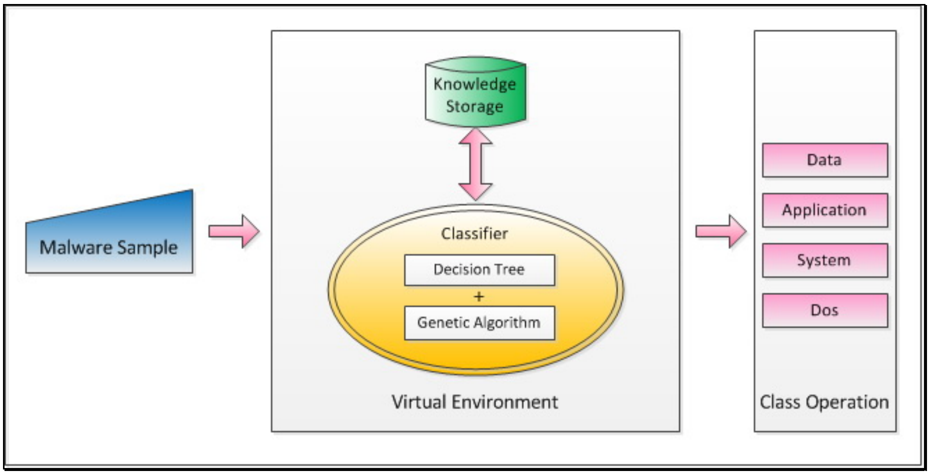

Fig. 4. A Framework for Optimizing Malware Classification Using Genetic Algorithm

4.1 Malware Sample

Most of malware are designed to attack windows-based operating system. According to Germany anti-virus firm G-Data (2010), 99.4% of the new malware are purposely target windows operating system [22]. Therefore, we are focusing our test on windows-based operating system only and narrowed down our scope and focus to Win32 portable execution file (PE file) that contains malicious code. This sample PE file is collected thru college network, internet and some suspicious execution file in windows operating system itself. Each of the file has been extracted to obtain their features and behavior. All malware targets and its operation behavior are compiled in malware profile and stored in the knowledge storage.

4.2 Virtual Environment

Virtual environment is a security mechanism for separating running programs. It is often used to execute untested code, or untrusted programs from unverified third-parties, suppliers and untrusted users. Virtual environment machine run isolated, so the benefit of a virtual environment machine is that it cannot directly modify the "real" operating system running on our machine. The virtual machine is assigned its own hard disk space, and that's what it treats as its virtual "entire hard disk".

In this proposed framework, we decided to use Windows XP Mode [39] in Windows 7 Platform as our testing platform. The purposed of this virtual environment is to create secure platform in order to minimize damage and attack to the actual machine when the malware sample is executed. Since virtual environment is assigned with its own hard disk space, if got infected, we can simply remove it from our test machine.

4.3 Knowledge Storage

Knowledge storage is the database storage where we stored all the malware profile after finish the analysis and features extracted. Malware profile consists of malware sample, MD5 hash, malware size, malware specific target and it class operation behavior. Table 3 shows an example of malware profile in our knowledge storage.

Table 3. Malware Profile in Knowledge Storage

Malware Sample	MD5 Hash	Size (byte)	Specific Target	Class Operation
umdmgr.exe	A2D09FE1EB487A799907527E494AA06B	61,440	• C:\Windows\System32	System
sdra64.exe	5A92608A111E80356F346E7405171C4A	127,488	• C:\Documents and Settings\[UserName]\Local Settings\Temp\ • C:\Windows\System32 • C:\Windows	Data
aa.exe	E9E1D7AD36C53C7B70C38DA1514BBD5D	58,749	• C:\Documents and Settings\[UserName]\Application Data • C:\Documents and Settings\All Users\Application Data • C:\Documents and Settings\All Users	Application
lsass.exe	09BA41F35106E94A9E5A3639AC52B280	22,528	• C:\Documents and Settings\[UserName]\Application Data • C:\Documents and Settings\All Users\Application Data • C:\Documents and Settings\All Users	Application

The knowledge storage is designed to be re-writable by the system. Certain unique malware has a relationship and link with the other malware when it is executed. This malware relationship is unable to analyze because during analysis process, malware is executed one by one [28-29]. At first, the information in the knowledge storage is not sufficient enough to be used in the classification system. For this reason, we used GA to conduct a training phase together with the DT. The system will update the database after gone thru the training phase.

