VOL. 23, NO. 2, 2024, 157-162
elektrika.utm.my
ISSN 0128-4428

ELEKTRIKA
Journal of Electrical Engineering

Obfuscated Computer Malware Classification Based on Significant Opcode

Yu Chii Heng and **Ismahani Ismail***

Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia.

*Corresponding author: ismahani@utm.my

Abstract: Computer malware has greatly impacted the computer network securities and even personal computer users. Signature-based detection is incapable to recognize the obfuscated computer malware since it is being covered by the obfuscation techniques. Therefore, machine learning is being explored and equipped in the malware detection to withstand the threaten of malware. In fact, there are many features available, i.e., text string to be implemented for malware classification. Nevertheless, opcode could be one of the features owing to its relative smaller data size compared to the text string. In this research, the significant opcodes of executable malware files which referring to the prevalent content from malware-to-malware generation are extracted as training dataset. Several machine learning classifiers are generated and compared in terms of classification accuracy and speed, as well as the comparison is done with text string-based detection and signature-based detection. From the finding, it is shown that machine learning detection performs more than 2 times better than signature based and machine learning generated based-on significant opcode features is able to detect obfuscated malware over 10 times faster than text string feature and still achieve up to 98% of accuracy.

Keywords: Computer Malware, Machine Learning, Obfuscated Malware, Opcode

Article History: received 23 April 2024; accepted 9 July 2024; published 29 August 2024

1. INTRODUCTION

Malicious software is the computer software that spreads automatically over the network from machine to machine to serves for malicious purpose. National Institute of Standards and Technology (NIST) reported that malwares are the most common external threat to most hosts which result in widespread damage, disruption and necessitating extensive recovery efforts within most organizations [1]. According to the report by Accenture, the estimated financial loss that caused by the malware attack is around $2.6 million [2].

Malware could be categories into many types i.e virus, worm, trojan house, rootkit, spyware, adware, botnet, keylogger, and ransomware. The malware has evolved so that they could hide or cover themselves from being detected by the antimalware software. The obfuscated computer malware is where the malware that able to change its binary code while preserving the malware functionality so that it would not be detected by the anti-malware software. Much more advanced obfuscation techniques have been invented by the hackers to protect their malwares from being captured by the antimalware software. In order to detect the malware and protect the computer or system being attacked by the hackers, machine learning is equipped to the malware detection methodology to enhance the detection ability [3].

According to the study of [4], its result proven that by using machine learning could enhance the robustness in malware detection application. There are various kinds of feature which available to be used as training features, such as opcode, text string and byte code. In research [5], text string feature was involved to train machine learning classifier. Text string was selected in this research owing to its informative and small memory size. Instead of using text string as the features to train the machine learning classifier, other features such as byte code, PE header, API calls and opcode are available for this purpose. In research [6], text string is also used as the feature to train and test the machine learning classifier for detecting obfuscated malware.

New malware variant can carry some prevalent content from the previous malware variant. Based on this hypothesis, this work is proposed where the significant opcode is referring to the prevalent content. In this paper, the opcode is chosen as the features to be trained by machine learning since it is informative and the data size is relatively smaller. N significant opcodes where n=10, 20, 30, 40 and 50 are extracted from the malware assembly code files which originally from the malware executable files. Based on machine learning classifier models that are generated using these significant features, the performance in term of accuracy and time taken between classifiers is observed to detect obfuscated malware. Then, using the best observed model, its performance is benchmarked by using text string features. The novelty in this work is by only training the significant opcode, the machine learning model is able to detect the malware kind with a promising performance.

