The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 17, No. 6, pp.11-17, Dec. 31, 2017. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2017.17.6.11
JIIBC 2017-6-2

Opcode와 Windows API를 사용한 멀웨어 탐지

Malware Detection Method using Opcode and windows API Calls

안태현*, 오상진*, 권영만**

Tae-Hyun Ahn*, Sang-Jin Oh*, Young-Man Kwon**

요 약 본 논문에서는 멀웨어 탐지 방법으로 Opcode (operation code)와 실행 파일에서 추출한 Windows API Call로 구성된 특징 벡터를 사용하는 방법을 제안한다. 먼저 PE 파일에서 추출한 opcode와 windows API로 특징 벡터를 구성하고 Bernoulli Naïve Bayes과 K-Nearest Neighbor 분류기 알고리즘을 사용하여 성능을 각각 측정하였다. 실험 결과, 제안한 방법과 KNN 분류기를 사용하여 분류하면 95.21%의 멀웨어 탐지 정확도를 얻을 수 있었다. 결과적으로 기존의 Opcode 또는 Windows API 호출 중 하나만 사용하는 방법보다 제안한 방법이 멀웨어 탐지 정확도에서 높은 성능을 보인다.

Abstract We proposed malware detection method, which use the feature vector that consist of Opcode(operation code) and Windows API Calls extracted from executable files. And, we implemented our feature vector and measured the performance of it by using Bernoulli Naïve Bayes and K-Nearest Neighbor classifier. In experimental result, when using the K-NN classifier with the proposed method, we obtain 95.21% malware detection accuracy. It was better than existing methods using only either Opcode or Windows API Calls.

Key Words : Malware, PE File Format, Opcode, Windows API Calls, Machine Learning, Bernoulli Naïve Bayes, K-Nearest Neighbor.

Ⅰ. INTRODUCTION

The term “Malware” stands for malicious software, and it usually means the hostile software application. The malware can be discriminated by the capability of replication, propagation, self-execution and corruption of the operating system[1]. According to McAfee's latest report, the large number of new samples for the malware are being distributed every day[2].

Because of the large number of samples for new malware, there are many difficulties in detecting and analyzing malware. There is a classical and popular way to detect malware, which is signature based method[3]. Signature was extracted manually from large number malware sample data by heuristic-based analysis. However, this signature-based detection methods have several disadvantages and require high maintenance costs to continue signature updates.

Recently, many research efforts has been reported on data mining techniques[4]. These methods use the

*준회원, 을지대학교 의료IT학과
**종신회원, 을지대학교 의료IT학과(교신저자)
접수일자: 2017년 9월 28일, 수정완료: 2017년 10월 28일
게재확정일자: 2017년 12월 8일

Received: 28 September, 2017 / Revised: 28 October, 2017 / Accepted: 8 December, 2017
**Corresponding Author: ymkwon@eulji.ac.kr
Dept. of Medical IT, Eulji University, Korea

many kind of feature extraction methods and data mining algorithms. In this paper, we used Opcode and Windows API Calls as the feature vector. And we apply Naïve Bayes (NB) and K-Nearest Neighbor (K-NN) algorithms for usefulness of our feature vector. As a result, we get the better result than the existing features that is used by same algorithms.

II. RELATED WORK

Malware detection methods can be divided by Static Analysis and Dynamic Analysis. Static analysis is the testing and evaluation of an application by examining the code without executing the application. So in this method, the performance of detection depends on feature vectors of files. On the other hand, dynamic analysis is the testing and evaluation of an application while executing the program in the virtual environment. It reveals subtle defects or vulnerabilities whose cause is too complex to be discovered by static analysis[5]. In this paper, We used Static Analysis, that is automated-behavior based malware detection using machine learning algorithm.

1. PE File Format

The PE (Portable Executable) file format is an executable file format such as EXE, DLL, Object code used in the Windows operating system. PE files were created by Microsoft based on the COFF (Common Object File Format), and transferred to PE file format along with the introduction of the 3.1 operating system on Windows NT. In the PE file format, the PE header consists of DOS Header, DOS Stub, NT Header, Section Header, and the section below it is PE body.

