TELKOMNIKA, Vol. 14, No. 3, September 2016, pp. 1157 ∼ 1165
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/telkomnika.v14.i3.3850

Metamorphic Malware Detection Based on Support Vector Machine Classification of Malware Sub-Signatures

Ban Mohammed Khammas , Alireza Monemi , Ismahani Ismail , Sulaiman Mohd Nor , and M.N. Marsono*
Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310, Johor Bahru, Johor Malaysia
*Corresponding author, e-mail: mnadzir@utm.my

Abstract
Achieving accurate and efficient metamorphic malware detection remains a challenge. Metamorphic malware is able to mutate and alter its code structure in each infection that can circumvent signature matching detection. However, some vital functionalities and code segments remain unchanged between mutations. We exploit these unchanged features by the mean of classification using Support Vector Machine (SVM). N-gram features are extracted directly from malware binaries to avoid disassembly, which these features are then masked with the extracted known malware signature n-grams. These masked features reduce the number of selected n-gram features considerably. Our method is capable to accurately detect metamorphic malware with ~99% accuracy and low false positive rate. The proposed method is also superior to commercially available anti-viruses for detecting metamorphic malware.

Keyword: *SVM classification, Metamorphic, n-gram, Snort*

1. Introduction

Malware is one of security attacks to Internet users as it breaches computer security and data confidentiality, which are categorized into general (non-mutable) and mutable types. Anti-virus softwares rely on signature-based detection as the primary detection mechanism. Mutatable malware such as packing, polymorphic, and metamorphic make the detection based on signature matching difficult. Metamorphic malwares mutate and change their codes structure and signatures in each infection that is difficult to detect [1]. Lately, several host-based dynamical analysis techniques were proposed for metamorphic malware detection [2]. However, these techniques require separate environment to analyze malware in order to be able to be detected. At the same time, the requirement of binary code disassembly in opcode-based methods [3–5] is not suitable for timely metamorphic detection on host-level intrusion detection systems.

We propose metamorphic malware detection based on static analysis of metamorphic malware binaries without disassembly. Features are extracted from binary, which can be in the form of packets payload in network detection system or files in host based detection system using n-gram feature extraction and machine learning SVM classification. Besides, extracted n-gram features are masked with known malware signature n-grams to represent only informative malware features. This technique can reduce the n-gram search space.

This paper is organized as follows. Section 2 provides a critical review on relevant literatures. The methodology for metamorphic malware detection in network and host-based IDS are described in Section 3. Section 4 highlights the experimental setup, datasets, and evaluation criteria. The data analysis and comparison with commercial anti-virus software are presented in Section 5. Section 6 concludes the research findings, and contributions of the paper.

2. Related Work

Host-based anti-viruses (AV) and NIDS such as Snort [6] and Bro [7] primarily rely on signature matching as one of the detection techniques. These tools search for specific signatures

Received April 5, 2016; Revised July 16, 2016; Accepted July 30, 2016

in the file or packet payload to detect malware and other types of attacks [6]. However, the inability to detect metamorphic malware or previously unseen malware without known signatures is the main limitation of signature-matching methods [5,8]. Metamorphic malware is a type of advanced mutating malwares that change their structure in each mutation. These changes may contain small number of instructions for specific functionality or enclose multiple instructions to perform similar functionality. In each mutation, these instructions are expanded or minimized according to obfuscation techniques used. The taxonomy of the morphing techniques that are used in metamorphic malware include insertion of redundant code, dead instructions, NOP instructions, unreachable code, reordering of instructions, register swapping, and substituting instructions with equivalent instructions. The detection of mutating malware is non-trivial [3,5,9].

Several researches proposed the use of machine learning (ML) to detect malware [10, 11]. Schultz et al. [12] in their pioneering study used ML technique to detect malware, where different feature extraction methods such as string features, program header, and byte sequence features were investigated. Kolter and Maloof [11] used n-gram features extraction method and improved the detection accuracy by combining n-gram technique with ML to classify malware executable files (worm, virus, and trojan). The n-grams features are extracted and selected from malware byte code using the Information Gain (IG) method. Furthermore, the effects of several ML classifiers (naive Bayes, instance based learning, SVM, decision tree, and boosted classifier) as well as the size of n-grams on the classification accuracy were analyzed. The 4-gram was reported to offer the best accuracy.

