Static Feature Selection for IoT Malware Detection

Nguyen Ngoc Toan, Luong The Dung, Dang Quang Thang

***Abstract*—Our world has recently witnessed the explosive growth of IoT networks as one of the pillars of the 4th industrial revolution. Malware on IoT devices also grows accordingly in number and sophisticated techniques. Therefore, it is necessary to come up with more efficient approaches to IoT malware detection with machine learning models that can be used in solutions using limited resources. In this paper, we study and evaluate the efficiency of using a weight of term frequency–inverse document frequency model in feature selection method combined with an effective machine learning model in IoT malware detection based on opcode sequence features. We performed experiments on a MIPS ELF dataset that included 4,511 malicious samples with main four classes and 4,393 benign programs. Experiment results show that our proposed method has very good performance on the above dataset with detection and classification accuracy which are 99.8% and 95.8% respectively while the models only use 20 opcodes that have the highest weight values.**

***Tóm tắt*— Cuộc cách mạng công nghiệp lần thứ 4 với sự phát triển của các thiết bị IoT đã và đang ảnh hưởng sâu rộng đến các lĩnh vực trong đời sống xã hội. Các mã độc trên thiết bị IoT ngày càng gia tăng về số lượng và sử dụng các kỹ thuật lẩn tránh tinh vi. Điều này đòi hỏi cần có các phương pháp tiếp cận hiệu quả hơn trong phát hiện mã độc trên thiết bị IoT với các mô hình học máy hiệu quả, có khả năng ứng dụng trong các giải pháp đảm bảo an toàn thông tin có tài nguyên hạn chế. Trong bài báo này, chúng tôi nghiên cứu và đánh giá hiệu quả của việc xác định trọng số trong tìm kiếm truy xuất thông tin trong phương pháp trích chọn đặc trưng kết hợp mô hình học máy hiệu quả cho việc phát hiện mã độc IoT dựa trên đặc trưng chuỗi opcode. Chúng tôi đã tiến hành thử nghiệm với một tập dữ liệu MIPS ELF gồm 4.511 mẫu độc hại với 4 loại chính và 4.393 chương trình lành tính. Các kết quả thực nghiệm đã chứng minh rằng phương pháp của bài báo đề xuất cho kết quả tốt đối với tập dữ liệu nêu trên, tỉ lệ phát hiện và phân 4 loại mã độc cao nhất tương ứng là 99.8% và 95.8% khi chỉ cần sử dụng 20 opcode có giá trị trọng số cao nhất.**

Keywords—feature selection; opcode; IoT malware; malware detection; machine learning.

Từ khóa—trích chọn đặc trưng; opcode; mã độc IoT; phát hiện mã độc; học máy.

I. INTRODUCTION

The quantity of IoT devices in information systems has increased rapidly from over 16 billion in 2015 to 30 billion in 2020 [2] and 30.9 billion devices are installed by 2025 [3]. Common cyber-attacks involve malware, APT, and ransomware, which control and destroy systems. According to Gibert et al. [1], malware metamorphosis increased over 50% on mobile devices in 2017. Besides, IoT attacks increased by nearly 600%, which the Mirai malware and its variants created some of the most powerful distributed denial-of-service attacks in the past. Therefore, security for IoT devices against attacks and malware is a crucial requirement today.

As usual, the fight against malicious software will begin with knowledge about the signatures of malicious activities. The multiped use of the Internet has resulted in organizations constructing more advanced technologies and criterion security solutions to resist attacking IoT devices from hackers. There are mainly two methods that are available for malware detection include behavioral-based and signature-based detection. Malware detection based on signature is a very effective method in detecting known malware thanks to its high accuracy and clearness in building detection systems. However, the method has disadvantages in detecting unknown malware, polymorphic and metamorphic malware. On the other hand, in malware detection based on behavior, an abnormal is defined as monitoring a dataset that takes shape to be incompatible with the residue of that dataset [4]. Malicious behavior is explored as a divergence from the regular behavior set of the users in the information system. The advantage of behavioral-based detection is in unknown intrusion detections. Anomaly-based detection methods can detect

unknown malware, but modeling normal behavior is complex work. The normal behavior is modeled based on machine learning techniques. Therefore, anomaly-based detection methods can be used in IoT malware detection and previously underserved architectural platforms such as Embedded Linux OS.

