Malware Detection Based on Gray Scale Image and OpCode Feature

Liu Liu[1], Baosheng Wang[1], Bo Yu[1] and Yong Tang[1]
[1] School of Computer, National University of Defense Technology
Changsha 410073, China
E-mail: [1]hotmailliuliu@163.com

Abstract: In order to evade the detection of security software, the malware authors often adopt the obfuscation technology by reusing well-known malicious code. The diversity of their variants has become a big challenge for traditional analytical methods. Therefore, we propose a novel method which investigates the gray scale and structural features of assembly file. Firstly, our method transforms the malware binary information into gray scale matrices. Then, we use control flow graph to analyze the operation code. Finally, we classify the malware by combining the gray scale matrices and the opcode features. The quantitative experiments with the dataset of 11740 malware show that the combined method improves the accuracy of 11.4% and 5% respectively on average, compared to use gray scale matrix and OPcode features individually. By applying gray scale, opcode and the combined method to seven classic classifiers, we find ensemble learning have the best performance to distinguish malware families. The accuracy rate of Random forest and B are 0.9889 and 0.975 respectively.

Keywords—**Gray scale image, Machine learning, Malware classification, Control flow graph**

1. Introduction

With the rapid development of the black market and obfuscation techniques, the number and variety of malicious codes are growing exponentially, which blames for huge economic losses. According to Ann-day laboratory reports, they caught 3109 new malware families and the 2,243,062 kinds of new variants. Therefore, malicious code detection technology has been widespread concern. Every year a large number of studies aimed at improving the ability to analyze malicious code. [1] [2] [3] and other articles propose some methods to improve the detection rate on different aspects of malicious code. At present, most security vendors use the signature method to detect the malware. This method requires experienced professionals, and it takes much time to constantly update signatures. The method clearly cannot meet the requirements of big data.

In order to better solve the problem of malware, numerous scholars have applied various feature extraction methods to the machine learning [3][7]. The machine learning is able to be divided into two parts: classification and clustering. Clustering is the unsupervised method. It has two advantages: one is that the sample doesn't need to be labeled; the other is to be able to detect abnormal samples. So it usually is used to find zero-day vulnerability. Classification is a supervised model which requires a large of samples labeled. Although Machine Learning provides an efficient way for automatic malware detection, it still faces two important challenges: (1) how to select the suitable malware features for the learning model; (2) how to modify the parameters of the model so as to obtain the good performance.

This paper proposes a novel method to extract the features of malicious code. First, we map the binary executable file into a gray scale image, which is compressed in equal proportion. Then, Opode n-gram is improved by utilizing the method of control flow graph. Finally, we combine the gray scale matrices and the Opcode feature as the input for machine learning. In the experimental part, we use a variety of classical algorithms to test our method, and the results show that our algorithm has a good advantage.

The rest of this paper is organized as follows. In section 2, the related work is introduced on malware classification. In section 3, our methods are described. In section 4, a lot of experiments show that our system has the ability to accurately classify malware and detect the new malware. The section 5 is the summary of this paper.

2. Related work

With the success of machine learning in the image processing and speech recognition [11][12], it also becomes the main method for automated analysis of malware. The input of machine can be divided into two categories: the static features and dynamic features. The static feature is obtained by a static analysis, and it is different from the dynamic feature is that it does not need to run malware. The static feature is vulnerable to confusion technology. The dynamic feature can reflect the function of the malware and is not easy to be influenced by obfuscation. But the extraction of dynamic features spends much time, and may be limited by the operation system.

Kinable et al. [1] used the clustering method to study the malicious code, and used the function call graph to extract the feature of the malware. Finally, the k-medoids and DBSCAN methods are used to measure the similarity, and the results show that the DBSCAN algorithm has better anti-noise effect. Hu X et al. [2] proposed a name of MutantX-S which used instruction sequences code as the input of the clustering algorithm. In this paper [4], the authors designed a system which can automatically extract API call features by using n-gram byte. In the experimental part, the article used SVM to classify the sample set into malicious code or benign, and its accuracy rate reaches 96.5%. Sami A et al. [5] also developed a PE analyzer, which can automatically extract calls API features. And the paper uses the improved score Fisher to reduce the features. It uses the naive Bayes, and, Random Forest J48 and additional algorithms to test their work. The results show that their method improves the previous work in three

