International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Enhanced Training Phase Reduction with Feature Filtering for Malware Detection Using Ensemble SVM

Shital Kuber[1], Prof. Digambar Padulkar[2]

[1]Savitribai Phule Pune University, Department of Computer Engineering, VPCOE, Baramati, Maharashtra, India
[2]Savitribai Phule Pune University, Department of Computer Engineering, VPCOE, Baramati, Maharashtra, India

Abstract: ***Malware is defined as software which is used with the aim of attempting to break the computer systems security policy with respect to confidentiality, integrity or availability. Thus malware detection is the vital issue in the computer security. There are various methods for malware detection viz. Signature based detection, Anomaly based malware detection and specification based malware detection. Out of this, Signature based malware detection is more accepted method to detect the malware attack but main drawback of this method is, not used to detect the Zero-day attack. We need to update the data repository regularly and human experts are required to create the signature. SVM classifier addresses this issue. Proposed system represents the idea of opcodes to detect the malware. The input given to the system is taken in the form of *.exe files which are both malware and benign files. Using the dataset the opcodes are generated. Then feature extraction and feature reduction steps are carried out. For feature reduction " Subspace analysis using eigenvectors" method is used. Then Ensemble SVM classification technique is used to perform the searching on all the opcode and decides which type of opcode having positive impact on detecting the malware.Ensemble SVM classifier provides good accuracy to classify malware and benign files as compared to other.***

Keywords: malware, feature extraction, feature reduction, ensemble svm, veto voting, classification

1. Introduction

Malware is common term for any malicious program which enters system without authorization of the users. Modern communication infrastructures are highly vulnerable to many types of malwares attacks. Due to malicious attacks cause several damages to private users, governmental organizations and commercial companies. The proliferation in high-speed internet connections facilitates malware to propagate and infect computer system very rapidly. Once the malware enters into the system, it finds way inside the system, it scans the system and find out the vulnerabilities of operating system. Then perform accidental actions on the system finally reducing the overall performance of the system. In every year the malwares are increasing in an frightening rate. Therefore malware detection becomes a most critical issue in today's computer systems.[1]

1.1 Malware Analysis Technique

Malware analysis is the process of analyzing the purpose and functionality of a malware. The purpose of Malware analysis is to understand the characteristics that all malwares and create a set of signatures in order to detect malwares.

There are two types of malware analysis that security experts perform:

1. Static analysis
2. Dynamic analysis.

1.1.1 Static Analysis

It is a technique that identifies malware program without executing it. With the static malware analysis technique, researcher performs reverse engineering by using disassemble tools, decompile tools, source code analyzer tools such as Hexdump, XXD, IDA pro and Ollydbg [1] in order to understand malware by seeing the structure of malware. Static analysis has an advantage that it can wholly determine the purpose and functionality of malware.

1.1.2. Dynamic Analysis

It is also called as behavioral analysis. Dynamic analysis involves executing the malware and observing its behavior, system interaction, and the effects on the host machine. In dynamic analysis, infected files are analyzed in computer-generated environment like a virtual machine, simulator, emulator, sandbox etc. After that, malware researchers use SysAnalyzer, Process Explorer, ProcMon and other tools to identify the general behavioral analysis techniques.

1.2 Malware Detection Techniques

1.2.1 Signature Based Malware Detection

It maintains the database of signatures. It detects malware by matching pattern against the database. It shall require less amounts of system resources to detect the malwares. This technique only identify the known malware accurately. The disadvantage of this technique is it not effective against the zero day attack means it cannot detect the new, unknown malware as no signature available for such kind of malware. Data mining and machine learning methods are used to overcome this drawback of signature based detection.

Signatures are created by observing the disassembled code of malware binary. Most of the antivirus tools are based on signature based detection techniques.