2. LITERATURE REVIEW

Obfuscation is mean uncertain and obscure. Obfuscation technique is applied by the malware creator to prevent that malware being recognized by the malware detection system. The malware could simply bypass the detection by rearrange the sequence of the code and insert redundant code into it [7]. Malware analysis techniques could be classified as static, dynamic and hybrid. Static analysis i.e signature-based and machine learning-based techniques determine whether a program or code is malicious or not without executing the program or code. Signature-based detection stores malicious behavior of malware that could be represented as signature and is stored in repository. At present, the signature-based detection would require the expertise to keep the signature database up to date by creating the signatures manually. This approach is not able to capture the obfuscated malware since the repository does not include those signatures [8]. Machine Learning (ML) based detection is able to enhance malware detection by using several kinds of data, network as well as cloud-based anti-malware components. According to the definition that given by Arthur Samuel, ML is a set of methods that enable computers to equip the ability to learn without being explicitly programmed. Algorithm is very significant to ML where it helps to discover and formalize the raw data. ML is also allowed diverse of approaches to reach to the desired solution rather than single method [9]. In this study, static technique (supervised ML) will be used where the technique predicts the value based on previous labeled training dataset [10]. The purpose of applying ML models is to perform predication via computers to accomplish high performance outputs.

Dynamic analysis refers to the process of analyzing and observing the functionality of software while it is being executed. A virtual environment is required to observe the behavior of executed malicious software. Anomaly-based detection is an example of dynamic analysis and is one of the common malware detection methods where it could implement detection in training and testing phase. It tends to learn the normal behavior of the host during training phase. The primary advantage of anomaly-based detection is its capability to recognize and detect the unknown attacks. Nevertheless, this detection approach suffers with high false alarm rate as well as its complexity of deciding key features to be learned in the training phase [8]. Hybrid analysis is an analysis approach which contains both static and dynamic characteristics [11].

Opcode is known as the instruction to the machine or hardware [12]. The significant opcode is referring to the frequently repeated opcode in the assembly code. In research [13], 20 frequently repeated opcodes were extracted based on the analysis on total of 100 benign and malicious samples. Based on the hypothesis, new or unknown malware can carry some prevalent content from the previous malware. Hence, significant opcode is referring to the prevalent content. In fact, a wide range of features are available for computer malware detection. Nevertheless, each feature comes with its advantages and disadvantages. Table 1 shows the differences between several features in terms of size, memory overload, runtime as well as information retrieved. Based on the comparison as shown in Table 1, byte code and text string have larger data size to contain more information as compared to opcode. Meanwhile, opcode is a better option for its smaller data size and memory overload.

According to [5], text string that extracted from the assembly code was the main feature that used to train the classifiers such as Naïve Bayes, Sequential minimal optimization (SMO) and J48. Those classifiers were tested with 10-fold cross-validation after training completed. The malware dataset and obfuscated malware dataset were uploaded to VirusTotal website and found that 86.57% of normal malware and 21.43% of obfuscated malware could be detected with signature-based detection. Nevertheless, the well-trained machine learning classifier in this research was capable to detect the obfuscated malware achieve 99.5% accuracy.

On the other hand, malware detection based on opcode frequency was proposed by [13]. The executable files were downloaded from the Sandbox Cuckoo website and converted to assembly code by using IDA pro disassembler. Instruction Counter Plug-in was used to analyze the assembly code to collect frequency of the opcodes that found in the assembly code. After that, top 20 frequently appear opcodes were chosen as the feature vectors to train the machine learning algorithms such as SVM, RF, BOOST and Decision tree. As a conclusion, the proposed methodology in this research could achieved 96.67% of success rate with RF classifier.

Table 1. Comparison between different features [5].

Features	Byte Code	Opcode	Text string
Data Size	Large	Small	Medium
Memory Overload	Large	Small	Small
Runtime	Slow	Fast	Fast
Information retrieved	Full	Part	Full

3. METHODOLOGY

This research methodology involved of two different data features which are opcode and text string to compare their performance for malware detection in computer devices. There are some steps to carry out the research methodology such as data collection, data preprocessing, training phase (feature extraction and classifier training) and testing phase as shown in Figure 1.