aspects: accuracy by 5.24%, detection rate by 2.51% and false alarm rate is decreased from 19.86% to 1.51%. Damodaran A et al. [6] used HMMs to classify the malicious code. The contribution of this paper is to train and test the static and dynamic features of the malicious code by using the hybrid strategy. Experimental results show the best detection rate can be achieved using the dynamic training set and dynamic testing. Due to successful application of deep learning in image recognition, Deep learning is also being used to detect malware. Saxe [10] proposed a deep leaning model to detect the malware based on two dimensional binary program features. The mode is a deep feedforward neural network which has four layers. The first three layers are composed of 1024 nodes. And the activation function of the first two layers is PReLU. The fourth layer is the output. Their experiments show that the model achieves a 95% detection rate at 0.1% false positive rate.

3. Methods

3.1 Overview

This paper aims to label the malware family by finding the shared patterns. As shown in the figure 2, we construct an automatic classification system, which is composed of three steps. In the first step, we map the binary malware into a gray scale image. Then, the OpCode features are extracted using 3-gram, where the malware is translated into the assembly codes by IDA Pro. And the OpCode features are reduced by our rules. Finally, we combine the image with the OpCode features to become the input of the classifiers.

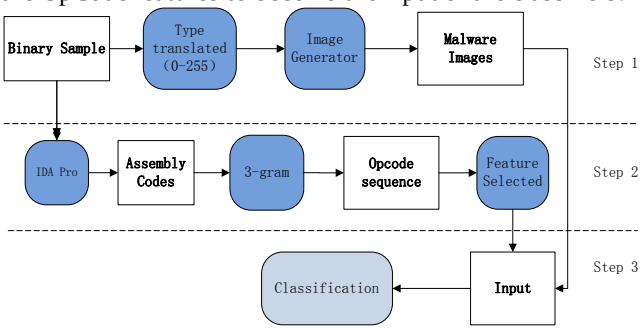

Figure 1. Overview

3.2 Gray scale image

Texture is a kind of visual features that can reflect the nature of the image, which does not depend on the color and brightness. The texture has three main features: the repeatability of local sequence, the non-random arrangement of the basic part and the approximate uniform in the area. The texture description of the malicious code attempts to convert binary malicious executable files into the form of gray scale images [8][9]. We use 8bit as the unit to convert the binary PE file into an unsigned integer matrix, and each element of the matrix belongs to [0, 255]. The width of the matrix depends on the size of the file. Since our executable files are compressed to the size of 1M, the matrix width is set to 512 columns. In the matrix, different values represent different gray scale, where 0 represents black and 255 represents white. So the matrices also are called the gray scale matrix.

In Figure 2, three malware which belong to two different malicious families are transformed into gray scale images. Apparently, the first two pictures belong to the same family from the texture. The red area in the graph shows the similar texture features between them. It should be explained that these three gray scale images have been reduced in proportion, but this does not affect their visual effect. Compared with the first two pictures, the texture of the last pictures is not similar to other graphs, so it belongs to different malicious families.