4.4 Classifier

Classifier is the main engine in our malware classification system. We have selected the DT as our machine learning classifier and combine it with the GA. In this new system, we are having the classifier training phase and GA is used to become a learning algorithm. During the classifier training phase, we used different malware samples as the training data set. We must use different malware samples because we want to let the classifier to learn and update the new malware into malware profile. One of the main goals is to detect and classify the unique malware that has a relationship during the execution. The other goal is to find unique malware that perform the same behavior but providing different syntax representation. Fig.5 shows the classifier training phase process in our proposed malware classification system.

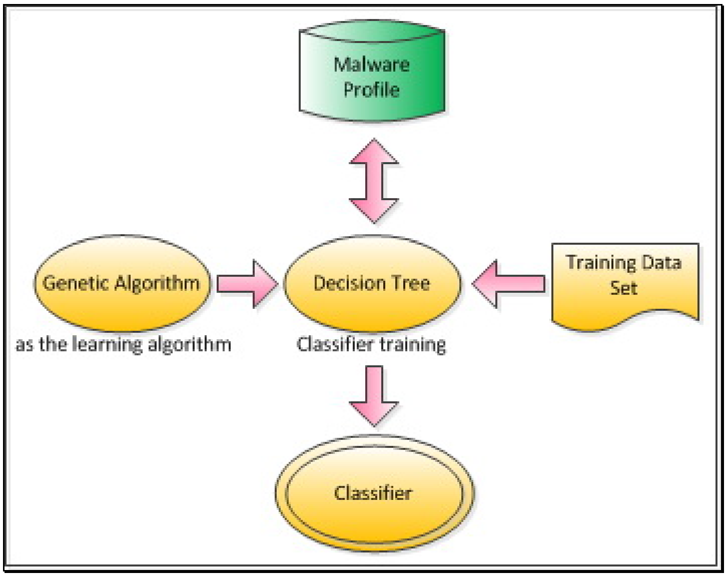

Fig. 5. Classifier Training Phase

As mention earlier, the malware profile in the knowledge storage is designed to be re-writable. Classifier will keep on updating the malware profile during this training phase. The classification result will become more accurate after this training phase. This process also shows the ability of GA in helping DT to optimize the classification process.

4.5 Class Operation

Class operation is our new proposed malware classes. Previously, we had done an analysis about the malware operation behavior using worms and Trojan horse. Based on that analysis, we proposed the malware class based on malware operation behavior as shows in Table 4.

The right and effective antidote is necessary in combating the malware. The antidote is produced by the other system in malware detection mechanism. Although the antidote is not produced by the classification system, the classification system can work together with other system in preparing the right antidote by looking at the malware class. The reason is, we need to find the cause and effect before come out with the solution which is antidote in our cases.

Table 4. Proposed malware class based on malware operation behavior

Class Operation	Rank	Attacked examples	Affected examples
Data	1	Malware attack office and adobe file	.doc, .ppt, .xls and .pdf file
Application	2	Malware attack application such as office application, audio application and video application	Microsoft Office, Winamp and Windows media Player
System	3	Malware attack the entire Operating System	Windows XP, Windows 7 and Windows Server 2008
Dos	4	Malware attack physical hardware and entire machine	CPU usage, memory and network access

In our proposed class, all four classes are related with each other in rank 1 to 4 starting with the Data class and end with the Dos class. If the classifier classified the malware sample in the Data class, the antidote is prepared based on Data operation and if the malware sample are classified in the Application class, the antidote is prepared based on Data and Application operation. It is same with the System and the Dos classes. Antidote is prepared for Data, Application and System if the malware sample is classified in System class and antidote for entire class is prepared if the malware sample is classified in the Dos class.

The reason we proposed malware class based on its operation is to reduce the process flow and cost in malware detection mechanism. For example, if the malware is in the Data class, the detection mechanism does not need to look and prepare the antidote for other classes because that malware only attack file without attempt to attack the application and operating system. If the malware attack Microsoft Office application which is under Application class, habitually it will also affect the office application file under Data class, but not necessary to attack that particular operating system, which is under System class.

However, not all malware will attack based on this rank class but it normally do so. Duplicating a file is considered under Data class but it also consumes and bottlenecks the memory usage, so for that cases, it is classified in the Dos class. The antidote is prepared for the Data and the Dos class only. All the decision of classification is made by DT and is trained by the GA to optimize the classification process.