1.2.2. Heuristic-Based Malware Detection

It is also called as anomaly based detection. The main goal is to analyze the behavior of known or unknown malwares. Behavioral parameters include various aspects such as source or destination address of malwares, forms of attachments and other measurable statistical features. It usually occurs in two phase:

1. Training phase
2. Testing phase.

During the training phase, the behavior of the system is observed in the absence of attack. Machine learning technique is used to create a summary of such normal behavior. In detection phase, this summary is compared against the current behavior, and deviances are identified as potential attacks. A key advantage of heuristic based detection is its capability to detect zero-day attacks.

1.2.3 Specification Based Detection

Specification-based detection is a derived from anomaly based detection. This technique tries to overcome the typical high false alarm rate associated with the anomaly-based detection. Specification-based detection depend on program specifications that define the intended behavior of security-critical programs. It monitors executions program involve and detecting deviation of their behavior from the specification. This technique is similar to anomaly detection where they detect the attacks as deviate from normal. The difference is that instead of machine learning techniques, it will be based on manually developed specifications that capture legitimate system behavior.

2. Literature Survey

A. Shabtai et al. [2] used static analysis approach to study the usefulness of malware detection. For this purpose they used different n-gram size. (n-gram size varies from 1 to 6)with various classifiers. Shabtai's results showed that malware detection rate is high for n=2. opcode n-gram patterns are used to detect the unknown malicious code. Opcode n-grams are generated by feature extraction technique. The feature extraction method is carried out by disassembling the executable files of both benign and malware files. They work onclass imbalance problem also.

Generally for the retrieval and categorization purposes, in textual domain TF-IDF is more successful representation but they found that TFIDF representation introduces extra computational challenges in the preservation of the collection.

D. Bilar [3] investigated opcode frequency distribuations mechanism to detect the malware. He discusses a malware detection mechanism by using n-gram approach through statistical analysis of opcode distribution. His results shows that for detecting the malware less frequent (add, sub, ja, adc etc.) opcodes are best indicators while most frequnt opcodes are bad indicator (move, push, call etc.)

His technique gives a prelude assessment of its usefulness for malware detection. This Technique gives better accuracy for differentiation of polymorphic and metamorphic malware.

D. Bilar [4] analyze the callgraph structure of 120 malware and 200 benign files. He represents each executable file as a graph of graphs. This follows the perception that in any procedural language, the source code is composed into functions This functions can be represented as a flowchart, called as a directed graph. These functions are dependent to each other, To describe this dependency each node is visualize as a function and the edges are calls-to relations between the functions. This graph of graphs is called as the callgraph. The structure of callgraph is obtained by disassembling procedure. By using the disassembling procedure the executable file is converted into individual instructions. He distinguishes between short and far branch instructions: In short branches a return address do not save while far branches a return address is saved. Intuitively, short branches are normally used t control within one function of the program, while far branches are used to invoke other functions. He statically constructs the Control Flow Graph of benign and malicious code.

His finding shows that the basic block count for malicious code is lower and having less interaction. While the basic block count is more in case of benign files. A benign file shows more complex interaction also.

Santos et al. [5] demonstrated that n-gram signatures based approach to detect unknown malware. They found that for n=2, the detection rate is low, for n=4, the detection rate is maximum. In this paper they use a new methodology for malware detection based on the use of n-grams for file signatures creation.

R. Sekar et al. [6] implemented a Finite State Automaton (FSA) method for malware detection. FSA-learning method is computationally expensive and the space usage of the FSA may be too much. To overcome these drawbacks they build compact FSA. The formation of a compact FSA in a fully automatic and skilled manner and without requiring access to source code for programs. The FSA approach is compared with n-gram analysis method. The FSA algorithm having less false positive rate as compared with the n-gram approach. FSA-learning is computationally expensive, and requires much space usage. The algorithm proposed in this project approach builds a compact FSA in a fully automatic and efficient, without requiring access to source code for programs. The space requirements are also reduced. The FSA uses only a constant time per system call during the learning as well as detection period. Due to this factor it having low overheads for malware detection. In FSA algorithm, the order of system calls made from libraries does not preserve .