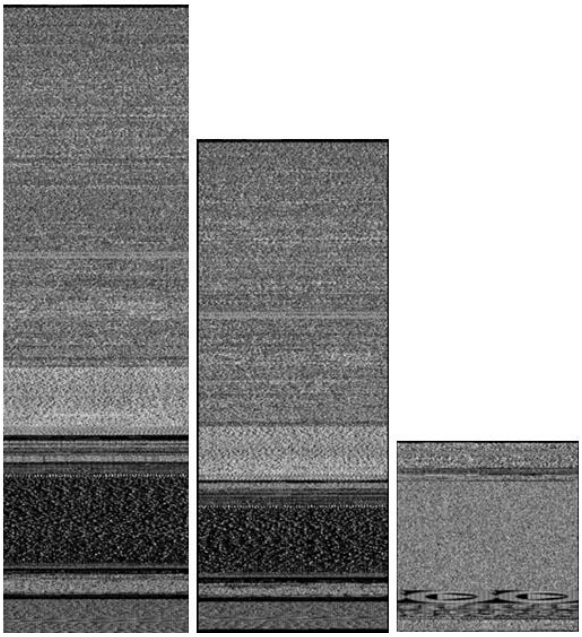

Figure 2. Malware texture

3.3 OpCode and Combined

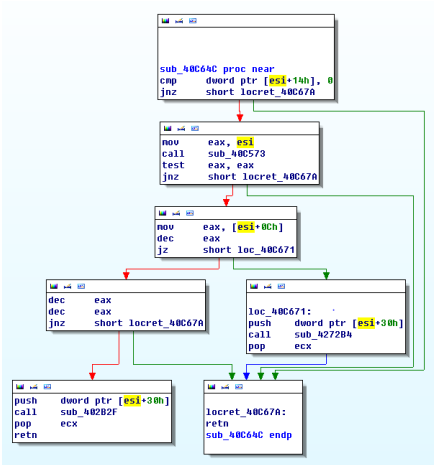

Figure 3. CFG

First of all, we can get the assembly form of executable files by Pro IDA. Then the control flow graph is used to describe the structural features of malicious code, including the function of the call relationship, the loop of the function

and the code block boundary, etc. The following figure is the control flow graph of a function from the malware which is called VIRUS.Wind32.Afgan.c. In the figure, the green arrows represent the successful jumps. The red arrows represent the failed jumps. And the blue arrows are the unconditional jumps.

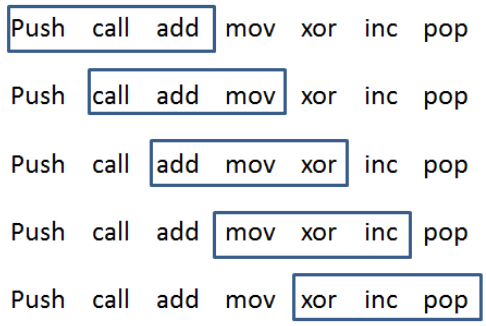

Figure 4. 3-gram.

We extract the architectural features of the operation code by n-gram which is used to extract the features of the text. And n denotes a size of a window. The n-gram model extracts the continuous n words as a feature by sliding a window. For example, there is a set of OpCode : {push, call, add, mov, xor, inc, pop}. In Fig 3, the 3-gram is employed to abstract the feature of the set. Because a malware usually have hundreds of thousands of OpCode, the number of features obtained by n-gram is enormous. Therefore we propose three reduction conditions: 1) In all malware families, high frequency features should be deleted 2) in all malware families, low frequency features is deleted; 3) The low frequency features exists in a few malicious family is deleted. We assume that $Y = \{y_0, y_1, y_2, \ldots, y_n\}$ represents the labels of malware family. s_i refers to a feature sequence. $f_j(s_i)$ is the frequency of s_i which belongs to the family y_j. $F(s_i)$ is the frequency of s_i in Y. We can convert these conditions to the inequation $\left|1 - \frac{F(s_i)}{f_j(s_i)}\right| \leq \xi$, where ξ is set according to the specific sample.

Finally, we combine the OpCode and gray scale image. The image is stored in the form of a matrix. The matrix must be map into one-dimensional vector. The different domains of attribute values may lead to poor classification results. Therefore, we normalize the feature value and the normalized formula is $F' = \frac{F - min_F}{max_F - min_F}$, where min_F and max_F respectively represent the minimum and maximum values of the feature F.

4. Experiment

Experimental data are collected online by a variety of anti-virus software in the last two months. The experimental data have 11740 malicious codes, which are derived from 9 different malicious families. We use Pro IDA tool to convert the executable files of malicious code into assembly files. And the binary files are converted into gray scale images. After using our method to extract features, the malicious code has three kinds of feature set, which are: gray-scale matrix, opcode 3-gram, and the combination. Experiments used 7 classic algorithms from the python machine learning library, such as Forest Random, K-Nearest Neighbour (K-NN), Gradient -Boosting Classifier, Logistic Regression, SVM-poly, Naïve Bayes and Decision-Tree. Forest Random and Gradient-Boosting Classifier are the ensemble learning. The ensemble learning integrates multiple classifiers to determine the classification result by voting system. This method is easy to be applied, and it can improve the overall performance of the classifiers.