Most of the researchers in the information security community work on the techniques used to identify and detect Windows malware samples among others, particularly, Intel's x86-64 architecture. However, the MIPS processor architecture is used in popular IoT devices such as switches, routers, access points, and IP cameras [3]. It is a fact that when an application runs on different processor architectures and operating systems, its behaviors are dissimilar. Therefore, it is necessary to study malware detection on IoT devices that are used the Embedded Linux OS and MIPS processor. In this direction, researchers had many promising results on malware detection based on machine learning using static or dynamic features. Dynamic features include memory usage [30], instruction traces [9], network traffic [11], API call trace [10], [12]. The effectiveness of dynamic analysis is highly dependent on the malware execution environment. With static features, common forms have been used include strings [13], bytes n-gram [14], opcode [15], function call graph [16], entropy-based [17], etc. In this paper, we first focus on the IoT malware detection method using the opcode sequences. Quality features are crucial for building effective machine learning-based classification models. The opcode sequence is used effectively in malware detection, but if there are too many features leads to the model complexity and the "curse of dimensionality" problem will occur [8]. In addition to that, data normally contains significant noises and irrelevant features that add little or no value to the performance of the learning algorithms. Therefore, we propose an effective opcode features selection method that can overcome those problems above. Following are 3 main contributions of our work:

- We propose a top MIPS opcode feature list that can be used well for IoT malware detection and classification.
- We formulate a framework for IoT malware detection based on static features and evaluate the malware detection and classify possibilities of the feature selection techniques using machine learning models.
- We demonstrate the usefulness of feature selection in IoT malware classification systems.

The rest of the paper is structured as follows. Related works malware detection based on static features are discussed in section 2. Our feature selection method is introduced in section 3. Section 4 highlights the framework used in this paper with experiments and evaluations. Finally, the conclusion and future works are discussed.

II. RELATED WORKS

A. MALWARE ANALYSIS BASED ON STATIC FEATURES

Malware analysis is a process of determining the malicious behavior of a program. Malware analysis is often based on static and dynamic features [3]. Dynamic behavior-based malware detection methods usually need a secure and controlled environment, such as virtual machines, sandbox, etc. In IoT malware analysis, dynamic features-based methods have significant limitations such as time consuming, considerable attention requirement, and analysis environment dependency. Meanwhile, the static analysis is unaffected by the analytical environments because static features are extracted by analyzing the software source code or binary code from the perspectives of syntax and semantics without running the software.

A lot of research has been done and many ways have been brought forward to detect malware based on machine learning using static features. Specific features for training the machine learning model include extracting operational codes (opcode) sequences after disassembling the binary file, or extracting the control flow graph (CFG) from the assembly file, extracting API calls from the binary, etc.

J. Z. Kolter and M. A. Maloof [5] proposed using n-gram instead of a byte sequence and compared the performance of Naive Bayes, decision trees,

support vector machines for malware detection. Later, artificial neural network [6] [7] were also used for malware detection. However, it takes more processing time with a large amount of data.

Ding at el. [24] proposed static characteristic extraction method, called Control flow-based features (CFG), has the ability to detect malware with high effectiveness. However, the method seems to work efficiently in case of a small file size with the average number of vertices CFG less than 6,000 peaks.

Bilar [2] analyzed executable files and demonstrated a difference in the opcode frequency distributions of benign files and malware. This research shows that rare opcodes seem to be a stronger predictor, explaining 12–63% of frequency variation. However, the authors only focus on the frequency of opcodes and the sample size for their experiments was only 97.

B. OPCODE FEATURE SELECTION

Operation code (Opcode) is one of the common features used in malware detection. An opcode is a single instruction that can be executed by a microprocessor (CPU), which can describe the behavior of an executable file.

The performance of any detection, classification, and prediction system is closely dependent on features. Today, feature selection approaches are organized into filters, wrapper, embedded and hybrid methods. The filter methods select a subset of features without altering their original representation [20]. This method is used in many machine learning models because it is not constrained by any machine learning method. Feature selection based on wrapper involves a classification model to assess the suitability of the features.

Canedo et al. [19] analyzed the seven feature selection methods based on filter, two methods based on the wrapper, and two embedded feature selection methods to creatmicroarray data sets under four machine learning classifiers. Xue et al. [21] compared the wrapper-based feature selection and filter-based feature subset selection methods with respect to classification performance and execution time. These researches show that the filter methods are faster than the wrapper methods but have much lower classification performance than the wrapper methods.

C. MACHINE LEARNING METHODS FOR MALWARE DETECTION BASED ON OPCODE FEATURES

A faster classification method is needed to recompense the exponentially increasing number of malware adaptations. A machine learning approach for malware detection should be adopted for faster and more efficient classification. Machine learning models for malware detection and classification can be trained on opcode features that are extracted from executable programs.