3.1 Data collection

Benign and malicious executable (exe) files are randomly selected and downloaded from diverse resources. Total 500 benign datasets are obtained from the Window Executable files website, NirSoft [14] and EXE files [15]. On the other hand, total 600 malicious datasets are downloaded from VxHeaven [16], DasMalwerk [17] and VirusSign [18].

3.2 Data preprocessing

Data preprocessing involves some processes such as conversion the exe files to assembly code and removal of redundancy of the code lines. After retrieving the benign and malicious executable files from the open source, these executable files will be converted into assembly code with the disassembler tool [19]. In order to handle multiple files without running the command in terminal manually, a Perl

script is designed to automate the process of conversion. The converted assembly code files will undergo the data preprocessing process to remove the redundant lines of code. The redundant code such as next line, comment and spacing will be removed. By removing the redundant code from the assembly code will help to improve the training quality and save time for training and testing. In this research, there are two types of training set used i.e opcode and text string. For opcode features dataset, only opcode and its counting will be included in the dataset. The opcode counting will be collected with an instruction count script. Meanwhile, for text string features dataset will include opcode and operand in the dataset. Nevertheless, text string feature dataset would need to undergo the data preprocessing to remove the line number, and hex number in the assembly code. In addition, text string feature dataset need to undergo the "StringToWordVector" filter where it will convert string attributes into a set of numeric attributes to represent word occurrence information from the text contained in the strings [20]. This filter is available in Weka [21] tool before the text string dataset ready to be used as training dataset.

Total 1100 malware and benign exe files are used in this research as mentioned in Section 3.1. 100 out of 600 malware files are reserved as unknown malware dataset to test the classifiers in testing phase. For opcode, the training dataset in csv file format is consisting of 500 malware and 500 benign files, whereas only 300 of malware and 300 of benign files were used in the training dataset owing to the huge file size. The details of the dataset distribution are shown in the Table 2.

Table 2. Dataset distribution

Attributes	Opcode	Text string
Training set	500 Malware + 500 Benign	300 Malware + 300 Benign
Testing set	100 Unknown Malware + 100 Obfuscated Malware + 100 Benign	50 Unknown Malware + 50 Obfuscated Malware + 100 Benign

