International Journal of Difference Equations (IJDE).
ISSN 0973-6069 Volume 15, Number 2 (2020), pp. 243-258

https://dx.doi.org/10.37622/IJDE/15.2.2020.243-258

A Dynamic Malware Detection in Cloud Platform

Nani Lee Yer Fui[1], Aziah Asmawi[2] and Masnida Hussin[3]

[1]Universiti Putra Malaysia,

[2]Universiti Putra Malaysia,

[3]Universiti Putra Malaysia,

Abstract

Cloud computing not only provides high availability on elastic resources, scalable, and cost-efficient. The platform is also widely used in information technology (IT) to support technology infrastructure and services. However, due to the complex environment and scalability of services, one of the highest security issues is malware attacks, where some of the antivirus scanner unable to detect metamorphic malware or encrypted malware where these kinds of malware able to bypass some traditional protection solution. This is why a high recognition rate and good precision detection are important to eliminate a high false-positive rate. Machine Learning (ML) classifiers are a critical role in artificial intelligent-system. However, machine learning will require to learn from the high amplitude of input data; classify then only able to generate a reliable model with a high detection rate. The objective of this work is to study and performs detection based on dynamic malware analysis and classification is through the WEKA classifier and Random Forest Jupyter Notebook. There are three classifiers chosen in this work, which are Random Forest, J-48, and Naive Bayes with 10-folds validation from the WEKA tool and another additional classifier from Random Forest - Jupyter Notebook to substantiate the accuracy.

Keywords: cloud security, malware, static malware analysis, dynamic malware analysis

1. INTRODUCTION

Cloud Computing platform is the on-demand availability of computer resources from applications to storage and processing power over the internet rather than own physical server, computing infrastructure, or data centers [19,20]. The moving toward Cloud computing resources as a service that is provided remotely to the users, makes computer hackers actively tried to steal online data without needed to rent for the services. There are very frequent incident occurs caused by malware such as data leak [8], the corporate

website being unavailable or incidents like ransomware and wannacry. ClearSky Research Team (2018) had reported there are nine Iranian hackers who were charges with infiltrating 144 US universities, 176 universities in 21 other countries, and 47 private companies in March 2017. The attack spreads malware knows as a VPN filter impacted more than 500,000 routers worldwide which can be used to create a massive botnet, spy, and manipulate web activity on the compromised router at the end of May 2017. Another incident reported was the US government hinted and attributed the NotPetya malware by Russian for power grid hacking in February 2017 [1].

2. MALWARE TYPES

The definition of malware is malicious software with the intent to bring harm or to infect the machine. This comprises many types of threats including ransomware, Trojan, computer virus adware, and many more [4]:

- **Spyware**

  The main purposes of spyware are to trail user browsing habits or any other personal information and send it to a remote user. It also serves as a host silently which will install an undesirable program from the internet.

- **Ransomware**

  Ransomware will affect the computer by encrypts the files or data stored in the machine. It also displays a warning notification asking for bitcoin to decrypt and recover back the data or files.

- **Adware**

  Adware has the aim to display advertisements or redirect search requests to other advertising websites. Adware can serve like cookies to gather data such as websites visited by the user. From the information gathered, a custom advertisement will be displayed.

- **Virus**

  The main feature of a virus is a malicious program created by cybercriminals by infecting files on the target machine. The virus is designed to spread from one machine to another.

- **Worm**

  The worm is an infectious virus design to reproduce itself to infect another machine. They spread through a computer network and be dependent on security failures to access the target machine. Many worms are intended to steal, delete data, and spread to other systems.

- **Trojan**

  A trojan is a malicious code that masquerading as useful code or legal software. It is employed by cybercriminals to gain access to user's systems. Users are

typically deceived by social engineering unintentionally executing Trojans in their systems.

- **Key logger**

  A key logger records all activities made on a machine like monitoring tools. It regularly executes with no permission of the users. A key logger is usually used to obtain confidential information such as private information, security phrase, usernames, and passwords.

- **Rootkit**

  A group set of software tools created to provide unauthorized user access with administrator privileged access to a computer is known as a rootkit. Once the rootkit is installed, it can remotely change system settings or execute files. Rootkits must be installed on a host and cannot self-propagate.

- **Bots and Botnets**

  Bot and botnets are malicious code created to penetrate a computer and will execute instruction once received instruction from a remote-control server. Bots can self-replicate like Trojans and viruses. A group of bots known as botnets can be used to launch distributed denial of service (DDoS) attacks in an attempt to make a network communication temporarily inaccessible.

3. STATIC MALWARE ANALYSIS VERSUS DYNAMIC MALWARE ANALYSIS

Malware analysis is a process of determining the purpose and functionality of a piece of malware. This process will reveal what kind of harmful program has affected the machine, the capable of it causing damage, and the most important is how to remove it. Malware analysis gives a better understanding by identifying the nature of the malware. There are two types of malware analysis: static malware analysis and dynamic malware analysis [6].