The use of opcode feature has brought a lot of efficiency in detecting malware in general and IoT malware in particular. Santos et al. [25] extracted opcodes by decompiling the binary file and converting them into a sequence of length n subsequences. The authors use the frequency of the opcode and category of the source binary to assign a weight to each opcode. The machine learning models include K-nearest neighbors (KNN), Decision Tree (DT), Support Vector Machine (SVM), Naïve Bayesian classifier, and Bayesian network were then used to detect and compare performances according to numerous sizes of n. Support Vector Machines (SVM) is a problem to find the interface so that the closest distance from a data point to the interface (margin) is found to be the largest, which means that the data points are "safest" compared to other data points. separation face. The SVM classifier method is chosen to be used in the text classification system, malware classification problem and promotes its ability with the n-gram feature extraction method. A decision tree (DT) is a structured hierarchical tree used to classify objects based on a series of rules. The attributes of the object (except the categorical attributes) can be of different data types including binary, nominal, ordinal, quantitative values, while the categorical attribute must have a data type of binary or ordinal. Given data about objects consisting of attributes and their classes, the decision tree will generate rules to predict the class of the unseen data. Random Forest (RF) is a member of the family of decision tree algorithms. Random Forest treats each decision tree as an independent voter (like a real election). At the end

of the election, the answer with the most votes from the decision trees will be selected. To make sure that not all decision trees give the same answer, Random Forest deletes some observations and repeats others randomly. Naive Bayes (NB) is based on a probability calculation, which has given good results in detecting malicious code [22]. Naive Bayes Classification is an algorithm based on Bayes theorem of probability theory to make judgments as well as classify data based on observed and statistical data. Naive Bayes Classification is one of the algorithms used to make the most accurate predictions based on a collected dataset.

Yewale et al. [26] selected the 20 most frequently used opcodes from a set of benign programs and detected malicious programs using DT, SVM models. However, only a small-scale dataset was used for the model training and testing, therefore the same performance on a larger dataset is not expected. Jerome et al. [27] used opcode sequences with machine learning and experimented with 2, 3, 4 and 5-gram opcode features. Feature ranking and selection were done based on computing the information gain of each n-gram of opcode sequences. BooJoong Kang [28] presented and evaluated to identify and categorize Android malware based on n-gram opcode features and machine learning. The method allows for automatic extraction and learning features from given datasets. Good classification accuracy can be achieved when by using frequency opcodes with a small n.

III. PROPOSED METHOD

As shown in Figure1, the general automated classification framework for IoT malware detection consists of four phases: (1) the pre-processing (includes feature extraction) phase, (2) a phase for selecting feature opcode, (3) detection phase, and (4) classification phase.

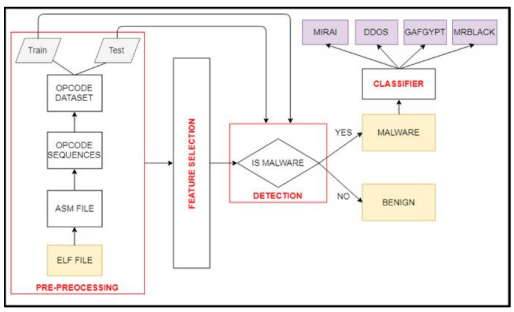

Figure1. Framework for IoT malware detection based on the static feature

A. *PRE-PROCESSING PHASE*

The opcode sequences collection process starts with using the IDA pro tool [29] to decompile ELF file samples. Then, the assembly files obtained after the decompile process are processed with a python script to get the opcode sequences. After extracting opcode sequences of the samples, the opcode sequences of the samples that are packed or too short are removed. As a result of this process, opcode datasets are collected.

B. FEATURE SELECTION

Both the number and quality of the features are considered to train ELF file classification models with respectable accuracy and fewer resources. Therefore, feature selection is used to remove irrelevant, constant, redundant, and correlated features from the raw features dataset before training the models. A variety of methods for selecting the best features in malware detection research have been widely deployed. In our approach, one opcode is treated as a word in the language model. An opcode sequence is taken as a sentence in the language model and as such we can predict the meaning of a sentence based on some keywords in the sentence. Therefore, in an opcode sequence, each opcode has a different level of meaning in the sentence, some opcodes can represent that opcode sequences.

Generally, the opcode feature selection method follows a typical scenario described in Fig.2. Our paper estimates the significance of an opcode based on the weight of Term Frequency-Inverse Document Frequency (TF-IDF) model. TF - IDF is the weight of a word in a document obtained through statistics showing the importance of this word in a document, which itself is in a set of

documents. Term frequency (TF) is used to estimate the frequency of occurrence of opcodes in the opcode sequences. Each sequence length is different, the number of occurrences of the opcodes can vary greatly. So, the number of occurrences of the opcodes will be divided by the length of the sequences (the same as the total number of words in a text). The term frequency of opcode x is calculated as follows:

$$TF(x, s) = \frac{fr(x)}{sum(s)} \quad (1)$$

where *fr(x)* is the number of times x occurs in opcode sequence *s*, *sum(s)* is the total number of opcodes of sequence set *s*.