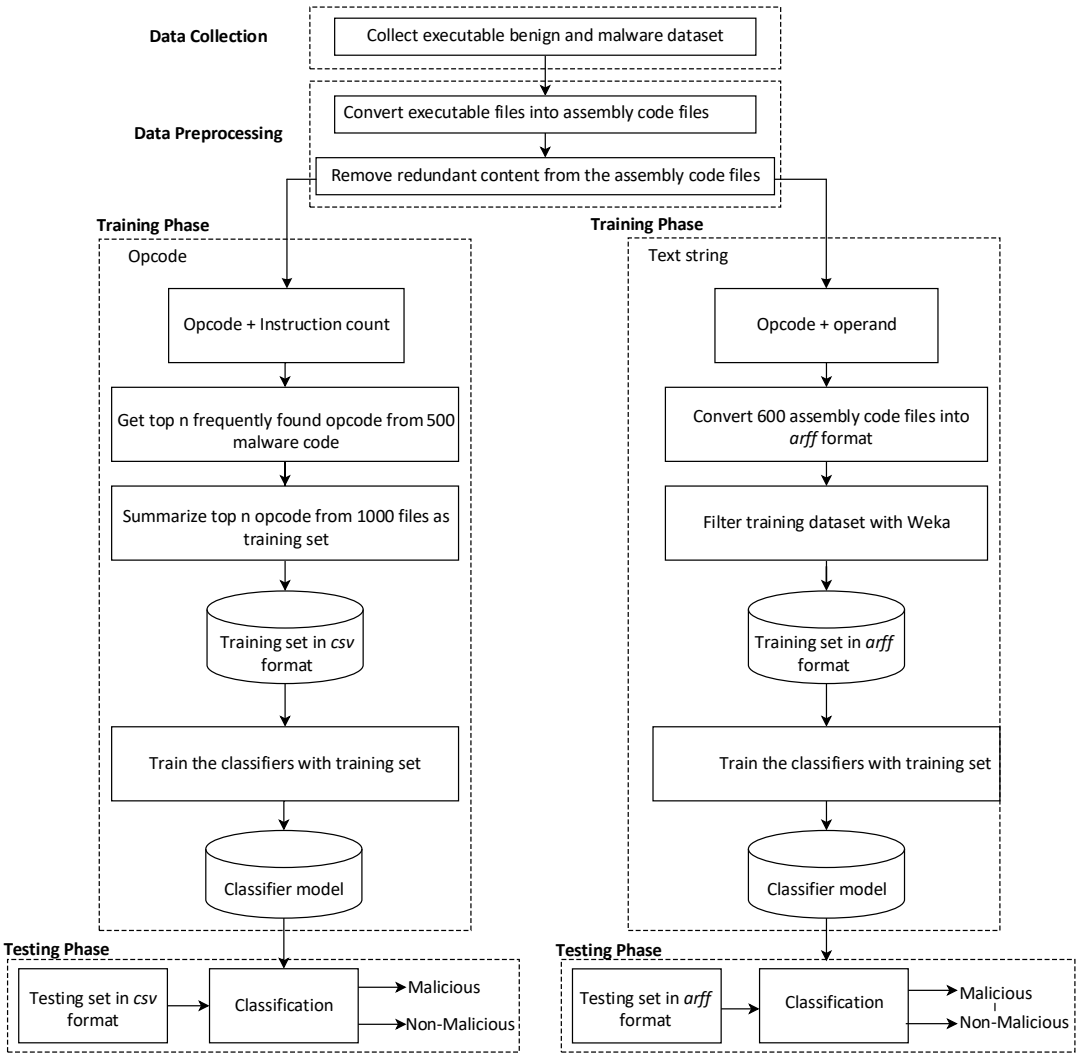

Figure 1. Flow of work

3.3 Training phase

In training phase, features are extracted from the assembly codes as the training set. These features are trained to generate the classifier models. Based on Figure 1, these two types of training set i.e opcode and text string are prepared by using different methods. Based on the hypothesis, prevalent content could be the content that exist in the previous malware variant also could be found in the new malware variant.

This prevalent content is referred as the most significant content for this study. Therefore, the instruction count of each malware assembly code has been recorded and only top 50 most commonly used opcode of each assembly code is listed. Then, the list of 50 most commonly used opcode from 500 malware assembly code was combined into a single file to sort out the global top 50 most commonly used opcode from 500 of malware files. After that, the instruction count of the top 50 opcode were extracted from each malware assembly code and saved into a csv file. The steps are repeated for top 10, top 20, top 30 and top 40 most commonly used opcode to observe the accuracy of classification.

To benchmark the opcode features classification, other type of features is used. Using the same preprocessed assembly code files, text string features are extracted as the training set. Text string features would gather more information as compared to opcode features, however, its training file size is larger than the opcode training set. Therefore, fewer files are used for text string training set. The text string training set that contains opcode and operand are converted into Attribute-Relation File Format (ARFF) files format. Then, the dataset has to be filtered with "StringToWordVector" filter that available in Weka before proceed to the training stage. Next, the features are trained based on machine learning algorithms to generate the classifier models. The classifier models are generated separately in Weka according to the types of training set.

3.4 Testing phase

The performance of the classifier models would be evaluated with the matrixes such as time taken to build model, time taken to test model and accuracy. Accuracy is defined as the proportion of correct forecasts to total predictions as shown in Equation (1)

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (1)$$

where, True Positives (TP) represents the correctly identified malware files, True Negatives (TN) represents the correctly identified benign files, False Positives (FP) represents the benign files that were incorrectly identified as malware files while False Negatives (FN) represents the malware files that were incorrectly identified as benign files.