3.1 Static Malware Analysis

The static malware analysis investigates a malware file without actually running the code. This is the securest way to analyze malware, as executing the code could affect your system. In its most basic form, static analysis gathers information from malware without even viewing the code. Metadata such as file name, type, and size can yield clues about the nature of the malware [2].

3.2 Dynamic Malware Analysis

Dynamic malware analysis, unlike static malware analysis. It involves analysis while running the code in a controlled environment. In dynamic malware analysis, the

malware is run in a closed, isolated virtual environment, and then its behavior will be analyzed. The objective is to study its functioning and behavior and use this knowledge to eliminate the infection. In advanced dynamic malware analysis, debuggers are used to determine the functionality of the malware [2].

4. RELATED WORKS

In recent years, several scholars have used data mining methods to analyze the features of malicious code. This approach has become the mainstay of malware detection because it is highly efficient and has a low rate of false positives compared with traditional heuristic-based detection methods [13].

[3] present a novel method for the signature generation which does rely on any specific aspect of the malware, thus being invariant to many modifications in the malware code. The proposed method consists of the following steps in the unsupervised training phase: Given a dataset of malware programs, run each program in a sandbox to generate a text file containing the behavior of the program. Then, parse the sandbox text file and convert it to a binary bit-string to feed it to the neural network. Next, a deep belief network implemented using deep denoising autoencoders is trained by layer-wise training.

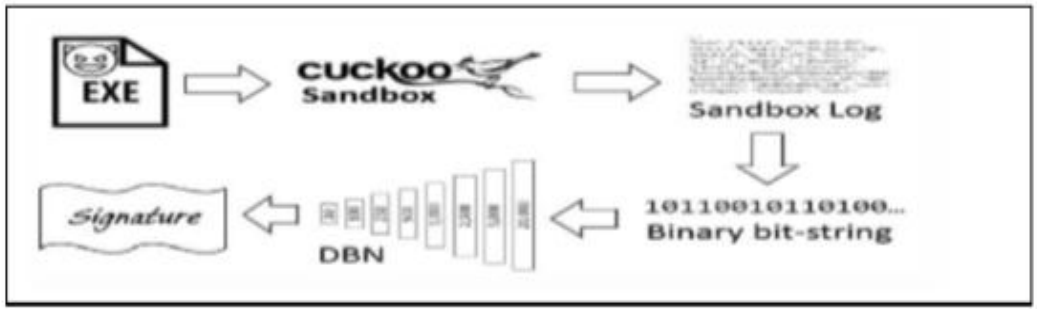

Figure 1. System Methodology Using Deep Belief Network (DBN) [3]

Figure 1 illustrates the methodology for data collection and feature vectors data extraction, execute dataset in a sandbox, extract and remove the unigrams with no information at all. The second step is to choose the highest frequency and lastly convert each sandbox file into a fixed-sized bit string. The feature vectors extract during the execution of malware are API calls, Websites, and also Ports accessed can be used for future analysis and process.

[11] indicate that metamorphic malware always evolves. The rise and widely used mutation engine enable malware to easily bypass detection using the conventional malware scanner. The mutation engine can be hidden and run in a small program, which will cause more complexity for malware analysts to detect metamorphic malware or encrypted malware. This approach more focuses on the detection scheme and classification of metamorphic malware according to the category.

Figure 2 shows the steps for the detection of metamorphic malware where after performing malware in portable executables, the collected data set is deconstruct using IDA-Pro for preprocessing and hence generates an API call graph. The author proposed the Gourmand Feature Selection algorithm for selecting desired features to eliminate unnecessary or irrelevant features. Once the metamorphic malware is detected, classification is done through the Chi-square distance measurement formula and histograms. Finally, the classification is done through the WEKA tool to identify whether the file is malware or benign. The highest accuracy obtained is J-48 with a 99.10% detection rate.