Table 1. Accuracy rate

	Image	OpCode 3-gram	Combined
Forest Random	0.9583	0.9361	0.9889
K-NN	0.8722	0.8306	0.8806
Gradient-Boosting Classifier	0.9611	0.9222	0.975
Naïve Bayes	0.775	0.7722	0.9528
Logistic Regression	0.9278	0.8722	0.9583
SVM-poly	0.9361	0.8	0.9694
Decision-Tree	0.9170	0.7693	0.9751

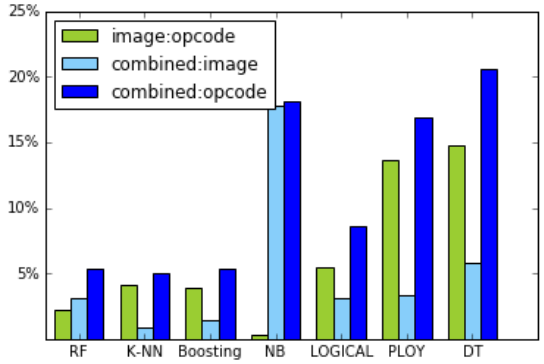

Figure 5. Performance comparison chart.

Table 1 shows the experimental results of seven algorithms on the dataset. The accuracy of the joint characteristics is the best. And the average accuracy of the combined method is 5% and 11.3% higher than the other two. The accuracy of gay scale image is 6.4% higher than the OpCode method. Fig 2 shows the degree of accuracy of the algorithm is improved by the use of the three methods. The performance improvement of Decision-Tree (DT) and Naïve Bayes (NB) is the most obvious. It is worth noting that the accuracy of Forest Random and Gradient -Boosting Classifier is the best. This suggests that the ensemble learning has better performance than the other methods.

5. Conclusions and Future Work

In this paper we have introduced a new feature extraction method for the malware. The method combines the gray scale image and the OpCode feature. Compared with the traditional methods, this method makes use of image texture which can effectively describe the difference between the families and the similarity in the same family. The OpCode feature is based on CFG, which is able to react to the overall structure of malware. In the experiments, we use many classifiers to identify our method. The results show that our method effectively improves the accuracy of

malware classification in the classifiers. In the future, we will build an automated platform, which integrates malware detection and family classification.

Acknowledgment

The author is grateful to Baosheng Wang and QiuxiZhong for the guidance and advice, and thanks to the support of the project.The work was supported by Science Foundation of China under (NSFC) Grant No. National 61303264 and the National Key Basic Research Program of China (973 Program) "Reconfigurable Tested for Basic Network Communication" (2012CB315906).

References

[1] Kinable J, Kostakis O. "Malware classification based on call graph clustering[J]," Journal in computer virology, 2011, 7(4): 233-245.

[2] Hu X, Shin K G, Bhatkar S, et al. "MutantX-S: Scalable Malware Clustering Based on Static Features[C]," USENIX Annual Technical Conference. 2013: 187-198.

[3] Ye Y, Li T, Chen Y, et al. "Automatic malware categorization using cluster ensemble[C]," Proceedings of the 16th ACM SIGKDD internati-onal conference on Knowledge discovery and data mining. ACM, 2010: 95-104.

[4] Alazab M, Layton R, Venkataraman S, et al. "Malware detection based on structural and behavioural features of api calls[J]," 2010.

[5] Sami A, Yadegari B, Rahimi H, et al. "Malware detection based on mining API calls[C]," Proceedings of the 2010 ACM Symposium on Applied Computing. ACM, 2010: 1020-1025.

[6] Damodaran A, Di Troia F, Visaggio C A, et al. "*A* comparison of static, dynamic, and hybrid analysis for malware detection[J]," Journal of Computer Virology and Hacking Techniques, 2015: 1-12.

[7] Rieck K, Trinius P, Willems C, et al. "Automatic analysis of malware behavior using machine learning[J]," Journal of Computer Security, 2011, 19(4): 639-668.

[8] Kancherla K, Mukkamala S. "Image visualization based malware detection[C]," Computational Intelligence in Cyber Security (CICS), 2013 IEEE Symposium on. IEEE, 2013: 40-44.

[9] Han K S, Lim J H, Im E G. "Malware analysis method using visualization of binary files[C]," Proceedings of the 2013 Research in Adaptive and Convergent Systems. ACM, 2013: 317-321.

[10] Saxe J, Berlin K. "Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features[J], " arXiv preprint arXiv: 1508.03096, 2015

[11] Xin-wang LIU, Lei WANG. "Multiple kernel extreme learning machine[J]," Neurocomputing. 2015, 149(Part A): 253-264.

[12]Wen YAO, Xiao-qian CHEN, ZHAO Yong, "Efficient resources provisioning based on load forecasting in cloud [J]," IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, 2012, 23(2): 247-259.