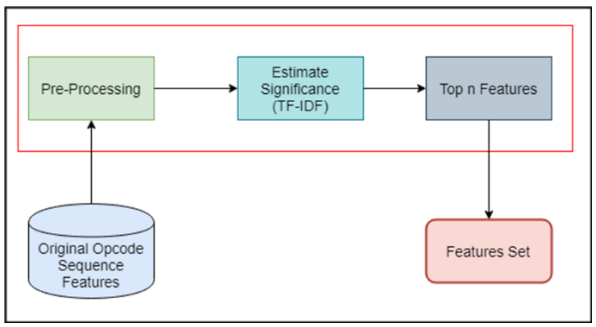

Figure 2. Paradigm of feature selection method

Inverse Document Frequency (IDF) is an estimate of the influence of opcodes. When only the frequency of occurrence of the opcode is calculated, the opcodes are considered equally important. However, there are some opcodes that are often used but are not important to express the meaning of the opcode sequence. Therefore, the IDF is capable of redefining the corresponding weights for major opcodes that always appear. The Inverse Document Frequency is described as:

$$IDF(x, D) = log\ (\frac{N}{D(x)}) \quad (2)$$

where *N* is the total number of opcode sequences set, *D(x)* is the number of opcode sequences containing opcode *x*.

Determining the importance of opcodes in sequences has much in common with relevance of words to documents. The researchers had many promising results on term frequency-inverse document frequency of words to documents such as [32], [33]. Therefore, we propose to use the measure Term Frequency-Inverse Document Frequency (TF-IDF) of opcode x is determined as:

$$TF\text{-}IDF\ (x) = TF(x).IDF(x,D) \quad (3)$$

By determining the TF-IDF of the opcodes in the opcode dataset, we use *n* highest weighted opcodes for the malware detection and classifier problem.

Besides, the n-gram method is used to calculate the quantities f(x) and D(x) in the formula of the TF-IDF and is used in feature extraction for training on machine learning models. N-gram is the frequency of occurrence of words in the corpus. In this paper, a sequence of opcodes is embedded into vector space using n-gram. Each element of a feature vector represents the presence or absence of the corresponding n-gram in the opcode sequence. The n-gram method has been proven effective in malware detection based on sequence [18], [22]. Kang et al. [23] presented and evaluated an n-gram opcode features-based approach that utilizes machine learning to identify and categorize Android malware with an f-measure of 98%. In the n-gram feature extraction method, if n is too small (unigrams), the information obtained will only be the frequency of occurrence of single system calls. If n is too large, the number of features is very large, especially malware that uses transformation techniques. There are some other studies also show that bigrams, trigrams can give good results.

C. Malware Detection and Classification Models

The ideal detection and classification models for evaluating the direct impact of the feature selection algorithms are models which not capable of embedding feature selection. The mentioned classification methods are not only popular machine learning models but also do not perform embedded feature selection. Therefore, they are suitable for evaluating the proposed feature selection method.

In this paper, we use experimental methods to choose the effective machine learning algorithm model based on unsupervised learning algorithms such as SVM, RF, NB.

IV. EXPERIMENTS AND EVALUATIONS

A. DATA COLLECTION

An IoT dataset used for testing includes 8,904 MIPS ELF samples including 4,511 malware and 4,393 benign samples. The malware dataset is collected from different sources on the Internet such as IoTPOT, VirusShare, VirusTotal, Detux, and available programs on Embedded Linux. In our experiments, the label of malware samples is taken under the Symantec label because it is a program with good malware detection and is explicitly named. There are 37 different families of malware with many popular families of malware such as Mirrai, LightAidra/ Aidra/ Zendran, Gafgyt/ BASHLITE/ Lizkebab/ Torlus, Dofloo/ Spike/ MrBlack/ Wrkatk/ Sotdas/ AES.DDoS/ DnsAmp, Moose, Hajime, Tsunami/Kaiten, Trojan.Gen, SecurityRisk, etc... We only select 4 families of malicious code with the number of samples in the set over 100 samples to avoid spreading out in number when classifying. The number of malware samples with labels is shown in Figure 3:

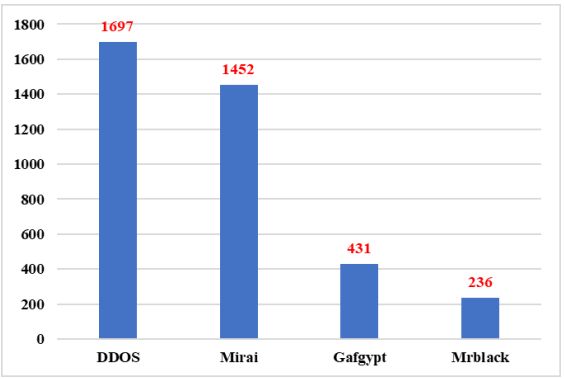

Figure 3. MIPS ELF malware labeled dataset

Then, assembly files are extracted by IDA Pro 6.6. After that, opcode sequences are generated by a python script. The opcode sequences of the samples that packed or are shorter than 50 will be removed. The opcode sequences dataset result collected are shown in Table I.

TABLE I. OPCODE SEQUENCE DATASET RESULTS

Label	Benign	Malware	Classifier			
			DDOS	***Mirai***	***Gafgypt***	***Mrblack***
The number of opcode sequences	4393	4423	1694	1442	416	206

B. FEATURE SELECTION

The feature selection methods discussed in section V are used to consider the top feature subsets. The top MIPS opcode features with the weights are shown in Table II.

TABLE II. TOP WEIGHTS OF MIPS OPCODE FEATURES

Rank	Opcode Feature	Weight	Describe
1	bgezal	8.29	Branch On >= 0 And Link
2	trap	7.09	Exception And Interrupt Instruction
3	sub	5.99	Subtract
4	addi	5.59	Add Immediate
5	mthi	4.89	Move To HI
6	bltzal	4.48	Branch On < 0 And Link
7	mtlo	4.31	Move to LO Register
8	add	3.89	Add
9	jal	2.63	Jump And Link
10	srav	2.18	Shift Right Arithmetic Variable
11	lh	1.91	Load Halfword
12	syscall	1.68	System Call
13	div	1.62	Divide
14	srlv	1.58	Shift Right Logical Variable
15	break	1.53	Breakpoint
16	multu	1.52	Unsigned Multiply
17	mult	1.49	Multiply
18	divu	1.47	Unsigned Divide
19	mflo	1.37	Move From LO Register
20	nor	1.35	Bitwise NOR (NOT-OR)

The top MIPS opcode features are combined with the n-gram method to feature selection before using machine learning models to perform classification stage.

C. EVALUATION METRIC

In our paper, several evaluation standards are used to calculate the effectiveness of the approach.

Accuracy can be described as:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (4)$$

where True Positive (TP) indicates that the number of malware samples identified correctly; False Positive (FP) is the number of benign samples truly predicted to be malware; True Negative (TN) is the number of trusted applications identified correctly; False Negative (FN) is the number of malware samples is taken as trusted programs.

The F1-score is the harmonic average of the recall and precision of one class.

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

In the above formula, False Positive (FP) is the number of trusted programs is detected as malware.

Recall is a fraction of system call sequence in ground truth that is correctly classified:

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

F1-Macro: Average of the F1-scores of classes, characterizing classifier performance on small classes.

F1-Weight: Weighted average of the F1 scores of classes, with weight proportional to their support in the ground truth.

D. DATA COLLECTION

The experiments analyze the influence of feature selection methods on the classifier models in terms of performance. Our experiments were run on the 64-bits Windows 10 operating system, with Intel Core i7-6500U, 2.59 GHz v, 8GB RAM. We consider the predictive performance of the opcode feature recommended by the feature selection methods discussed in section V. In order to evaluate performance, three machine learning models which use n-gram feature selection method are conducted on the same dataset as our method. In SVM, we have chosen the corresponding parameters including gamma='scale', decision_function_shape='ovr', cache_size=500, tol=2e-3, and break_ties='True'. Max_depth=200 and random_state=2 are used in RF method. The comparison results are shown in the tables below.

In the malware detection approach, when the number of the top opcode features is 20, the Random Forest model has the highest accuracy of 99.8 % and F1-Weight of 99.8%. If the number of opcode features is small, malware detection results are not satisfactory due to the lack of information to identify the characteristics of the malware and benign. Besides, if many opcode features are chosen, it has noisy features, which is not effective in classifying malicious code. In research [31], 14 opcodes most frequently in Intel are selected for detection models. In the experiment, we consider choosing the number of top opcode features (include 5,10,14,16,20,30,40) with three machine learning models based on n-gram, results as in Table III, and Table IV.