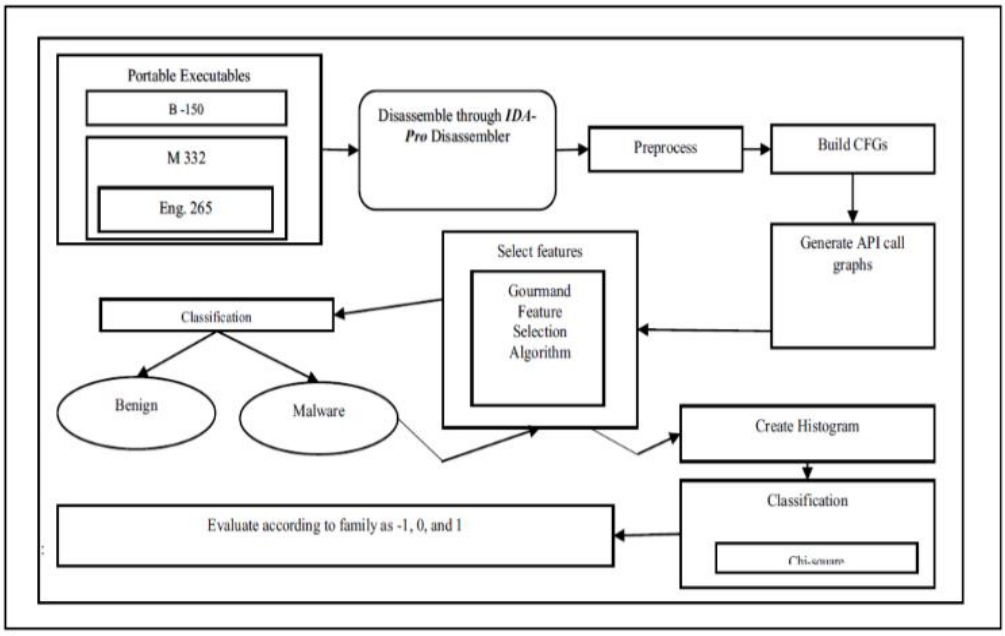

Figure 1. An Overview of Detection Classification Metamorphic malware [11]

During the validation stage [11] had run a total of 600 malware dataset, out of which 268 malware dataset is created using a mutation engine and another 332 corrupted malware dataset generated from the affected system that bypassed the antivirus software. The freshly installed windows generate another 150 benign samples for processing. This work only implements in executable format, hence it will be expanded in PDF files format or web browser in the future.

[9] indicate that malware has spread quickly. Most of the anti-malware in the market have unsuccessful to detect new variants malware mainly because most of the anti-virus relies on signature or token-based technique. The authors proposed a classic model for malware detection where malware executes in a cuckoo sandbox. The cuckoo will perform tracking and capture behavior of the executable files at the system level by extracting only the system call (i.e., operation field) for process and analysis. The collected or extracted data are then used to generate the sequence of N-Grams of specified length, and then applied the Information Gain (IG) feature selection method

to calculate a score.

After obtaining the highest score, the chosen top N-Grams will process by the classifier or classification purposes. Therefore this approach can efficiently detect malware based on a predefined signature. And lastly for data classification is done WEKA tool, where ARFF files were submitted to identify which classifiers achieved a low False Positive Rate (FPR) and high malware detection rate. The best classifier chosen in this study were Information Gain.

[5] work indicates that most of the traditional signature-based detection primarily relies on signature-based and payload information, to decrypt the payload information; indirectly additional cost is incurred, and processor usage is increased. To solve this issue, the author's proposed a method where components of detection reside in cloud resilience architecture using Support Vector Machine (SVM) at the KVM hypervisor level in Linux. KVM hypervisor is compatible with most of the cloud architecture. The system analysis setup consists of 2 nodes; one acts as a controller server for data collection purposes and another node serves as the storage server to store all the collected data.

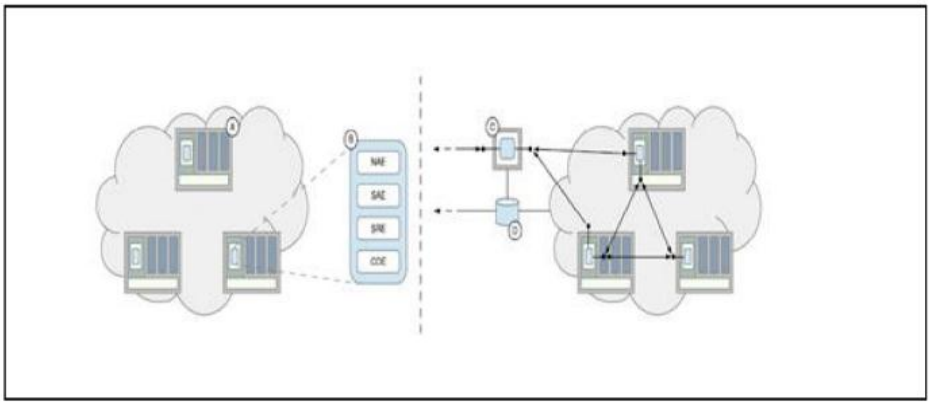

Figure 2. Overview of Detection scheme in cloud architecture [5]

Figure 3 illustrates the mechanism to collect data and analysis. To capture the real-time traffic tools such as tcpdump, libVMI CAIDA CoralReef software installed on each computer node for data collection. Authors extract the following raw feature actual size of memory usage, peak memory allocation or memory usage, and numbers of files. All these feature vectors extracted are useful for process and analysis. Authors have chosen the dataset which consists of multiple variants of malware because they have been identified as an evolving thread of window OS and have already compromised millions of machines worldwide between in years from 2010 until 2014. However, this model only workable in anomaly detection.