Shabtai et al. [13] used n-gram features that extracted from opcode instead of byte code. They examined the influence of different n-gram sizes (1 to 5) with assorted feature selection and classifiers. The effect of term frequency inverse document frequency (TFIDF) and normalized TF were compared. It is reported that TFIDF and TF produced almost identical results. Santos et al. [10] proposed the use of opcode-sequence frequency to represent features based on sampled malwares and normal files, where top 1000 features are selected using IG. Several ML classifiers such as decision trees, SVM, k-nearest neighbours, and Bayesian were used to analyze the data set.

Generally, static metamorphic malware detection uses opcodes or opcode n-grams as extracted features for different detection techniques such as ML and statistical analysis [3–5, 14]. Lately, control flow based method such as by Alam et al. [1] statistically analyzed the performance of host-based metamorphic malware detection. In their method, the dataset is translated into an intermediate language called MAIL after disassembling the program binary. The detection accuracy for the dataset containing 1020 metamorphic files and 2330 normal files was reported to be 94.69% with false positive ratio (FPR) of 10.59. Runwal et al. [15] proposed opcode graphs technique to find the similarity of executable file for detecting metamorphic malware. Opcode sequences are extracted and weights are measured based on the frequency of opcode occurrences.

Most existing literatures used opcode-based features to detect metamorphic malware despite their computational cost. For instance, opcode-based methods require disassembly of binary code to obtain the opcode features, which are not suitable for timely metamorphic detection, especially when targeting for network-level detection. To circumvent this shortcoming, we propose the use of ML and n-gram term frequency [10, 13] by modifying the feature extraction and selection processes. The n-gram search space is further minimized via a two-stage feature selection scheme. First, features that match the sub-signature features are selected. Second, the most effective features are chosen using IG before being classified using SVM classifier.

3. Proposed Method for Metamorphic Malware Detection

Earlier studies on metamorphic detection [3–5, 14] mostly employed opcode-based feature extraction, which is applicable only in host-based detection because of the disassembly requirement. The proposed method overcomes it by extracting the features directly from binaries. Those features are masked with sub-signature n-grams for classification. Similar to previous methods [10, 13], the term frequency of features are computed and the ML classifier is used to classify unknown files based on the term frequency of these features.

Figure 1 displays a screen-shot of one original metamorphic and its mutated version. The left pane shows all similar and dissimilar metamorphic codes of these two files. Some instructions remain unaltered during the metamorphic malware mutation when generating new metamorphic file.

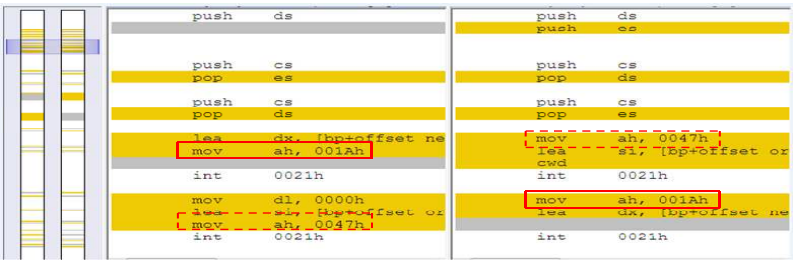

Figure 1. Comparison of two NGVCK viruses.

It is important to note that the mutation process generates metamorphic malware with inherited code segments from their ancestor. Some features in old metamorphic malware are kept unchanged in the mutated malware, as malware writers reuse old code segments [1]. Complete mutation of a metamorphic malware is deemed impossible due to the need to keep the same functionality [14]. Most versions of the same malware share a combination of several unchanged code segments [16]. Based on this hypothesis, an n-gram analysis for metamorphic malware detection can be made by mining these inherited code segments. Figure 2 shows different processing steps of the proposed method. n-gram sub-signatures are augmented with sub-signature obtain from existing metamorphic file.