By training and testing using the same dataset, classifiers that performed better in term of classification accuracy were selected to be tested with other supplied test set which is obfuscated malware files. The obfuscated malware data was prepared in advanced by using Crypto Obfuscator [22].

The supplied test set are categorized also into two different types i.e opcode test set and text string test set. For opcode feature testing set, 100 unknown malware, 100 obfuscated malware and 100 benign data are involved to validate the performance of the classifiers. The best performance of classifier in term of accuracy is observed. To benchmark the performance of the classifier, the classification is repeated using the text string features. Since the file size of text string testing set is too large, only 50 unknown malware, 50 obfuscated malware and 100 benign data are included in the testing set. Figure 2 shows the overall testing process for both opcode and text string features.

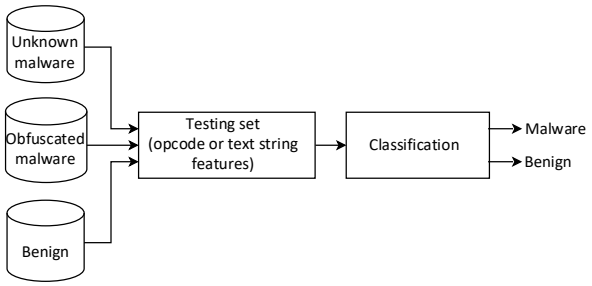

Figure 2. Testing process for opcode and text string feature dataset

4. RESULT AND DISCUSSION

4.1 Top 50 opcode in malware

Table 3 shows the comparison of top 50 opcodes in known malware and obfuscated malware. Top 50 frequently used opcodes in 500 known malware files and 100 obfuscated malware files are mostly similar. Nevertheless, some of the opcode in known malware are missing in obfuscated malware. On top of that, some new opcodes have been found in the obfuscated malware where they are not existing in known malware.

Table 3. Top 50 opcode in known malware and obfuscated malware

Known Malware	Obfuscated Malware
adc, add, and, arpl, bound, call, cmp, cmpl, dec, gs, imul, in, inc, incl, insb, insl, ja, jae, jb, jbe, je, jg, jge, jl, jle, jmp, jne, jns, jo, js, lea, lods, mov, movb, movl, or, out, outsb, outsl, pop, push, pushl, ret, sbb, scas, stos, sub, test, xchg, xor	adc, add, addr, and, arpl, bound, call, cmp, cs, dec, es, fs, gs, idiv, imul, in, inc, insb, insl, ja, jae, jb, jbe, je, jle, jmp, jne, jnp, jns, jo, js, lcall, ljmp, lods, mov, nopw, or, out, outsb, outsl, pop, push, sbb, scas, ss, stos, sub, test, xchg, xor

4.2 Classification accuracy of classifiers

Total of 34 classifiers have been implemented in the research to find the suitable or best classifier which has less time taken to build and test model and high accuracy. However, only 6 classifiers have been chosen owing to their outstanding performance to malware classification with 100% accuracy, which are IBk, KStar, Random Committee, Randomizable Filter Classifier, Random Forest and Random Tree. Top 10 opcodes, top 20 opcodes, top 30 opcodes, top 40 opcodes and top 50 opcodes are used in the experiment to figure out the dependency of number of opcodes to the classification accuracy. As the result, some of the classifiers are perform better with the incremental of opcodes number. Nevertheless, there are few classifiers would have worse performance when the number of opcodes increases.

4.3 Testing result of classifiers

In this research, top 50 most commonly used opcode could help to enhance the accuracy of classification, therefore top 50 opcode are taken as the primary features for training and testing of classifiers. Based on the classification result that described in Section 4.2, several classifiers have outstanding performance in classifying the malware, where their classification accuracy are 100%. Hence, all these classifiers have been listed out and tested with supplied testing set to find out their testing performance. IBk, KStar, RandomCommittee, Randomizable Filter Classifier, Random Forest and Random Tree are tested with supplied 100 obfuscated malware, 100 unknown malware and 100 benign files (as shown in Table 2). Table 4 shows the top performance classifiers in testing dataset classification. As a result, Random Forest has spectacular performance in testing phase where its accuracy is 98.67%. Based on the obtained result as shown in Table 4, evenly distribution on benign and malware dataset still able to contribute good classification accuracy.