Wei-jen Li et al. [7] describe n-gram(n=1) analysis, at byte level. N-gram analysis at byte level (N=1) is performed on PDF files with embedded malware. This technique proved an efficient technique for malware detection of PDF files. This method detects the malware embedded at the start or end of a file. However, this technique is failed to detect malware embedded in the middle of the file.

I. Santos et al. [8] proposed the use of a single-class learning method based on opcode sequences for unknown malware detection. In this technique the frequencies of the appearance of opcode sequences is used to build a machine-learning classifier. But only one set of labeled instances within a specific class of either malware or genuine software are taken into consideration.

This method can reduce the effort of labeling software and maintaining high accuracy. Single-class learning method needs several instances that belong to a specific class to be labelled Therefore, Single-class learning method can reduce the cost detecting the unknown malware.

3. System Implementation

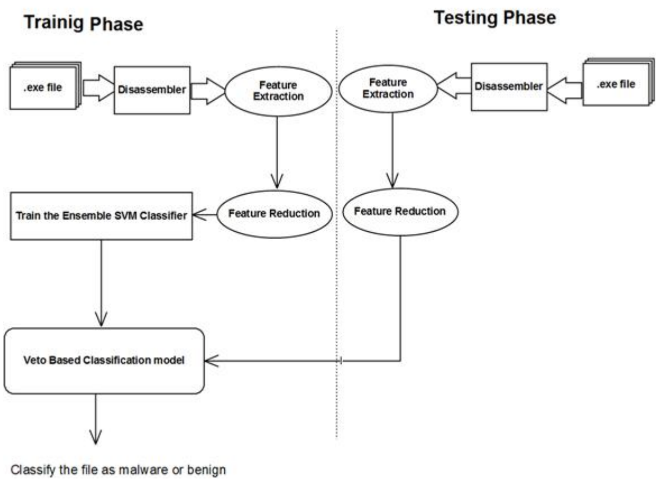


Figure 1:: System Architecture

3.1 Dataset Creation

The Process of translating the machine code instructions stored in executable to a more human-readable language, namely, Assembly language is known as disassembly process. In proposed system architecture the Ollydbg disassembler is used which is the most superior commercial disassembly program available today. ollydbg implements sophisticated techniques which enabled us to disassemble most of our malware and benign collection successfully.

The dataset is constructed by using Ollydbg disassemble. The input for disassembler is taken in the form of *.exe files as shown in figure 1, which are both malware and benign files. The disassembler generates the runtime traces. Runtime traces are in the *.txt format.

3.2 Feature Extraction

Feature Extraction step is carried on runtime traces generated by disassemble process. Runtime traces consist of assembly language instructions i. e. opcodes associated with operands, memory locations or registers. In feature extraction process only opcodes are selected. Operands, memory locations and registers are omitted.

3.3 Feature Reduction

To determine the usefulness of individual opcod for malware detection the eigenvalues and eigenvectors are used to determine the ranking.

Principle Component Analysis is a transformation of Co-variance matrix and it is defined in [1] as

$$C_{ij}=\frac{1}{n-1}\sum_{m=1}^{n}\left(X_{jm}-\bar{X}_i\right)\left(X_{Jm}-\bar{X}_J\right) \quad (1)$$

Where,

C Covariance matrix of PCA Transformation;

X Dataset value;

$\bar{X}$ Dataset mean;

n and m Data length;

The goal of PCA is to find a new set of attributes that better capture the variability of original dataset. PCA tends to identify strongest pattern in the data. The dimensionality reduction is achieved by PCA algorithm. Dimensionality reduction can eliminate much of the noise.

After applying the PCA algorithm, By multiplying the significant eigenvector column by the respective eigenvalues we calculate the significant values and then summing each row.

$$R_k=\sum_{k=1}^{8}V.d_k \quad (2)$$

Where,

R Sum of matrix variance;

C Covariance;

V Eigenvector;

d EigenValue scalar.