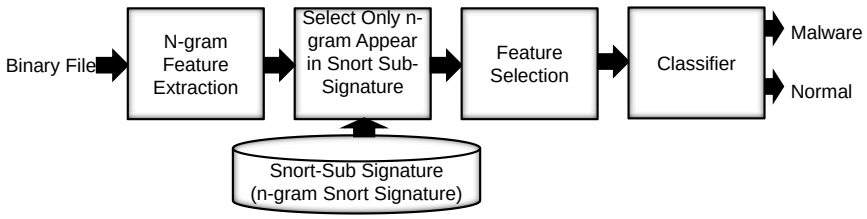

Figure 2. Various stages of metamorphic malware detection via the proposed method.

1. References [17, 18] used the method of splitting the signature to equal pieces. Present study uses overlapping 4-gram feature extraction together from Snort signature for malware detection. These signatures are used as a case study although Bro [7] rules can also be used. Snort signatures are split into overlapping 4-grams sub-signature. Then, term frequency (TF) of each unique n-gram in every file is counted and normalized, as term or byte frequency analysis is effective in files classification [19].

2. In order to minimize the large search space of n-gram, feature selection method is used. To select the most informative sub-signature n-gram features that appear in a file, the information gain (IG) feature selection method is employed [11].

$$IG(j) = \sum_{v_j \epsilon \{0.1\} C_i} \sum P(v_j, C_i) log \frac{P(v_j, C_i)}{P(v_j) P(C_i)} \quad (1)$$

where v_j is the value of the j-th attribute, C_i is the i-th class, $P(C_i)$ is the probability that the training data is in class C_i, $P(v_j)$ is the probability that the j-th *n*-gram have v_j value in the training dataset, and $P(v_j, C_i)$ is the probability that in class C_i, the j-th attribute has the value v_j.

3. Classification process requires generation of classifier model from the training dataset. The learning algorithm trains the SVM classifier to predict if a new file is malware or non-malware. Moreover, the SVM kernal methods can map the features into higher dimensional space by converting nonlinear features into linear ones [14]. Compared to other ML techniques, SVM has been reported to provide the highest malware attack detection accuracy [20]. The success of SVMs is due to the use of statistical learning theory [21], which is characterized by low estimation probability of generalization errors. There are several SVM kernel functions that can be used in our proposed technique. These include Polynomial kernel (PLY), Linear (LN), Sigmoidal, and Gaussian Radial Basis Function (RB) as summarized in Table 1 [22].

Table 1. SVM Kernel functions.

Function	Equation
Polynomial kernel	$k(x_i, x_j) = (\gamma x_i^T x_j + \Upsilon)^d, \gamma > 0$
Linear	$k(x_i, x_j) = \mathsf{x}_i^T x_j$
Sigmoidal Function	$k((x_i, x_j) = tanh(\gamma x_i^T x_j + \Upsilon)$
Gaussian RB Function	$k((x_i, x_j) = exp(-\gamma \parallel x_i - x_j \parallel^2), \gamma > 0$

where x_i and x_j are the training vectors, d is natural number,Υ is a shifting parameter that control the threshold, andγ is a scaling parameter. Variables d,Υ, γ are the kernel parameters. Further discussion on each kernel function can be found in [22].

It is reported [23] that SVM is widely used in IDS because it offers high-speed classification, and delivers scalability. Moreover, it is comparatively insensitive to the quantity of information points with low generalization error. As aforementioned, the proposed method can be used also for metamorphic malware detection in both host and network-based detection system (directly from traffic flows). Since NIDS also used signature maching to detect attack and malware, the present study adopt sub-signature features similar to Varghese et al. [17]. The proposed method uses only the n-gram features that appear in known malware sub-signature to reduce the feature search space.