TABLE III. MALWARE DETECTION RESULTS FOR TOP OPCODE FEATURES BASED ON 2-GRAM METHOD

Number of opcodes	**RF**		**SVM**		**NB**	
	ACC	***F1-Weight***	***ACC***	***F1-Weight***	***ACC***	***F1-Weight***
5	50.6	36.1	50.3	35.4	49.8	33.7
10	71.9	69.7	60.9	54.3	52.8	40.0
14	98.7	98.7	98.0	98.0	56.6	46.6
16	99.0	99.0	98.6	98.6	90.7	90.7
20	**99.8**	**99.8**	98.7	98.7	94.3	94.3
30	99.7	99.7	**99.3**	**99.3**	**96.1**	**96.1**
40	99.7	99.7	99.2	99.2	96.0	96.0

TABLE IV. MALWARE DETECTION RESULTS FOR TOP OPCODE FEATURES BASED ON 3-GRAM METHOD

Number of opcodes	RF		SVM		NB	
	ACC	*F1-Weight*	*ACC*	*F1-Weight*	*ACC*	*F1-Weight*
5	50.6	34.4	50.2	34.2	50.3	33.9
10	70.9	68.4	58.5	49.9	53.0	40.3
14	98.7	98.7	98.0	98.0	58.1	49.2
16	99.7	99.7	98.0	98.0	95.1	95.0
20	**99.8**	**99.8**	98.4	98.4	97.0	97.0
30	99.7	99.7	**99.1**	**99.1**	**98.1**	**98.1**
40	99.7	99.7	99.0	99.0	98.0	98.0

These results can be illustrated by the fact that the unfiltered opcode features accommodate various anomalies such as noise that impact the performance of a machine learning algorithm.

Overall, the Random Forest gives the best results as compared to the other machine learning models with an accuracy of 99.8% for both 2-gram and 3-gram with selecting 20 opcodes. Besides, Figure 5 shows the execution time of the various models. We can observe from the figure that Naive Bayes (NB) is the fastest classifier. The RF is naturally robust for small number opcode but it consumes larger memory and computation time for training. In the same case, SVM needs more computation time for training because of the need for large memory. By filtering out extraneous opcode features from the feature set, the running time of the learning algorithms and space complexity can be significantly reduced and the space complexity and yields a more general classifier. Therefore, an optimal set of features is necessary to build efficient machine learning models.

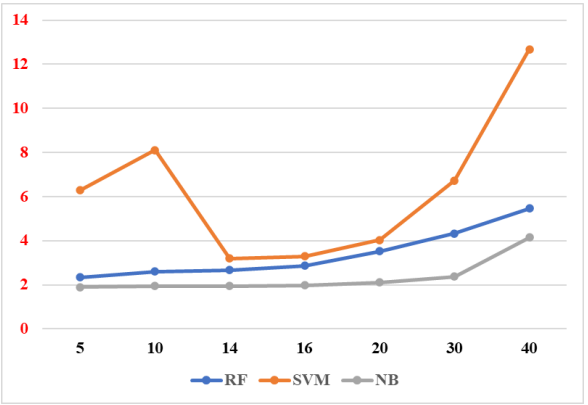

Figure 4. The run time for all machine learning models with 2-gram (second)

In the malware classifier approach with selecting 20 opcodes, the highest accuracy of 95.8 % and F1-Weight of 95.7% are achieved with random forest based on 4-gram. Malware classification results using 20 opcode features as shown in Figure 5.

Similar to the detection approach, Figure 6 shows that the Naive Bayes (NB) is also the fastest classifier model.

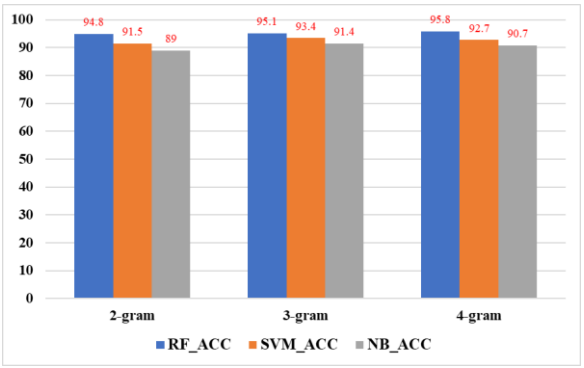

Figure 5. Accuracy of malware classification models using 20 opcode features (%)