A manual classic approach inefficient to detect modern malware with obfuscation or encrypted techniques as the number of new variants of malware increased every day [10]. This research resolves the problem of obfuscation by analyzing, detecting, and classifying from a large amount of malware in an accurate and effective approach. The

authors have modified the malware analysis setup based on Cuckoo Sandbox, so that malware analysis capable execute in Virtual Machines (VMs) that are distributed and parallel manner, hence the efficiency is faster and accurate.

For classification and feature selection, the authors used Information Gain Ratio in the WEKA tool to eliminate the unrelated features or features that are redundant which can impact the precision rate. With Information Gain Ratio feature selection able to generate the most relevant attributes, these make the data are more accurate and high accuracy based on the feature selection by the WEKA tool.

An overview of the detection and classification scheme is illustrated in Figure 4 below. Each malware and clean ware that was executed in the cuckoo sandbox were labeled according to their families. Sample feature vectors extracted include API calls, registry keys as an input argument for further analysis. For data extraction and detection are implemented in Python and classification is via WEKA software.

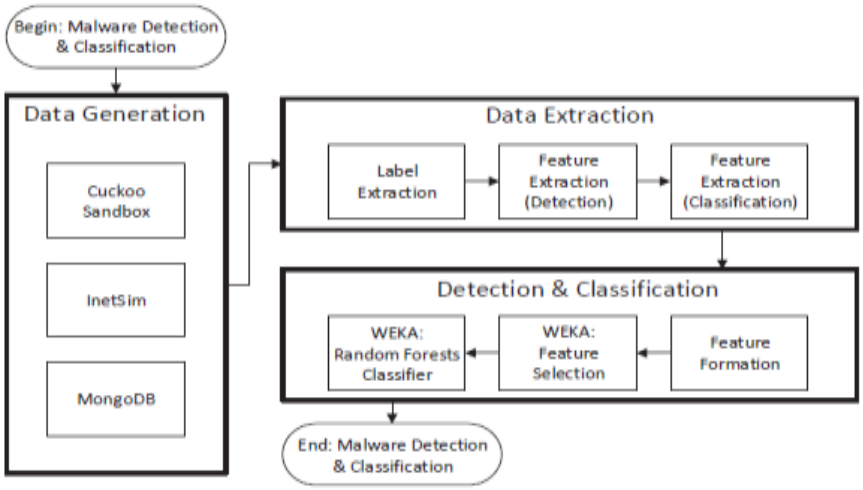

Figure 3. Overall System Flow for Both Malware Detection and Classification [10]

[12] indicate that static malware analysis is ineffective to detect malware that uses encryption technology. Authors have designed, developed, and implemented malware detection mechanism by capture the Kernel Function Calls trace of the guest system and produce a behavior file which is used as the data input for the deep learning neural network. These will be later be used for processing and classification.

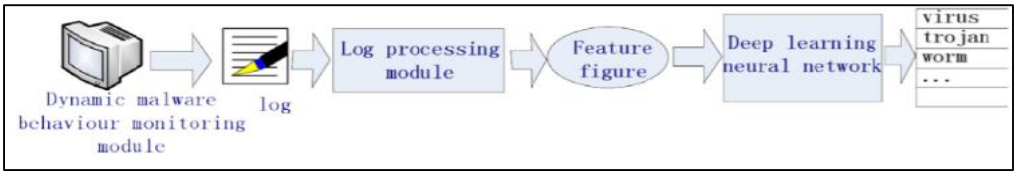

Figure 4. Mechanism Module [12]

Figure 5 shows the mechanism for data collection and classification, where malware is executed in a dynamic malware behavior monitoring module to capture malware

behaviors such as Kernel Function Calls. The output from this phase will be processed by Log Processing Module to obtain feature vectors that including function, Kernel Function Call, and call parameters. After obtaining all this information, a feature figure is generated and will pass to the deep learning neural network, followed by classification according to malware families.

[7] proposed three comprehensive phases; signature extraction phase, capture the signatures via the lightweight process, calculate the hash executables files. Figure 6 depicted a system flow of the proposed method.