The section data of the PE body part is divided into ordinary code (.text), data (.data), and resource (.rsrc) sections and saved[6]. The overall PE file format is shown in figure 1. For execution, the PE file is loaded (or mapped) in memory and the section including code and data of the executable file has different shapes such as size and position on the process of loading. That is, the PE file format is distinguished from the file and memory format.

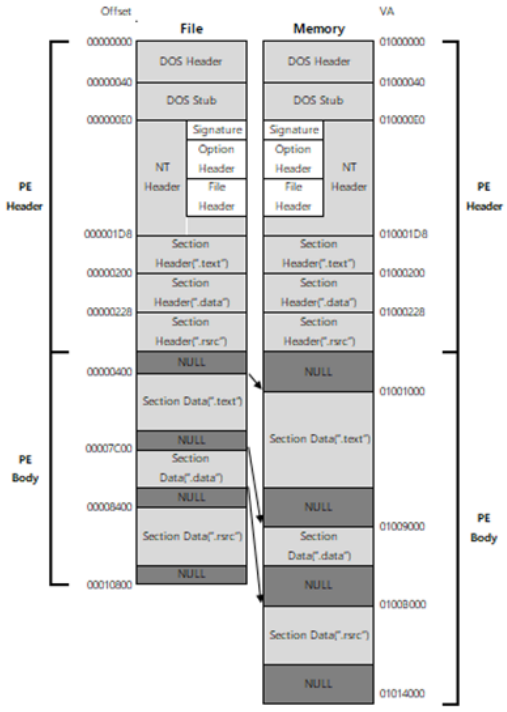
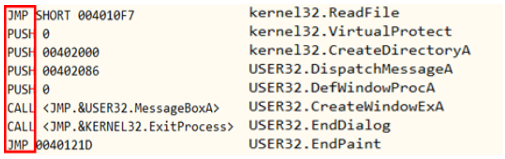


그림 1. PE파일 형식
Fig. 1. PE file format

2. Feature vectors

There are several kinds of feature vectors for the file to detect malware. These used Opcode, Windows API Calls, byte sequence and Header etc.[4]. In Santos etc. al. [7], they used a frequency of the appearance of opcode sequences. Alazab etc. al. [8] introduced the method of classifying whether or not the executable file is malware by using the n-gram statistical analysis of the binary contents. And, they used the Support Vector Machine (SVM) classifier with n-gram that was varied from 1 to 5. Alazab etc. al. [9] had proposed and evaluated a novel method of employing several data mining techniques to detect and classify zero-day malware with high levels of accuracy and efficiency based on the frequency of Windows API Calls.



(a) Opcode (b) Windows API Calls

그림 2. 특징 벡터들
Fig. 2. Feature Vectors

The Opcode is a core part of a machine instruction that embodies an operator executed by a machine and provides functions of logical operation, program flow control, memory processing, and arithmetic operation[10]. The Opcode is shown in figure 2 (a). In the Windows API Calls, API stands for Application Programming Interface. This is functions provided by the operating system, programming language and etc. so that it can be used in applications. For example, The Windows API functions consist of over 1000 functions that can create Windows and can opening a files. These APIs was implemented in several program modules or routines that already exist or need to be connected to conduct the tasks requested by a function calls[11]. API Calls is shown in figure 2 (b).

3. Classifier

To detect malware, someone use classifier as addressed in the above (feature vector). Another use clustering method in existing study for spam detection[14]. The purpose of this paper is to design the feature vector and verify the efficiency of it. So, we will simply use the existing classifier, especially BernoulliNB and K-Nearest Neighbor classifier that is implemented in scikit-learn.

The classification is a data mining (machine learning) technique used to predict group membership for data instances. The Naïve Bayes algorithm is based on Bayes' theorem. It is as follows.

$$p(c_i|w) = p(w|c_i) \times p(c_i)/p(w)$$

Where conditional probability $p(c_i|w)$ is posterior probability about belong in which classes when feature vector w of data was given. Also, $p(w|c_i)$ is prior probability, $p(c_i)$ is probability of class, and $p(w)$ is probability of feature vector. When there are several features, joint probability should be computed, but it have many difficulties. The Naïve Bayes is that each of features is assumed to be independent of each other. Then, prior probability is simply computed by the following.

$$p(w|c_i) = p(w_0|c_i) \times p(w_1|c_i) \times \cdots \times p(w_l|c_i)$$

In this paper, we use Bernoulli Naive Bayes use there are two classes (malware or benign). Bernoulli Naïve Bayes implements learning for data that is distributed according to Bernoulli distributions.