3.4 Classification

Every classifier has its own decision. In proposed system there is a committee in classification model. Here we used classifier ensemble SVM which can use method of veto voting to reach the final prediction. It performs better than single classifier and helps to improve the detection accuracy. The decision from more than one expert(classifier} may be required in certain situations, so a committee of experts is formed as it is expected that a committee always performs better than a single expert. Normally committee uses majority voting for combining the decisions of the experts to reach a final conclusion. In some cases, the committee may give the right to veto the decision of the committee to any member. Any vote indicating an instance as malware, alone can determine the outcome of the classification task as malware regardless of the count of other votes. Figure. 2 shows the ensemble SVM approach using veto Voting.

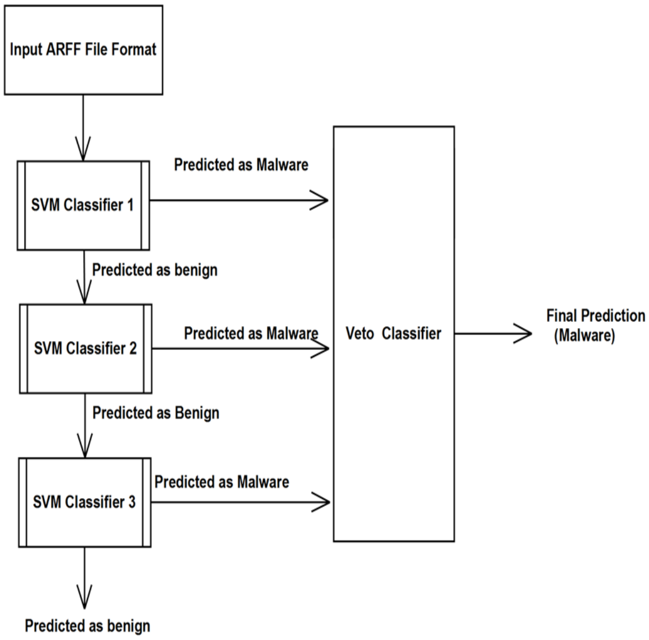
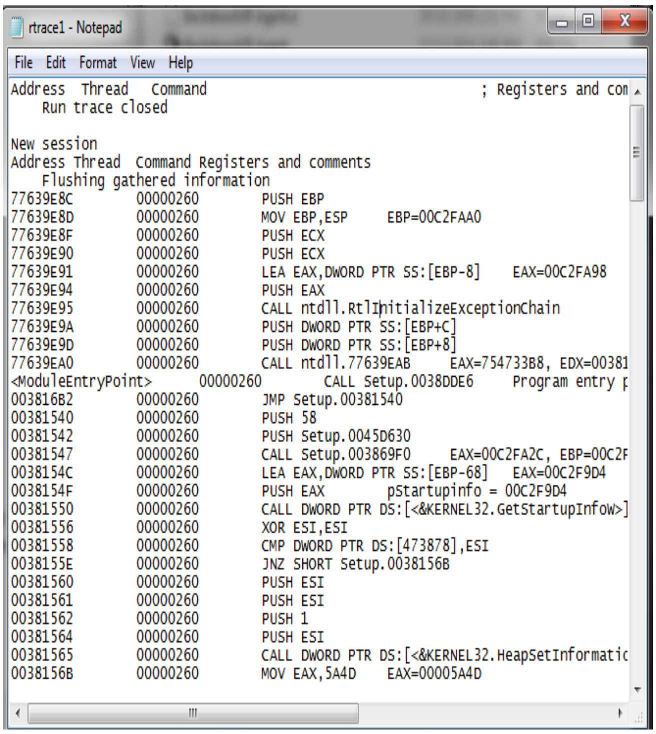