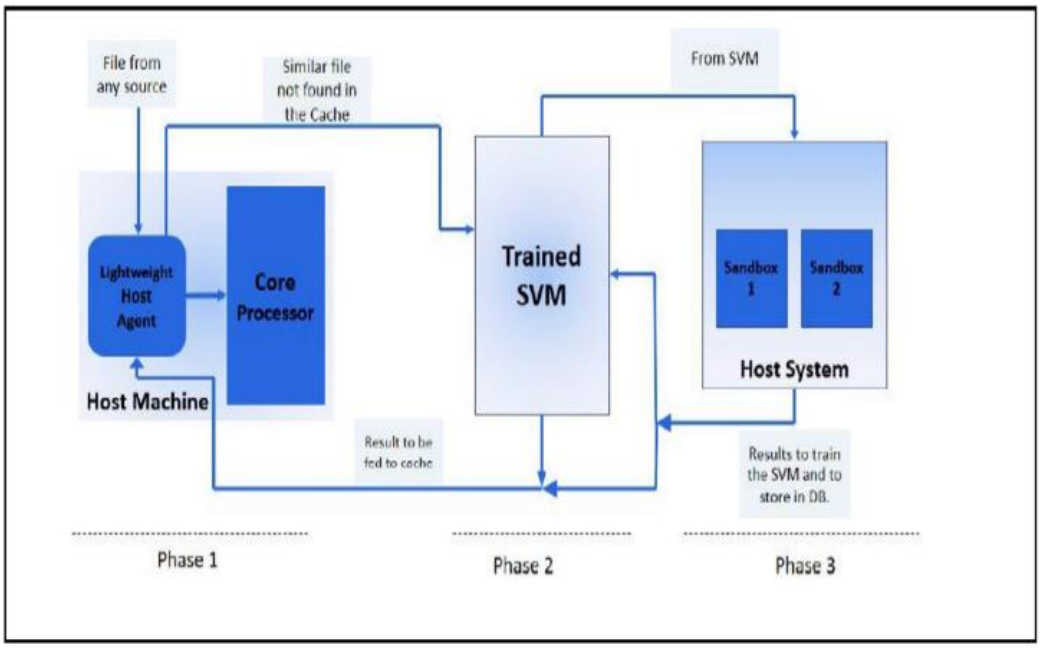

Figure 5. System Flow For Detection Technique [7]

Raw features such as memory allocation, number of files, or network requested by the process will be extracted by the authors. All the information will be used for further process and analysis.

5. METHODOLOGY

This section will explain the methodology used in this work. This chapter will detail out the component of the framework, system setup, dataset, software, and hardware applied in this work.

5.1 Research Framework

In Figure 7 depicted an overview of the system flow, starting from detection to classification phase by using Random Forest in WEKA tool [15] and Random Forest,

in Jupyter Notebook [16].

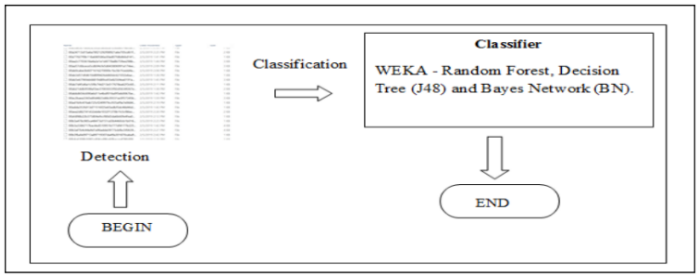

Figure 7. Overview of System Flow

5.1.1 Detection and Classification

This section present a method to detect malware by using certain parameters or feature vectors from the malware dataset. The goal of detecting malware is to distinguish malware and clean ware. The cloud tested used in this work is based on the XEN cloud platform and cuckoo sandbox. Figure 8 illustrates the flow of the framework to distinguish malware and benign.

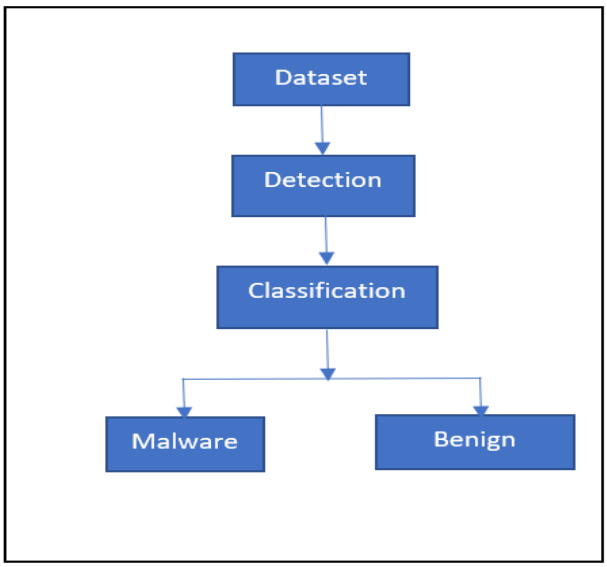

Figure 8. Illustration of The Framework

5.1.2 Malware and Clean Ware Dataset

In this study, the malware dataset was made available from Kaggle database [17] was chosen in this work, which contains around 9000 sample dataset. Based on the sample SHA files provided, some of the feature vectors collected are API call, permission, activity, and URL will use for further analysis. Some example work uses Kaggle dataset is [14].