Table 4. Top performance classifiers in testing dataset classification

Algorithm	Time to build model (s)	Time to test model (s)	FP Rate (%)	Accuracy (%)
IBk	0.00	0.03	4.00	96.00
KStar	0.00	2.33	10.00	90.00
Random Committee	0.04	0.01	2.33	97.67
Randomizable FilterClaasifier	0.00	0.02	5.33	94.67
RandomForest	0.23	0.01	1.33	98.67
RandomTree	0.01	0.01	8.67	91.33

4.4 Opcode vs Text string features

The performance of Random Forest classifiers that trained with opcode feature and text string feature are being compared since it performed outstanding in classification accuracy while relatively still shows less time taken to build and test the model. Total amount of dataset is being used in training and testing stages are not identical for opcode feature and text string feature (as shown in Table 2). This is owing to the constraint of file size where text string feature dataset is too large to be managed. Based on the result as shown in Table 5, classification using opcode features is still resulting an acceptable accuracy even though around 2% less compared using text string features. However, total time of the classification is more than 10 times faster than using the text string features.

Table 5. Performance of Opcode and Text string features

Classifier	Attributes		Opcode	Text string
Random Forest	Training	Time taken to build model (s)	0.23	1.96
	Testing	Accuracy (%)	98.67	100
		Time taken to test model (s)	0.01	0.85

4.5 Performance of signature-based detection

In order to know the performance of signature-based classifier, 100 of known malware and 100 of obfuscated malware have been tested with VirusTotal [23]. The result shows that the known malware executable files could be detected with detection rate of around 42% whereas, the obfuscated malwares executable files are around 20%. In another word, VirusTotal has difficulty to detect the obfuscated malware.

4.6 Related works comparison

In research [5] and [6], text string feature is the primary feature that being used for classifiers training and testing. However, research [5] and [6] had different total numbers of malware and benign samples. Total of 500 benign and 500 malware samples have been used in research [6]. While research [5] only utilize 150 benign and 150 malware samples for classifier training and testing. SMO classifier in research [5] able to achieve 98% of testing accuracy within 0.08s. Nevertheless, SMO classifier in research [6] is able to achieve 99.1% of testing accuracy within 0.47s. The Random Forest classifier used in this research is substantially more slowly than the classifiers used in studies [5] and [6]. However, Random Forest classifier has a higher testing accuracy than the SMO classifier.

5. CONCLUSION

Obfuscated malware is barely detected by the signature-based detection application, VirusTotal, where only around 20% detection rate. Machine learning classifiers are performed far better than the signature-based detection with more than 98% of accuracy. On the other hand, the machine learning classifiers that trained with text string and opcode are obviously different in term of time taken to build and test the model. From this study, it is observed that the performance of the classifier that is trained with significant opcode features only is as good as the classifier that is trained with more informative features such as text string features. For the future woks, instead of using opcode and text string as main features to train the

classifiers, other features could be used to observe their performance in malware classification.

ACKNOWLEDGMENT

The authors would like to thank Elektrika Editor Team for their helpful feedback and support.

REFERENCES

[1] L. E. S. Jaramillo, "Malware Threats Analysis and Mitigation Techniques for Compromised Systems," *Journal of Information Systems Engineering & Management*, vol. 4, no. 1, Mar. 2019, doi: 10.29333/jisem/5742.

[2] A. Darem, J. Abawajy, A. Makkar, A. Alhashmi, and S. Alanazi, "Visualization and deep-learning-based malware variant detection using OpCode-level features," *Future Generation Computer Systems*, vol. 125, pp. 314–323, Dec. 2021, doi: 10.1016/j.future.2021.06.032.