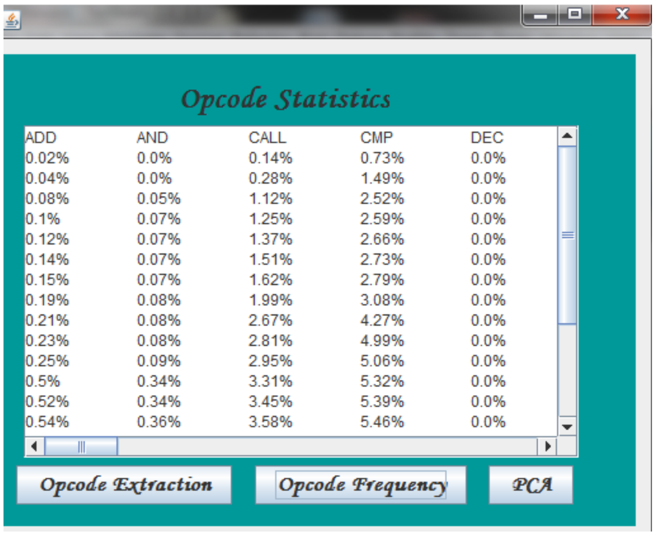
Figure 2:.Ensemble classification approach using veto Voting

4. Results

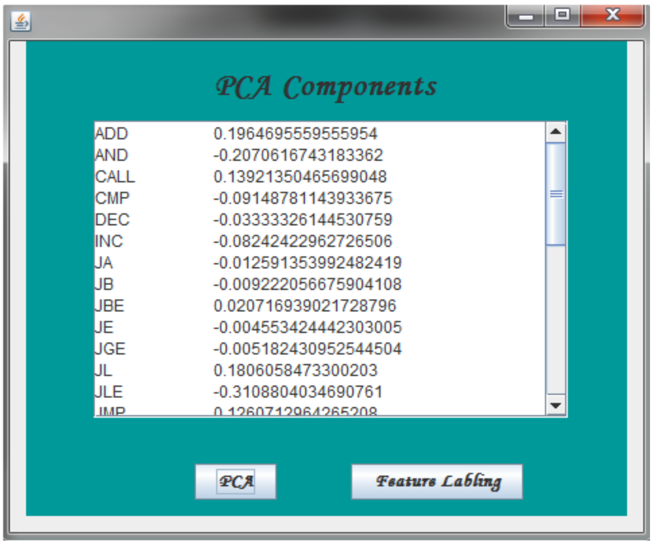
4.1 Dataset Creation using OllyDbg Disassembler



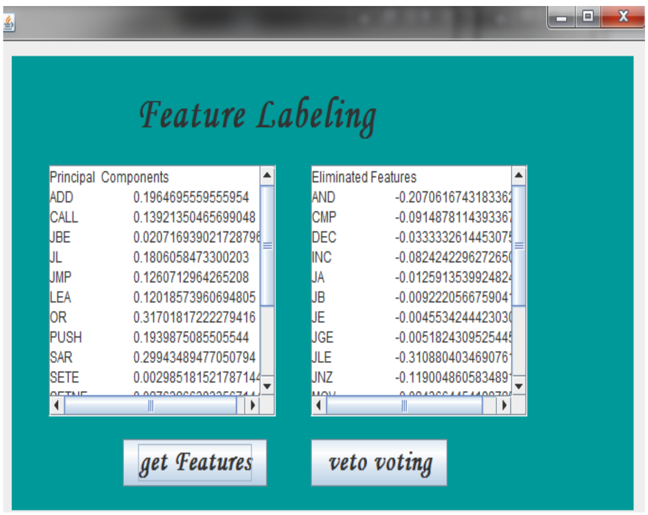
4.2 Opcode Extraction with Occurences for each *.txt file in the dataset