5 Conclusion

In this paper, we have proposed a new framework for optimizing malware classification by using GA. By using this new way of classification system, new and unique types of malware can be classify by checking the similarity of malware target and its operation behavior. We also proposed a new malware classes which mainly based on malware operation behavior. This new classification system is important because current method of classification cannot detect and classify unique malware, hence drop down the performance of anti-malware products. There are several limitations in this research. We only focus on host-based machine and windows based operating system. The test subject is also limited to Worms and Trojan horses only. In order to improve efficiency and better classification performance, this research will continue with other domains and other malware types.

Acknowledgments. The author would like to thanks to the members in the Security Research Group, Universiti Sains Malaysia for their helpful discussion and suggestion. This work is supported by Short-term Grant No.304/PKOMP/639021.

References

1. Gheorghescu, M.: An Automated Virus Classification System. In: Virus Bulletin Conference Microsoft (2005)
2. Aycock, J.: Computer Virus and Malware. Springer, Heidelberg (2006)
3. Filiol, E.: Viruses and Malware. In: Handbook of Information and Communication Security, pp. 747–769. Springer, Heidelberg (2010)
4. Rieck, K., Holz, T., Willems, C., Düssel, P., Laskov, P.: Learning and Classification of Malware Behavior. In: Zamboni, D. (ed.) DIMVA 2008. LNCS, vol. 5137, pp. 108–125. Springer, Heidelberg (2008)
5. Apel, M., Bockermann, C., Meier, M.: Measuring Similarity of Malware Behavior. In: The 34th Annual IEEE Conference on Local Computer Networks, pp. 891–898. IEEE Press, Zurich (2009)
6. Preda, M., Christodorescu, M., Jha, S., Debray, S.: A Semantics-Based Approach to Malware Detection. Journal of Transactions on Programming Languages and Systems 30(5) (2007)
7. Szor, P.: The Art of Computer Virus Research and Defense. Symantec Press, Addison-Wesley Professional (2005)
8. Noreen, S., Murtaza, S., Shafiq, M., Farooq, M.: Evolvable Malware. In: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, pp. 1569–1576. ACM, New York (2009)
9. Martignoni, L., Paleari, R., Bruschi, D.: A Framework for Behavior-Based Malware Analysis in the Cloud. In: Prakash, A., Sen Gupta, I. (eds.) ICISS 2009. LNCS, vol. 5905, pp. 178–192. Springer, Heidelberg (2009)
10. Bayer, U., Habibi, I., Balzarotti, D., Kirda, E., Kruegel, C.: A View on Current Malware Behaviors. In: Proceedings of the 2nd USENIX Conference on Large-scale Exploits and Emergent Threats, USENIX, USA (2009)
11. Mehdi, S., Tanwani, A., Farooq, M.: IMAD: In-execution Malware Analysis and Detection. In: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation. ACM, New York (2009)
12. Zolkipli, M.F., Jantan, A.: Malware Behavior Analysis: Learning and Understanding Current Malware Threats. In: Second International Conference on Network Applications, Protocols and Services, pp. 218–221. IEEE Press, Kedah (2010)
13. Edge, K., Lamont, G., Raines, R.: A Retrovirus Inspired Algorithm for Virus Detection and Optimization. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp. 103–110. ACM, New York (2006)
14. Zhao, H., Xu, M., Zheng, N., Yao, J., Ho, Q.: Malicious Executable Classification Based on Behavioral Factor Analysis. In: Proceedings of the 2010 International Conference on e-Education, e-Business, e-Management and e-Learning, pp. 502–506. IEEE Press, Sanya (2010)