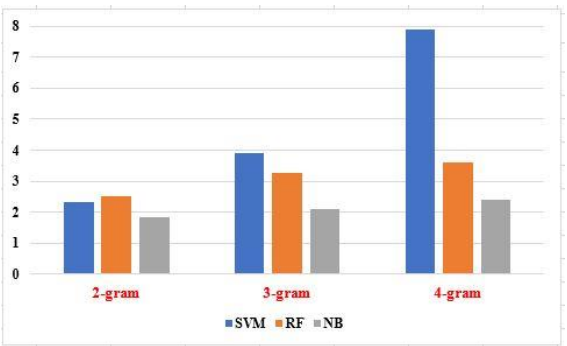

Figure 6. The run time for three machine learning models using 20 opcode features (second)

V. CONCLUSION AND FUTURE WORKS

In summary, the opcode sequence of programs has been employed and showed good performance in detecting IoT malware. In our proposed method, we estimate the significance of an opcode based on the weight of Term Frequency-Inverse Document Frequency to select the most effective opcodes in malware detection and classification. To evaluate the performance of our method, some experiments have been done and the results show that our method can achieve the highest accuracy of 99.8% for detection and 95.8% for classification approaches with only 20 opcodes.

In a static analysis in general, and opcode-based malware analysis in particular, feature extraction is still difficult when malware uses complex techniques such as code encryption, obfuscation, polymorphic, etc. In the future, other opcode sequence analysis methods can be extended to solve more complicated malware detection problems such as combining dynamic features and static features. Deep learning methods combined with more other features could also be considered to detect early detection and exact classification.

ACKNOWLEDGMENT

Toan Nguyen Ngoc was funded by Vingroup JSC and supported by the Master, PhD Scholarship Programme of Vingroup Innovation Foundation (VINIF), Institute of Big Data, code VINIF.2021.TS.128. And, this research is fractionally funded by Vietnam Ministry of Public Security, grant number SCN.2021.T01.05.

REFERENCES

[1] D. Gibert, C. Mateu, and J. Planes, "The rise of machine learning for detection and classification of malware: Research developments, trends and challenges", Journal of Network and Computer Applications, 153(January), 102526, 2020.

[2] S. Smith, IoT Connections To Reach 83 Billion By 2024 Driven By Maturing Industrial Use Cases, Apr. 2020. ([online] Available: https://www.juniperresearch.com/press/press-releases/iot-connections-to-reach-83-billion-by-2024-driven).

[3] Nguyen Ngoc Toan, Luong The Dung, Tran Nghi Phu, A novel approach to detect IoT malware by system calls and LSTM model, Journal of Theoretical and Applied Information Technology 31st August 2021 -- Vol. 99, 2021.

[4] Johnson, Richard Arnold, and Dean W. Wichern, Applied Multivariate Statistical Analysis, 5th ed. Prentice Hall, 2002.

[5] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," Journal of Machine Learning Research, vol. 7, no. Dec, 2006, pp. 2721–2744.

[6] G. E. Dahl, J. W. Stokes, L. Deng, and D. Yu, "Large-scale malware classification using random projections and neural networks," in Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013, pp. 3422–3426.

[7] J. Saxe and K. Berlin, "Deep neural network-based malware detection using two dimensional binary program features," in Malicious and Unwanted Software (MALWARE), 2015 10th International Conference on. IEEE, 2015, pp. 11–20.

[8] Abawajy, J.; Darem, A.; Alhashmi, A.A. Feature Subset Selection for Malware Detection in Smart IoT Platforms. Sensors 2021.

[9] D. Carlin, A. Cowan, P. O'Kane, S. Sezer, "The effects of traditional anti-virus labels on malware detection using dynamic runtime opcodes", IEEE Access 5, 2017, pp. 17742–17752,

[10] Yuxin Ding, Xuebing Yuan, Ke Tang, Xiao Xiao, Yibin Zhang, "A fast malware detection algorithm based on objective-oriented association mining", Computers & Security, Volume 39, Part B, Pages 315-324, ISSN 0167-4048, 2013,

[11] G. Zhao, K. Xu, L. Xu, B. Wu, "Detecting apt malware infections based on malicious dns and traffic analysis", IEEE Access 3, 2015, pp. 1132–1142.

[12] Z. Salehi, A. Sami, M. Ghiasi, "Maar: robust features to detect malicious activity based on api calls, their arguments and return values", Eng. Appl. Artif. Intell. 59, 2017, pp. 93–102.

[13] Y. Ye, D. Wang, T. Li, D. Ye, and Q. Jiang, "An intelligent pe-malware detection system based on association mining", J. Comput. Virol. 4 (4), 2008, pp. 323–334.

[14] Z. Fuyong, Z. Tiezhu, "Malware detection and classification based on n-grams attribute similarity",

in: 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), vol. 1, 2017, pp. 793–796.

[15] D. Yuxin, Z. Siyi, "Malware detection based on deep learning algorithm", Neural Comput, Appl. 31 (2), 2019, pp. 461–472.