[3] 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*, vol. 153. Academic Press, Mar. 01, 2020. doi: 10.1016/j.jnca.2019.102526.

[4] W. Fleshman, E. Raff, R. Zak, M. Mclean, and C. Nicholas, "Static Malware Detection & Subterfuge: Quantifying the Robustness of Machine Learning and Current Anti-Virus", *2018 13th International Conference on Malicious and Unwanted Software (MALWARE)*, Nantucket, MA, USA, 2018, pp. 1-10, doi: 10.1109/MALWARE.2018.8659360.

[5] T. H. Xin, I. Ismail, and B. M. Khammas, "Obfuscated Computer Virus Detection using Machine Learning Algorithm," *Bulletin of Electrical Engineering and Informatics*, vol. 8, no. 4, pp. 1383–1391, Dec. 2019, doi: 10.11591/eei.v8i4.1584.

[6] M. Hasan A. Ali, *Metamorphic Malware Detection Using Machine Learning*, Master's Thesis, Faculty of Engineering, Universiti Teknologi Malaysia, 2020.

[7] J. Singh and J. Singh, "Challenges of Malware Analysis: Obfuscation Techniques", *International Journal of Information Security Science*, Vol. 7, No. 3, pp. 100-110, 2018.

[8] N. Idika and A. P. Mathur, *A Survey of Malware Detection Techniques*, Technical Report, Department of Computer Science, Purdue University, 2007.

[9] K. Eugene, "Machine Learning for Malware Detection". Accessed on: Dec. 15, 2021. [Online].Available: https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning.pdf

[10] Kateryna Chumachenko, *Machine Learning Methods for Malware Detection and Classification*, Bachelor's Thesis, Information Technology, University of Applied Sciences, Berlin, Germany, March 2017.

[11] R. Tahir, "A Study on Malware and Malware Detection Techniques," *International Journal of Education and Management Engineering*, vol. 8, no. 2, pp. 20–30, Mar. 2018, doi: 10.5815/ijeme.2018.02.03.

[12] B. Ashutosh, "What is Opcode". Accessed on: Jan. 29, 2022. [Online]. Available: https://www.engineersgarage.com/what-is-opcode/

[13] A. Yewale and M. Singh, "Malware detection based on opcode frequency," in *Proceedings of 2016 International Conference on Advanced Communication Control and Computing Technologies, ICACCCT 2016*, Jan. 2017, pp. 646–649. doi: 10.1109/ICACCCT.2016.7831719.

[14] "NirSoft". Accessed on: Jun. 02, 2024. [Online]. Available: https://www.nirsoft.net/

[15] "EXE Files". Accessed on: Jun. 02, 2024. [Online]. Available: https://www.exefiles.com/en/

[16] "VX Heaven". Accessed on: Jun. 02, 2024. [Online]. Available: https://vxug.fakedoma.in/archive/VxHeaven/

[17] "DasMalwerk". Accessed on: Jun. 02, 2024. [Online]. Available: https://dasmalwerk.eu/

[18] "VirusSign". Accessed on: Jun. 02, 2024. [Online]. Available: https://www.virussign.com/downloads.html

[19] "objdump". Accessed on: Dec. 15, 2021. [Online]. Available: https://command-not-found.com/objdump

[20] "Class StringToWordVector". Accessed May. 19, 2022. [Online] https://weka.sourceforge.io/doc.dev/weka/filters/unsupervised/attribute/StringToWordVector.html

[21] "Weka 3". Accessed Jan. 15, 2022. [Online] https://www.cs.waikato.ac.nz/ml/weka/

[22] "Crypto Obfuscator For .Net". Accessed Jan. 15, 2022. [Online] https://www.ssware.com/cryptoobfuscator/obfuscator-net.htm

[23] "VirusTotal". Accessed Jan. 15, 2007. [Online]. Available: https://www.virustotal.com/gui/home/upload