K-Nearest Neighbor is based on instance learning. This Algorithm classify whether the software is malware or not by majority decision after searching k instances of nearest neighbors. Therefore in this case, it is important to set the appropriate the number of instance as value of k.

Ⅲ. MPLEMENTATION OF THE PROPOSED SYSTEM

As mentioned above we proposed feature vector as opcode and Windows API calls to detect malware. So we verified the efficiency of the feature vector. The overall algorithm of detecting malware is divided by two steps, extracting feature vector in the file and classifying the file as malware or benign. We used classifier in scikit-lean(Bernoulli Naive Bayes, K-NN).

In order to extract features from the PE file format (exe, dll, etc.), we used the 'pefile' module [15]. At first, we extract opcode by using 'capstone' module [16] that disassembles the machine code. The second, we find Windows API calls goal by using 'pefile' import library module. The overall procedure is described in figure 3 and the detailed procedure is as follows.

Step 1: We find the code section by using 'pefile' module. Then we extract it and feed it into 'capstone' module. It is hex code format.

Step 2: Disassembler, the 'capstone' module, convert

the hex code into the assembly language as showed in figure 3. When using 'capstone', we should use it after checking whether the OS environment of PE file is 32-bits or 64-bits.

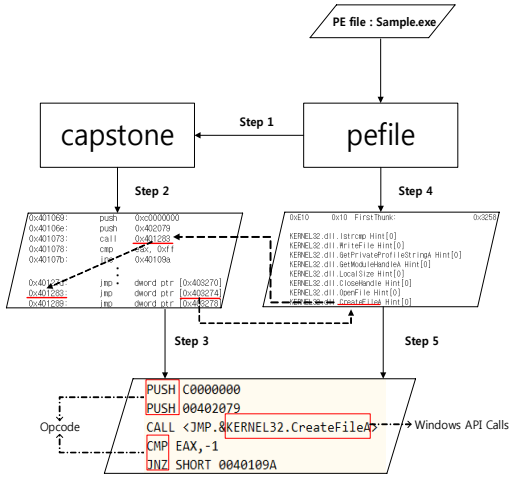


그림 3. 특징 추출 흐름도
Fig. 3. Feature Extraction Flowchart

Step 3: We extract Opcode such as push, cmp, jnz, etc. from assembly language. After this steps, we can get the Opcode as feature. In this step, we excluded call function because it is used to extract Windows API call function. This is explained in step 5.

Step 4: As preprocessing step before step 5, we created the sorted list that consist of DLL's name and the address of FirstThunk from IMAGE_IMPORT_DESCRIPTOR. For example, we can know information such as FirstThunk of Kernel32.dll is 0x3258(shown in figure 3). Where the FirstThunk of IAT reveals the start of RVA for the imported API functions. Also, we prepare the Import Address Table (IAT) as dictionary in order to extract individual Windows API call function.

Step 5: In this step, we look for Windows API Functions of all opcode 'call' by using the sorted list and IAT dictionary created in the 4 step. For example, opcode 'call' at the address '0x401073' has '0x401283' as value of operand. And by using this value of operand, we searched new value of address. Because value of operand indicates new value of address. The operand of new address is 'jmp dword ptr [0x403274]'. And we subtract ImageBase '0x400000' from RVA '0x403274'. In the result, we obtain '0x3274'. To get the FirstThunk of DLL table involve '0x3274', we get the values of FirstThunk smaller than '0x3274' and get the value of FirstThunk which has the largest value among the obtained values through sorted FirstThunks in the step 4. The value of FirstThunk eventually obtained is '0x3258'. And we calculate position of Windows API function using the dictionary of imported Windows API functions created in step 4. In the process of calculating, subtract FirstThunk '0x3258' from the obtained value '0x3274'. Because FirstThunk is composed by dword, we divide the difference into 4. As a result, we could obtain a value of 7, which means that it is positioned 8 th. Eventually Windows API function indicated by the 'call' at address '0x401073' is CreateFileA of kernel32.dll.