5.1.3 Cuckoo Automated Malware Analysis

Cuckoo is the open-source malware analysis system. File formats such as PDF or malicious websites can analyze in cuckoo.

5.1.4 Random Forest

Random Forest [18] is a powerful ensemble method that uses a machine learning technique for classification. The reasons why Random Forest widely used are: (i) Eliminate overfitting and underfitting of data, (ii) Bootstrap method works decently fine on small datasets, (iii) Parallel processing while training, (iv) Automatic feature selection.

5.1.5 WEKA Software

WEKA is an open-source machine learning system written in java language, a GUI based for data mining, a classification that using a machine learning algorithm [15]. There are many kinds of research that have used the WEKA tool as a base for classification namely [9] and [10].

5.1.6 Machine Learning Classifier

Few classifiers have been used in this work, namely Random Forest, Decision Tree (J48), and Bayes Network (BN) in WEKA and Random Forest from Jupyter Notebook.

5.1.7 Naming and Calculation

Table 1. Naming and Calculation

Sr. No	Naming Conventions	Naming	Calculation
1	TP-Rate	True positive rate	TP/(TP+FN)
2	FP-Rate	false positive rate	FP/(FP+TN)
3	Precision	Precision	TP/(TP+FP)
4	Receiver Operating Characteristic Curve	ROC	A graphical plot equating the true positive rate and the false positive rate of a classifier.

5.2 SYSTEM SETUP

The cloud tested used in this work is based on the XEN cloud platform. The malware sample will run on a virtual machine and the monitoring server will serve as a node to

capture the entire feature vectors such as API, URL, and all this feature will then extracted for processing. The illustration of the system setup is depicted in Figure 9.

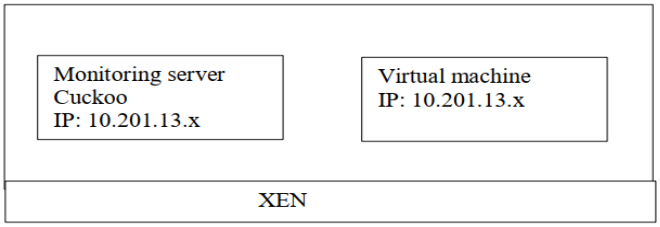

Figure 9. Illustration of the System Setup

5.3 SOFTWARE AND HARDWARE REQUIREMENT

Table 2 shows the software and hardware requirement required for the implementation of this work.

Table 2. Software and Hardware requirements

Software	**Hardware**
Windows 10 : Operating system	Computer Model : HP , 64-bit operating system
Microsoft Office 2016: To support entire work progress in terms of writing proposal, design and implementation.	Processor : Intel ® Core ™ i5-8250c
Cuckoo software : Detect and extract the malware dataset behavior.	Input device: keyboard. Output devices : optical mouse, monitor and printer
WEKA software : malware classification	Input device: keyboard, Output devices : optical mouse, monitor and printer

6. RESULTS AND DISCUSSIONS

This section elaborates on the results gathered from the work, result comparison between the proposed method and previous method. Table 3 below shows the result of 9600 dataset running with 10 fold cross-validation means to divide the dataset into 10 parts and average the results. Each data point was used once for testing, and 9 times for training. We have evaluated various machine learning classifiers to enhance the malware detection outcome for a large sample file of malware with an additional 600 malware size compare to the previous solution to reach maximum. The model has correctly classified 4774 as benign and 4826 samples as malware class. 99% of the instances are correctly classified. These results are promising and this experiment needs

to be taken further by increasing the sample space.

Table 3. Results from WEKA Algorithm

Sr. No.	WEKA algorithm	TP Rate	FP Rate	Precision	ROC Area	Accuracy %
1.	Random Forest	0.99	0.0	1.0	1.0	99.95
2.	J-48	0.99	0.0	1.0	1.0	99.93
3.	Naive Bayers	0.97	0.17	0.85	0.96	90.37

The results were obtained from a tenfold validation test using three selected classifiers to perform the experiments. The classifier's performance is measured with five evaluation metrics, namely True Positive Rate (TPR), False Positive Rate (FPR), precision, Receiver Operating Characteristic Curve (ROC), and accuracy depicted in Table 3.

Table 4. Results In Random Forest Jupyter Notebook

Sr.No.	Jupyter Notebook	AUC	Accuracy
1.	Random Forest	0.97	97%

In Table 4 the result shows that the algorithm achieved 97% accuracy and area under the curve (AUC) 0.97% by using Random Forest Jupyter Notebook.