4. Experimental Setup

Experiments were performed on Intel ® core™ i7-4710 HQ at 2.50 GHZ (8GB RAM) on Linux Ubuntu 14.04 platform. All executable files (metamorphic malware and normal) were first converted to overlapping 4-grams features. Only the 4-gram metamorphic features that appeared in Snort features (sub-signatures) were selected. Snort signatures were extracted from Snort rules version 2.9.5.5. The most important features are chosen using the IG feature selection method, using Wakaito Environment for Knowledge Acquisition (WEKA) [24]. The important features with high rank are used to train and built the classifier model. Next, the model has used to classify the testing dataset. The SVM classifier was built using Library for Support Vector Machines (LIBSVM) (https://www.csie.ntu.edu.tw/~cjlin/libsvm/), to simulate online metamorphic detection after extracting informative 4-gram features directly from executable binary files. Grid Search is used in training to select optimum effective SVM parameters [25].

The training data set contained labeled metamorphic and non-malware files. Hexdump utility tool is used to convert the content of binary executable to hexadecimal code, where n-grams are extracted by combining each 4-byte sequence as one feature [11]. Following [11, 18], the size of n-gram is selected to be 4-grams. Features are represented in terms of normalized frequency instead of representing each of them as Boolean attribute.

4.1. Dataset Used

The training datasets consisted of normal and metamorphic files. The normal (benign) executable files of total 1,971 were collected from Window 7, Window XP, and Cygwin [26]. A total of 1020 metamorphic files were collected: 109 metamorphic files are collected in assembly format from [27], where Turbo Assembler is used to convert metamorphic files from assembly to executable format. . 50 files were generated by the Next Generation Virus Construction Kit (NGVCK), 50 files were used from Second Generation (G2) viruses, and the other 9 files were employed as Mass Produced Code Generation Kit (MPCGEN) viruses. Furthermore, the NGVCK kit was also used to generate another 1000 metamorphic files under identical configuration setting. This construction kit is available in VX Heavens [28]. It can generate strong metamorphic variants with various obfuscation techniques [1,9]. 911 files were compiled using Turbo Assembler (TASM) version 5 under Oracle VM VirtualBox version 4.2.16. .

4.2. Dataset Preprocessing

n-grams features were extracted from the training dataset executable files, which include thousands of features, and most do not contribute to classification. Therefore, Snort malware rules are used to extract Snort 4-grams sub-signatures (called Snort features). The primary aim being the detection of metamorphic malware from its payload only the signatures content of malware are extracted to generate 4-grams.

5. Results and Discussion

In total, 1542 Snort 4-gram features appeared in metamorphic files. For a balanced training data set, 50% of the metamorphic files are selected in training and building the model (510 metamorphic and 510 normal files). The rest of the files are used as testing dataset (510 metamorphic and 1461 normal files).

5.1. Accuracy

Figure 3 shows the accuracy of the testing data set after building the classifier from the training data set. It uses different number of features selected by IG and different kernel functions: Gaussian Radial Basis Function (RB), Linear (LN), and Polynomial Kernel (PLY). The number of features affected the accuracy, area under the curve (AUC), false negative rate (FNR), false positive rate (FPR), true negative rate (TNR), and true positive rate (TPR) as shown in Figure 3. It is clear that for the number of features greater than 500, the accuracy of the testing data set appeared stable in terms of AUC, FPR, FNR, TPR, and TNR. The best SVM kernel function is also analyzed. Radial Base produced the best accuracy, FPR, FNR, TPR, and TNR compared to other tested kernals.

For further comparison with the work of Alam et al. [1], 359 new normal files with their sizes range from 1 to 10 MB were added. Thus, the dataset contained 2330 normal files and 1020 metamorphic files with the total size of 3.9 GB, which are classified using 5-fold cross validation in 4.68 sec. Experiments were repeated with different training and testing dataset ratio, as shown in Figure 4, where the values of accuracy, FPR, FNR, TPR, and TNR are illustrated. The accuracy of the classifier is shown to be high when the percentage of metamorphic files in the training dataset ranges between 10% and 50%. Furthermore, the accuracy is found to decrease when the amount of metamorphic files in the training dataset became greater than 50%. This reduction in accuracy may be attributed to the imbalance problem due to the less number of normal files than metamorphic one.

In the rest of the experiments, the classifier was trained with 10% metamorphic files (102 files) and 90% normal files (918). These ratios of malware and normal files were chosen to be similar to the real-life situation with the number of malware is less than 10% [13]. The testing data set contained the remaining 90% of the metamorphic files (918 files) and 1412 normal files. The accuracy is found to be 99.7% when RB kernal function has been used.