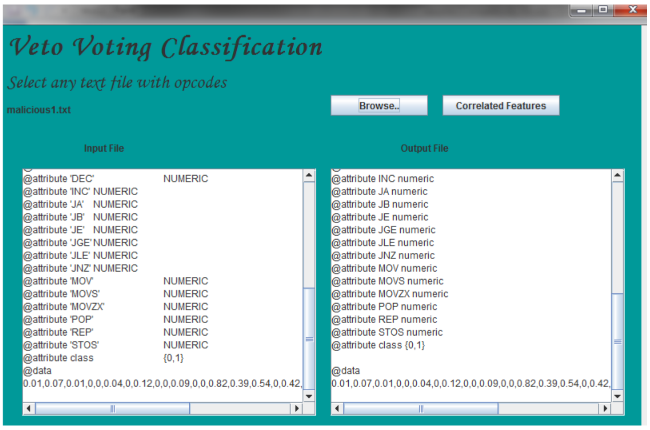
4.3 Principle Component Analysis

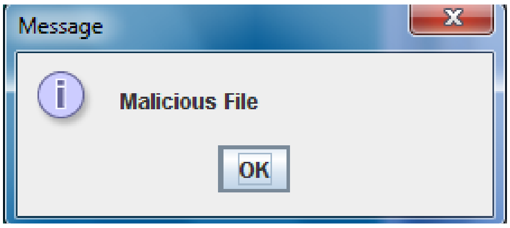


4.4 Feature Labelling By using PCA method



4.5 Veto Voting Ensemble Classification





5. Conclusions

1. The less frequent opcodes having an importance rating for classifying benign and malicious software. While mov has a negative impact on the classification and identification of software.
2. Opcode mov is a poor indicator of benign and malicious software. mov opcodes inhibits the ability to correctly classify software when used with other opcodes. Using the eigenvector prefilter, irrelevant features can safely remove.
3. Ensemble SVM classifier provides good accuracy to detectmalware as compared to other methods.

Acknowledgement

I avail this opportunity to express my deep sense of gratitude and whole hearted thanks to my guide Prof. D. M. Padulkar for giving his valuable guidance, inspiration and encouragement to embark this paper. Without his Coordination, guidance and reviewing, this task could not be completed alone.

References

[1] P. O'Kane, S. Sezer, K. McLaughlin, and E. Im, “SVM Training Phase Reduction Using Dataset Feature Filtering for Malware Detection”, IEEE Transaction on information Forensic And Security, VOL. 8, NO. 3, March 2013.
[2] A. Shabtai, R. Moskovitch, C. Feher, S. Dolev, and Y. Elovici, “Detecting unknown malicious code by applying classification techniques on opcode patterns, Security Informatics, vol. 1, pp. 1-22, 2012.
[3] D. Bilar, “Opcodes as predictor for malware, Int. J. Electron. Security Digital Forensics, vol. 1, no. 2, pp. 156 - 168, 2007.
[4] D. Bilar, “Callgraph properties of executables and generative mechanisms, AI Commun., Special Issue on Network Anal. in Natural Sci.and Eng., vol. 20, no. 4, pp. 231-243, 2007.
[5] I. Santos, Y. K. Penya, J. Devesa, and P. G. Garcia, “N-grams-based file signatures for malware detection,” S3Lab, Deusto Technological Found., 2009.
[6] R. Sekar, M. Bendre, D. Bollineni, and Bollineni, R. Needham and M.Abadi, Eds., “A fast automaton-based method for detecting anomalous program behaviors,” in Proc. 2001 IEEE Symp. Security and Privacy,IEEE Comput. Soc., Los Alamitos, CA, USA, 2001, pp. 144-155."
[7] Wei-Jen Li, W. L. K. Wang, S. Stolfo, and B. Herzog, “Fileprints: Identifying file types by n-gram analysis, in Proc. 6th IEEE Inform. Assurance Workshop,June 2005, pp. : 64-71.
[8] I. Santos, F. Brezo, B. Sanz, C. Laorden, and Y. P. G. Bringas, “Using opcode sequences in single-class learning to detect unknown malware,” IET Inform. Security, vol. 5, no. 4, pp. 220227, 2011.

Author Profile

Ms. Shital Kuber, Received the Bachelors degree(B. E.) computer Engineering in 2013 from VPCOE, Baramati. she is now pursuing M. E. degree, from VPCOE, Baramati.

Prof. Digambar Padulkar is working as Assistant professor, Department Of Computer Engineering, Vidya Pratishthan’s College Of Engineering, Baramati, Pune, Maharashtra.. His research areas include Uncertain Data Mining, Applications of Data Mining in Business and Intelligent predictions.