[16] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel feature extraction, selection and fusion for effective malware family classification", CODASPY 16. In: Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy, ACM, New York, NY, USA, 2016, pp. 183–194.

[17] D. Gibert, C. Mateu, J. Planes, R. Vicens, "Classification of malware by using structural entropy on convolutional neural networks", In: IAAI Conference on Artificial Intelligence, 2018, pp. 7759–7764.

[18] Mas'ud, M.Z.; Sahib, S.; Abdollah, M.F.; Selamat, S.R.; Huoy, C.Y. A comparative study on feature selection method for N-gram mobile malware detection. Int. J. Netw. Secur, 2017,pp.727–733.

[19] Bolón-Canedo, V.; Sánchez-Maroño, N.; Alonso-Betanzos, A. A review of feature selection methods on synthetic data. Knowl. Inf. Syst,2013, pp.483–519.

[20] Abawajy, J.; Darem, A.; Alhashmi, A.A. Feature Subset Selection for Malware Detection in Smart IoT Platforms, 2021.

[21] Xue, B.; Zhang, M.; Browne, W.N. A comprehensive comparison on evolutionary feature selection approaches to classification. Int. J. Comput. Intell. Appl. 2015

[22] Tran Nghi Phu, Hoang Dang Kien, Ngo Quoc Dung, Nguyen Dai Tho, "A Novel Framework to Classify Malware in MIPS Architecture-Based IoT Devices", Hindawi Security and Communication Networks Volume 2019, Article ID 4073940, 13 pages, 2019.

[23] BooJoong Kang, Suleiman Y. Yerima, Sakir Sezer, Kieran McLaughlin, N-gram Opcode Analysis for Android Malware Detection, International Journal on Cyber Situational Awareness, Vol. 1, No. 1, 2016, pp.231-255.

[24] Yuxin Ding, Wei Dai, Shengli Yan and Yumei Zhang. Control Flow-Based Opcode Behavior Analysis for Malware Detection. Computers & Security 44, 2014, pp.65–74.

[25] Santos, I., Brezo, F., Ugarte-Pedrero, X., Bringas, P.G., Opcode sequences as representation of executables for data-mining-based unknown malware detection. Inf. Sci. (Ny), 2013.

[26] Yewale, A., Singh, M., 2017. Malware detection based on opcode frequency, in: Proc. 2016 Int. Conf. Adv. Commun. Control Comput. Technol. ICACCCT, 2016.

[27] Jerome, Q., Allix, K., State, R., & Engel, T. Using opcode sequences to detect malicious Android applications. In Proc. IEEE International Conference on Communications, 2014, pp.914–919.

[28] Kang, BooJoong & Yerima, Suleiman & Sezer, Sakir & Mclaughlin, Kieran. (2016). N-gram Opcode Analysis for Android Malware Detection. International Journal on Cyber Situational Awareness, 2016, pp.231-255.

[29] [Online] https://hex-rays.com/ida-pro/

[30] M. Ghiasi, A. Sami, Z. Salehi, "Dynamic vsa: a framework for malware detection based on register contents". Eng. Appl. Artif. Intell. 44, 2015, pp. 111–122.

[31] Daniel Bilar, Opcodes as predictor for malware, Int. J. Electronic Security and Digital Forensics, Vol. 1, No. 2, 2007.

[32] Qaiser, Shahzad & Ali, Ramsha. (2018). Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. International Journal of Computer Applications. 181. 10.5120/ijca2018917395.

[33] Kim, SW., Gil, JM. Research paper classification systems based on TF-IDF and LDA schemes. Hum. Cent. Comput. Inf. Sci. 9, 30 (2019).

ABOUT THE AUTHOR

Nguyen Ngoc Toan

Workplace: People's Security Academy

Email: ngoctoan.hvan@gmail.com

Education: Received Bachelor's degree in 2015 from PSA, Master's degree in 2019 and PhD student in Information Security from Academy of Cryptography Techniques.

Recent research direction: IoT malware detection, anomaly detection, machine learning for information security.

Luong The Dung

Workplace: Academy of Cryptography Techniques.

Email: thedungluong1@gmail.com

Education: Received Bachelor's degree in 2001, and PhD in 2011 in Mathematical foundation for computers and computing systems from Military Technology Academy.

Recent research direction: data privacy, cryptographic, data mining, machine learning for information security.

Dang Quang Thang

Workplace: People's Security Academy.

Email:dangquangthang0001@gmail.com

Education: People's Security Academy, Ministry of Public Security of Socialist Republic of Viet Nam.

Research direction: Malware detection, machine learning.