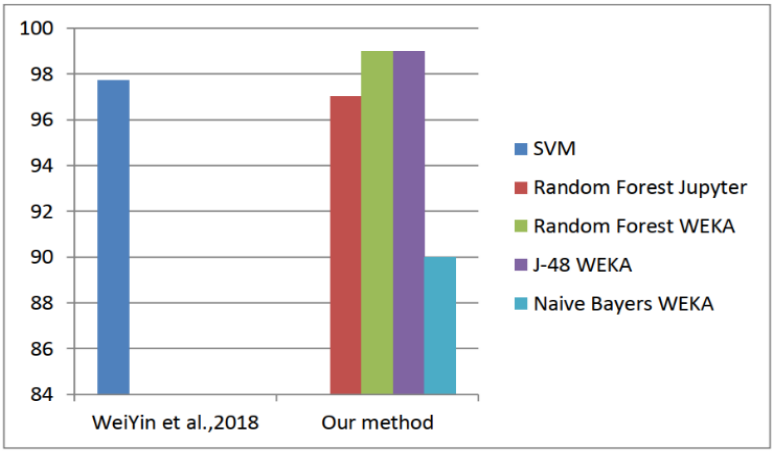

Figure 9. Comparison Performance of our Proposed Method against Anchor Paper

In Figure 9 shown the result comparison performance between anchor paper and our

proposed solution. The highest accuracy achieved in this work is 99.95% with Random classifier, followed by J-48 with 99.93 %, and Naive Bayers with 90.37%.

Table 5. Comparison of Processing Time (In Seconds) In WEKA Algorithm

Sr.	Classifier	Dataset size 9000	Dataset size 9600
1.	Random Forest	1.15	1.16
2.	J-48	0.27	0.25
4.	Naive Bayers	0.06	0.07

Subsequently, in Table 5 displays a processing time comparison (in seconds) to build a model depends on the dataset size, whereby with larger dataset sizes, the processing time increase.

Table 6. Comparison Performance of Our Proposed with Previous Solutions

Dataset	Algorithm	Dataset	Accuracy %
Vishakha et al.,2015	J-48	600	99.10
David et al.,2015	SVM	1800	96.4
Shivad et al.,2016	N-grams weka	6100	89.77
			-
Michael et al.,2016	SVM	8000	Above 90
Steven et al.,2016	WEKA	270,837	
WeiYin et al.,2018	Tensoflow	9000	97.73%
Saket et al.,2018	SVM	2k	-
Our method	Random Forest	9600	99

Table 6 lists out the comparison of the classification algorithm, dataset, and detection accuracy by the previous solution and our solution. The highest dataset uses were Steven et al., 2016 which contains 270,837, and Saket et al., 2018 with 20 malware

dataset. However, they did not mention what is the detection accuracy for their work.