15. Zolkipli, M.F., Jantan, A.: A Framework for Malware Detection Using Combination Technique and Signature Generation. In: Proceedings of the Second International Conference on Computer Research and Development, pp. 196–199. IEEE Press, Kuala Lumpur (2010)
16. Panda Security Lab. One third of existing computer viruses were created in (January-October 2010) Panda, `http://www.channeltimes.com/story/one-third-of-existing-computer-viruses-were-created-upto-october-2010-panda/`
17. F-Secure IT Threats Security Summary, `http://www.f-secure.com/en_EMEA-Labs/news-info/threat-summaries/`
18. McAfee Labs, Malware Is Their Business...and Business Is Good, `http://blogs.mcafee.com/mcafee-labs/malware-is-their-businessand-business-is-good`
19. Kaspersky Security Bulletin. Malware Evolution (2010), `http://www.securelist.com/en/analysis/204792161/Kaspersky_Security_Bulletin_Malware_Evolution_2010`
20. Sophos – Security Threats Report: Mid-Year (2010), `http://www.sophos.com/sophos/docs/eng/papers/sophos-security-threat-report-midyear-2010-wpna.pdf`
21. Cyber War - Much Ado about Nothing or the Real Deal? `http://www.invincea.com/blog/2010/07/cyber-war-much-ado-about-nothing-or-the-real-deal/`
22. G-Data - Number of New Computer Viruses at Record High, `http://www.gdatasoftware.co.uk/about-g-data/press-centre/news/news-details/article/1760-number-of-new-computer-viruses.html`
23. ESET Threat Center, `http://www.eset.com/us/threat-center`
24. Vinod, P., Laxmi, V., Gaur, M.S.: Survey on Malware Detection Methods. In: Proceedings of the 3rd Hackers' Workshop on Computer and Internet Security, IITK, Kanpur, India (2009)
25. Zhang, Q., Reeves, D.: MetaAware: Identifying Metamorphic Malware. In: Twenty-Third Annual Computer Security Applications Conference, pp. 411–420. IEEE Press, Miami Beach (2007)
26. Han, S., Lee, K., Lee, S.: Packed PE File Detection for Malware Forensics. In: 2nd International Conference on Computer Science and its Applications, CSA, Korea (2009)
27. Alazab, M., Venkataraman, S., Watters, P.: Towards Understanding Malware Behavior by the Extraction of API Calls. In: Second Cybercrime and Trustworthy Computing Workshop, pp. 52–59. IEEE Press, Ballarat (2010)
28. Desfossez, J., Dieppedale, J., Girard, G.: Stealth Malware Analysis from Kernel Space With Kolumbo. Journal of Computer Virology 7(1), 83–93 (2011)
29. Liu, L., Chen, S.: Malyzer: Defeating Anti-detection for Application-Level Malware Analysis. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 201–218. Springer, Heidelberg (2009)
30. Lau, B., Svajcer, V.: Measuring Virtual Machine Detection in Malware Using DSD Tracer. Journal of Computer Virology 6(3), 181–195 (2010)
31. Daewon, K., Ikkyun, K., Jintae, O., Jongsoo, J.: Behavior-Based Tracer to Monitor Malicious Features of Unknown Executable File. In: Fifth International Multi-Conference on Computing in the Global Information Technology, IEEE Press, Spain (2010)

32. Wang, C., Pang, J., Zhao, R., Fu, W., Liu, X.: Malware Detection Based on Suspicious Behavior Identification. In: First International Workshop on Education Technology and Computer Science, pp. 198–202. IEEE Press, Wuhan (2009)
33. Farid, D.M., Harbi, N., Rahman, M.Z.: Combining Naive Bayes and Decision Tree for Adaptive Intrusion Detection. International Journal of Network Security & Its Applications 2(2), 12–25 (2010); arXiv.org
34. Mezghani, D., Boujelbene, S., Ellouze, N.: Evaluation of SVM Kernels and Conventional Machine Learning Algorithms for Speaker Identification. International Journal of Hybrid Information Technology 3(3), 23–34 (2010)
35. Komashinskiy, D., Kotenko, I.: Malware Detection by Data Mining Techniques Based on Positionally Dependent Features. In: Proceedings of the 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing, pp. 617–623. IEEE Press, USA (2010)
36. Hall, P., Park, B., Samworth, R.: Choice of Neighbor Order in Nearest-Neighbor Classification. Journal of the Institute of Mathematical Statistics 36(5), 2135–2152 (2008)
37. ThreatExpert - TrojanDropper:Win32, `http://www.threatexpert.com/report.aspx?md5=045f8c12b349dafa8c0180a9237f5319`
38. Cha, S.-H., Tappert, C.: A Genetic Algorithm for Constructing Compact Binary Decision Trees. Journal of Pattern Recognition Research 4(1), 1–13 (2009)
39. Windows XP Mode, `http://www.microsoft.com/windows/windows-7/features/windows-xp-mode.aspx`