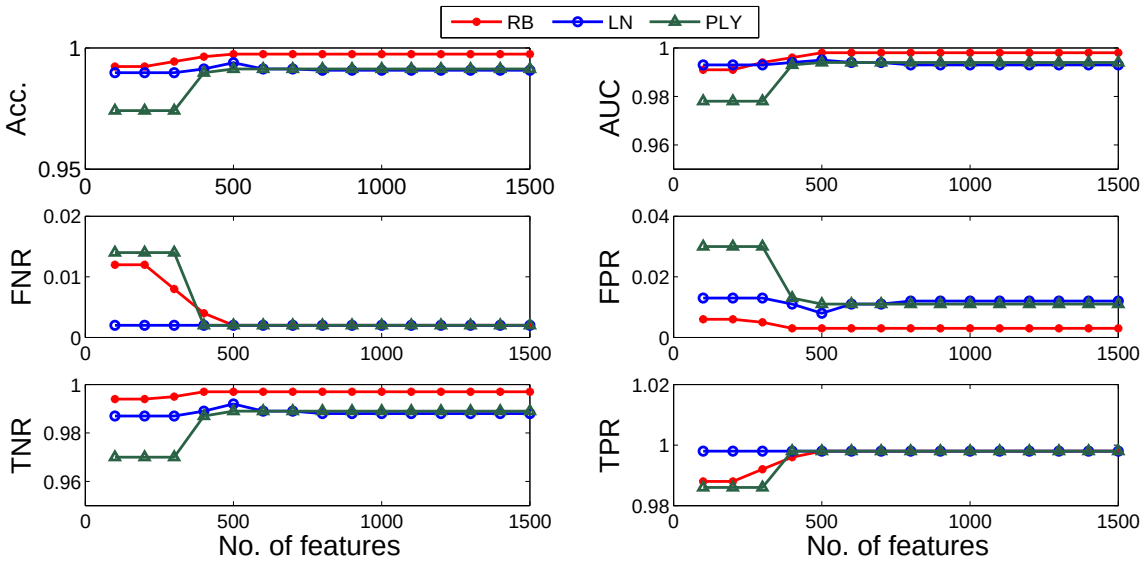

Figure 3. Testing dataset results for different number of features and different Kernel functions.

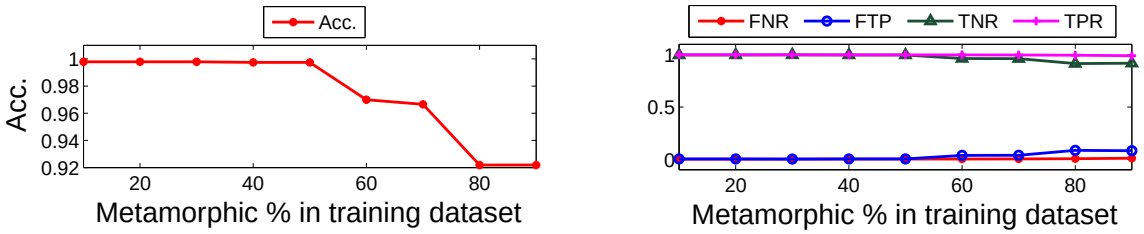

Figure 4. Accuracy, FPR, TPR, FNR, TNR of the testing dataset.

5.2. Comparison with Related Works

The performance of the proposed metamorphic malware detection method was compared with the results of Alam et al. [1] using the same kit-generated the metamorphic files [28]. In the experiment, NGVCK was used to generate highly metamorphic viruses [29]. Identical size data set (1020 metamorphic files and 2,330 normal files that were used [1] through 5-fold cross validation. The results were compared with several results in literature as summarized in Table 2. The proposed technique revealed very high detection rate. The results presented in [4, 15, 30] showed better accuracy compared to the present one and [1] possibly because of limited dataset size.