Ⅳ. EXPERIMENT AND RESULT

In this paper, we prepared 1224 files as dataset for experiment. Among that, the 445 files are benign and extracted from the Windows file directory (windows \system32). In that directory, we exclude the files with non-PE file format. We crawled 779 malware files from the Web sites of "Virusshare[17]" and "malwareurls. joxeankoret[18]", "malc0de[19]", "malwareblacklist[20]" and used it for learning. Malware consists of Trojan 418, PUP 176, Virus 58, Backdoor 34, Adware 29, Downloader 21, Spyware 13, and so on.

Classifier used Bernoulli Naïve Bayes and K-Nearest Neighbor. In the case of Bernoulli NB, we set value of alpha to 0.5, where the alpha is hyperparameter used in smoothing of maximum likelihood. In the case of K-NN, we used k =2, because performance is highest when k is 2 of 2-5. And, we used Euclidean metric to compute distance between the instances. Also, to measure performance, we used

accuracy, true positive rate and false positive rate.

Experimental results are shown in the table 1. TPR(True Positive Rate) is probability of correctly identified malware. Also, FPR(False Positive Rate) is probability of wrongly identified benign, when a detector identifies benign as a malware. For example, in a medical perspective, TPR is the probability that we accurately diagnosed that cancer patient is suffering from a cancer. On the other hand, FPR is the probability of diagnosing that a person who does not have cancer is suffering from cancer. If FPR is very high, it can diagnose that everyone has cancer. Therefore, in order to obtain reliable results, the TPR must be high and the FPR must be low.

To obtain reliable results, the seed of two classifiers used in the experiment was set to 0 - 10 and each value of result was obtained. And, The average of these results was obtained, which shown as table 1.

표 1. 분류 테스트 결과
Table 1. Result of Classification test

Feature Extraction	Bernoulli Naïve Bayes			K-Nearest Neighbor(k=2)		
	TPR	FPR	ACC	TPR	FPR	ACC
Opcode	0.812314	0.096553	0.844368	0.944919	0.037896	0.951087
Windows API Calls	1.00000	0.184692	0.93330	0.752503	0.120117	0.797183
Proposed method	0.86601	0.086759	0.882905	**0.950263**	**0.044760**	**0.952075**

In the results, when we use Bernoulli Naïve Bayes and the feature vector is set to windows API calls, it shows the highest accuracy of 93.33%. However, since FPR of Bernoulli Naïve Bayes is the highest, the probability of identifying benign as malware is high. Overall, when looking at the accuracy of all classifier, the accuracy of the proposed method is the highest in case of K-NN was used. The result of the proposed method was that TPR is about 95.03%, FPR is about 4.48%, and the accuracy is about 95.21%. The method we proposed could have the highest accuracy results when using the K-NN classifier than the Naive Bayes classifier. They followed figure 4.

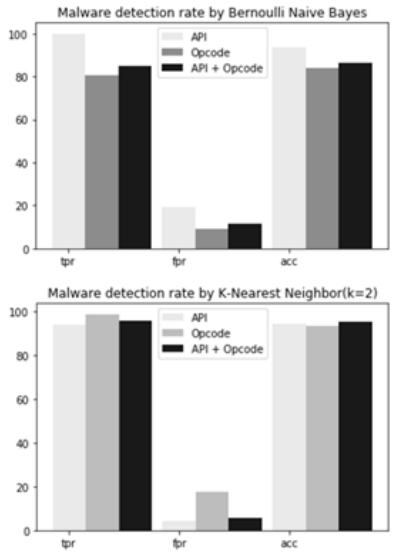


그림 4. 분류기별 멀웨어 탐지 비율 정확도
Fig. 4. Malware Detection Rate by Classifier

V. Conclusion

In this paper, we proposed method that classify whether unknown executable file is malware or not. The method used feature vectors of Opcode and Windows API Calls extracted from executable file and got the better performance than existing methods. One disadvantage is that it takes a long training time because of the high dimension. Therefore, in the future, we need to study feature selection to reduce dimension.

Also, In this paper, we used two classifiers such as Bernoulli Naïve Bayes, K-Nearest Neighbor, but future studies should derive better results through more diverse classifiers.