Table 2 compares the performance of our proposed method with some existing works. The high DR (TPR) percentage achieved by the proposed method is a clear indication of major metamorphic malware detection such as NGVCK, which is not possible to detect using the other methods [9]. Furthermore, the use of small data set size in other methods [4, 15, 30] makes the detection rates somewhat higher. The occurrence of extremely low FPR of our method may be ascribed to the feature selection by Snort sub-signatures which can select informative metamorphic features and effective implementation of radial-based SVM.

5.3. Comparison with Anti-Viruses Scanners

The effectiveness of the proposed technique was evaluated by comparing the results against commercial anti-viruses to detect metamorphic files. Other anti-virus tools were tested with identical dataset of metamorphic files. The detection rates are shown in Table 3. The best

Table 2. Comparison with related works.

System	Analysis	DR	FPR	Malware/Normal	Platform
Proposed method	Static	99.6%	0.3%	1020/2330	Win&Linux64
Opcode-Histogram [4]	Static	100%	0%	60/40	Win&Linux32
Opcode- histogram [14]	Static	99.5%	1.3%	1090/921	Win32
SWOD-CFWeight [1]	Static	94.69%	10.59%	1020/2330	Win&Linux64
Opcode-Graph [15]	Static	100%	1%	200/41	Win&Linux32
Opcode-SD [5]	Static	~ 98%	~ 0.5%	800/40	Linux32
Chi-Squared [29]	Static	~ 98%	~ 2%	200/40	Win&Linux32
Opcode-HMM [30]	Static	100%	0%	200/40	Win&Linux32
Opcode-PHMM [9]	Static	~100%	-	240/70	Win32

detection rate is obtained with Kaspersky that is able to detect MPCGEN correctly and several G2 files. However, it could not detect any NGVCK viruses that were generated by the kit [28]. Thus, it is affirmed that our method can detect complex metamorphic malware types that remain unrecognized by commercial anti-viruses. This high level of detection capability is ascribed to the combined effects of term frequency features extraction and their masking with known malware sub-signature for- accurate metamorphic classification.

Table 3. Comparison with commercial anti-viruses.

Anti-virus	Detection Rate
Proposed method	99.69%
Kaspersky	30.68%
AVG	17.35%
Comodo	14.80%
Avast	6.96%
Avira	6.17%

6. Conclusion

We proposed an effective technique to detect metamorphic malware with high accuracy and low false positive rate by combining n-gram Snort signatures and SVM. Metamorphic malware executable detection is achieved using only 500 features of n-gram Snort-masked sub-signatures. The proposed method exhibits its superiority with higher accuracy and lower false positive over various AVs.- It improves the metamorphic detection accuracy remarkably, but at the same time the proposed method can not detect several metamorphic files if they mutated several times using substituting instructions with equivalent instructions. Since the detection of some malware types in network is very arduous [31], the present method can be extended in NIDS to detecting metamorphic malware in network-based with real traffic traces especially when implemented as hardware system such as using field-programmable gate array (FPGA) .

Acknowledgment

The first author would like to thanks the Ministry of Higher Education and Scientific Research, Iraq for providing Doctoral scholarship for her study.

References

[1] S. Alam, R. N. Horspool, I. Traore, and I. Sogukpinar, “A framework for metamorphic malware analysis and real-time detection,” *Computers & Security*, vol. 48, pp. 212–233, 2015.