References

[1] G. Bala Krishna, V. Radha, K. Venugopala Rao, “Review of Contemporary Literature on Machine Learning based Malware Analysis and Detection Strategies,” Global Journal of Computer Science and Technology, vol. 16, Issue. 5, version 1.0, pp

11-16, 2016.

[2] B. Kolosnjaji, A. Zarras, G. Webster, C. Eckert, “Deep Learning for Classification of Malware System Call sequences,” in Australasian Joint Conference on Artificial Intelligence, pp 137-149, 2016.

[3] Z. Bu et al., McAfee Threats Report: Second Quarter 2012, McAfee Labs, 2012.

[4] Ye, Yanfang, et al. "A Survey on Malware Detection Using Data Mining Techniques," ACM Computing Surveys (CSUR) vol.50,no.3, 41p, 2017. DOI: http://doi.org/10.1145/3073559

[5] analysis method, https://software.intel.com/

[6] Seung-Won Lee, Reversing Important Principles: Malware analyst's reversing talk, Insight, pp 141-143, 2012.

[7] I. Santos, F. Brezo, X. Ugarte-Pedrero, PG. Bringas, “Opcode Sequences as Representation of Executables for data-mining-based unknown malware detection,” Information Sciences, vol. 231, pp. 64-82, 2013.
DOI: http://doi.org/10.1016/j.ins.2011.08.020

[8] M. Alazab, R. Layton, S. Venkataraman, P. Watters, “Malware detection based on structural and behavioural features of api calls”, School of Computer and Information Science, Security Research Centre, Edith Cowan University, Perth, Western Australia, 2010.

[9] M. Alazab, S. Venkatraman, P. Watters, M. Alazab, “Zero-day malware detection based on supervised learning algorithms of API call signatures”, Proceedings of the Ninth Australasian Data Mining Conference-Volume 121, pp. 171-182, 2011.

[10] Jeong-been Park, Kyoung-Soo Han, Eul-Gyu Im, “Malware Classification Using Worth Opcodes,” Proceedings of the Korea Information Science 2014 Korea Computer Conference, pp943-945, Jun, 2014.

[11] Yu-Jin Shim, Eul-Gyu Im, “Malware Detection And Classification System based on API Call Sequence,” Ph.D. Thesis. University of Hanyang, Seoul, Republic of Korea 2016.

[12] Python Library, scikit-learn, Bernoulli naïve bayes, http://scikit-learn.org/stable/modules/naive_bayes.html.

[13] Galit Shmueli, Nitin R. Patel, Peter C. Bruce, Data Mining for Business Intelligence, E&Bplus, pp 166, 2006.

[14] Kwon, Y. M., Lee, I. R., Kim, M. G., “A Study on Clustering of SNS SPAM using Heuristic Method”, The Journal of The Institute of Internet, Broadcasting and Communication, 14.6, pp 7-12, 2014
DOI: http://doi.org/10.7236/JIIBC.2014.14.6.7

[15] E. Carrera, Pefile, https://github.com/erocarrera/pefile.

[16] Capstone, capstone, http://www.capstone-engine.org.

[17] virusshare, https://virusshare.com.

[18] joxeankoret, http://malwareurls.joxeankoret.com.

[19] malc0de, http://malc0de.com.

[20] malwareblacklist, http://www.malwareblacklist.com.

※ This research was supported by the R.O.K. National Research Foundation under grant NRF-2017R1D1A1B03036372. We thanks our colleagues from laboratory who provided insight and expertise that greatly assisted the research. and thanks to National Research Foundation who supported laboratory. Especially, I am extremely grateful to the professor who passionately lead to me from beginning to end.

저자 소개

안 태 현(준회원)

- 2016.2 을지대학교 의료IT마케팅학과 학사
- 2016.3 ~ 을지대학교 의료IT학과 석사 과정 재학 중

오 상 진(준회원)

- 2012.3 ~ 을지대학교 의료IT학과 학사 과정 재학 중

권 영 만(종신회원)

- 1985.2 KAIST 전기및전자공학과 석사
- 1998.2 KAIST 정보통신공학과 박사수료
- 2007.2 광운대학교 전자공학과 박사
- 1993.3 ~ 을지대학교 의료IT학과 교수