[2] A. A. E. Elhadi, M. A. Maarof, B. I. Barry, and H. Hamza, "Enhancing the detection of metamorphic malware using call graphs," *Computers & Security*, vol. 46, pp. 62–78, 2014.
[3] G. Canfora, A. N. Iannaccone, and C. A. Visaggio, "Static analysis for the detection of metamorphic computer viruses using repeated-instructions counting heuristics," *Journal of Computer Virology and Hacking Techniques*, pp. 1–17, 2013.
[4] B. B. Rad, M. Masrom, and S. Ibrahim, "Opcodes histogram for classifying metamorphic portable executables malware," in *International Conference on e-Learning and e-Technologies in Education (ICEEE)*. IEEE, 2012, pp. 209–213.
[5] G. Shanmugam, R. M. Low, and M. Stamp, "Simple substitution distance and metamorphic detection," *Journal of Computer Virology and Hacking Techniques*, vol. 9, no. 3, pp. 159–170, 2013.
[6] "Snort." [Online]. Available: https://www.snort.org/
[7] "Bro." [Online]. Available: https://www.bro.org/
[8] P. Li, M. Salour, and X. Su, "A survey of internet worm detection and containment," *Communications Surveys & Tutorials, IEEE*, vol. 10, no. 1, pp. 20–35, 2008.
[9] S. Attaluri, S. McGhee, and M. Stamp, "Profile hidden markov models and metamorphic virus detection," *Journal in computer virology*, vol. 5, no. 2, pp. 151–169, 2009.
[10] I. Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas, "Opcode sequences as representation of executables for data-mining-based unknown malware detection," *Information Sciences*, vol. 231, pp. 64–82, 2013.
[11] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," *The Journal of Machine Learning Research*, vol. 7, pp. 2721–2744, 2006.
[12] M. G. Schultz, E. Eskin, E. Zadok, and S. J. Stolfo, "Data mining methods for detection of new malicious executables," in *Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on*. IEEE, 2001, pp. 38–49.
[13] 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, no. 1, pp. 1–22, 2012.
[14] D. K. Mahawer and A. Nagaraju, "Metamorphic malware detection using base malware identification approach," *Security and Communication Networks*, vol. 7, no. 11, pp. 1719–1733, 2014.
[15] N. Runwal, R. M. Low, and M. Stamp, "Opcode graph similarity and metamorphic detection," *Journal in Computer Virology*, vol. 8, no. 1-2, pp. 37–52, 2012.
[16] A. H. Sung and S. Mukkamala, "The feature selection and intrusion detection problems," in *Advances in Computer Science-ASIAN 2004. Higher-Level Decision Making*. Springer, 2005, pp. 468–482.
[17] G. Varghese, J. A. Fingerhut, and F. Bonomi, "Detecting evasion attacks at high speeds without reassembly," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 327–338, 2006.
[18] I. Ismail, M. N. Marsono, B. M. Khammas, and S. M. Nor, "Incorporating known malware signatures to classify new malware variants in network traffic," *International Journal of Network Management*, vol. 25, no. 6, pp. 471–489, 2015.
[19] J. Clemens, "Automatic classification of object code using machine learning," *Digital Investigation*, vol. 14, pp. S156–S162, 2015.
[20] P. Wang and Y.-S. Wang, "Malware behavioural detection and vaccine development by using a support vector model classifier," *Journal of Computer and System Sciences*, vol. 81, no. 6, pp. 1012–1026, 2015.
[21] V. N. Vapnik and V. Vapnik, *Statistical learning theory*. Wiley New York, 1998, vol. 1.
[22] C.-W. Hsu, C.-C. Chang, C.-J. Lin *et al.*, "A practical guide to support vector classification," 2003.
[23] Y. Chen, Y. Li, X.-Q. Cheng, and L. Guo, "Survey and taxonomy of feature selection algorithms in intrusion detection system," in *Information Security and Cryptology*. Springer, 2006, pp. 153–167.

[24] "Weka." [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/downloading.html
[25] M. Shankarpani, K. Kancherla, R. Movva, and S. Mukkamala, *Computational Intelligent Techniques and Similarity Measures for Malware Classification*. Springer, 2012, pp. 215–236.
[26] "Cygwin." [Online]. Available: http://cygwin.com/
[27] D. Sayali, P. Younghee, and S. Mark, "Eigenvalue analysis for metamorphic detection," *springer*, 2013.
[28] "Ngvck." [Online]. Available: http://vxheaven.org/vx.php?id=tn02
[29] A. H. Toderici and M. Stamp, "Chi-squared distance and metamorphic virus detection," *Journal of Computer Virology and Hacking Techniques*, vol. 9, no. 1, pp. 1–14, 2013.
[30] W. Wong and M. Stamp, "Hunting for metamorphic engines," *Journal in Computer Virology*, vol. 2, no. 3, pp. 211–229, 2006.
[31] T.-F. Yen and M. K. Reiter, "Traffic aggregation for malware detection," in *Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 2008, pp. 207–227.