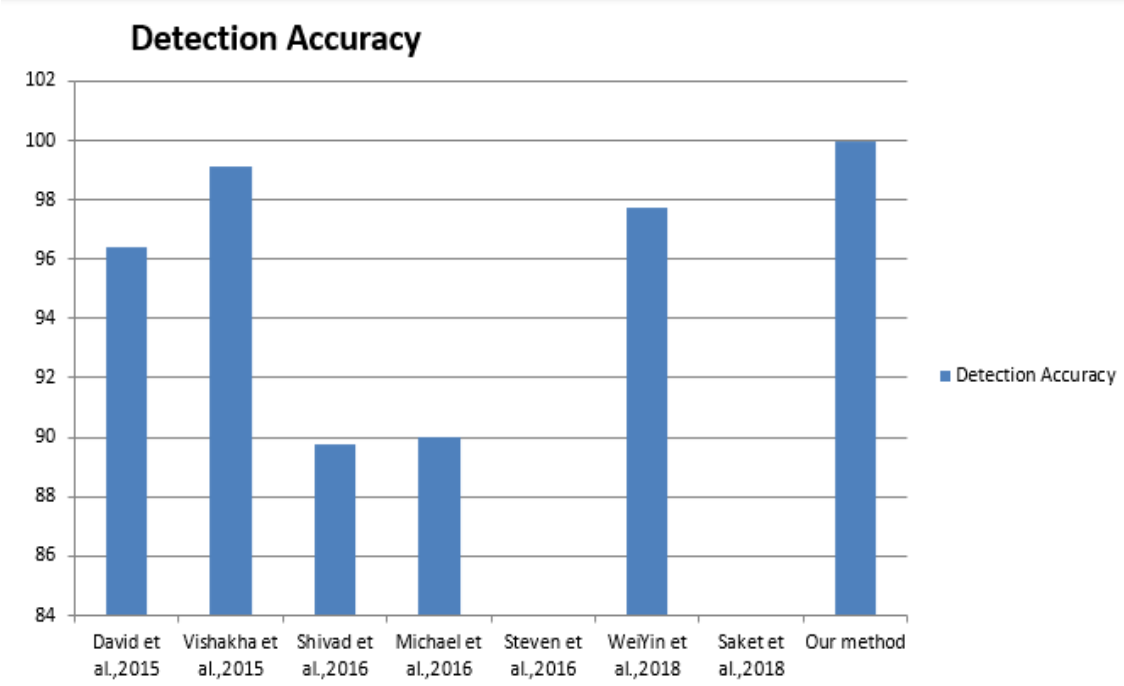

Figure 10. Comparative Performance of Our Proposed with Previous Solution

This work re-implement the previous solution from [12] by enhancing accuracy. However, in this work, data extraction is run in Jupiter Notebook and different datasets from kaggle.com. The reason why this dataset was chosen is because of the large sample of malware. Our work achieved a promising result which is around 99.99% detection accuracy classifier by WEKA tool and achieved 97% accuracy by using Random Forest Jupyter Notebook as depicted in Figure 10.

7. CONCLUSION

This work presented an evaluation result using machine learning classifiers to detect malware samples. In this study, we evaluated based on various machine learning classifiers to enhance the malware detection outcome for a large sample of malware's file in which an additional 600 malware size compare to the previous solution to obtained maximum high accuracy results. There are three classifiers chosen in this work, which are Random Forest, J-48, and Naive Bayes with 10-folds validation from the WEKA tool and another additional classifier from Random Forest - Jupyter Notebook to substantiate the accuracy. The experimental result indicates 99.9% detection rate, the larger the size of the dataset the higher processing time is needed to output the model.

8. LIMITATION

Due to the limitation on hardware compatibility issues and datasets, this work unable to implement the same as the base paper. Fortunately with the combination of several methods and a different set of malware datasets, this research able to enhance the accuracy of detection.

Acknowledgement

The authors would like to thanks Universiti Putra Malaysia as this study is part of the Putra IPM Grant (GP-IPM/2019/9676100).

REFERENCES

[1] ClearSky Research Team. "Cyber Intelligent 2017 Summary Report". www.clearskysec.com. January 1, 2018.Url: https://www.clearskysec.com/cyber2017/

[2] Damodaran, A., Troia, F.D., Visaggio, C.A. et al. A comparison of static, dynamic, and hybrid analysis for malware detection. J Comput Virol Hack Tech 13, 1–12 (2017). https://doi.org/10.1007/s11416-015-0261-z

[3] O. E. David and N. S. Netanyahu, "DeepSign: Deep learning for automatic malware signature generation and classification," 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, 2015, pp. 1-8, DOI: 10.1109/IJCNN.2015.7280815.

[4] John Love. "Malware Types and Classification." Lastline.com. March 28, 2018. Url: https://www.lastline.com/blog/malware-types-and-classification

[5] Michael R. Watson, Noor-ul-hassan Shirazi, Angelos K. Marnerides, Andreas Mauthe, and David Hutchison. (2016). Malware Detection in Cloud Computing Infrastructures. IEEE Transactions on Dependable And Secure Computing, Vol. 13, No. 2, March/April 2016.

[6] Sagar Khillar. "Difference Between Static Malware Analysis and Dynamic Malware Analysis." DifferenceBetween.net. July 23, 2018. Url: http://www.differencebetween.net/technology/difference-between-static-malware-analysis-and-dynamic-malware-analysis/ >.

[7] Saket Kumar,Chandra Bhim Bhan Singh. (2018). A zero-day resistant malware detection method for securing Cloud using SVM and SandboxingTechniques. Proceedings of the 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT 2018). IEEE Xplore Compliant - Part Number: CFP18BAC-ART; ISBN:9781-5386-1974-2.

[8] Satar, S. D. M., Hussin, M., Hanapi, Z. M., & Mohamed, M. A. (2018). Data Privacy and Integrity Issues Scheme in Cloud Computing: A Survey. *International Journal of Engineering & Technology*, *7*(3.28), 102-105.

[9] Shiva Darshan S.L.1, Ajay Kumara M.A.2, and Jaidhar C.D. (2016). Windows Malware Detection Based on Cuckoo Sandbox Generated Report Using Machine LearningAlgorithm. 2016 11th International Conference On Industrial and Information Systems (ICIIS).

[10] Steven Strandlund Hansen, Thor Mark Tampus Larsen, Matija Stevanovic and Jens Myrup Pedersen. (2016). An Approach for Detection and Family Classification of Malware Based on Behavioral Analysis. 2016 International Conference on Computing, Networking and Communications (ICNC), Workshop on Computing, Networking and Communications (CNC).

[11] Vishakha Mehra, Vinesh Jain, Dolly Uppal. (2015). DaCoMM: Detection and Classification of Metamorphic Malware. 2015 Fifth International Conference on Communication Systems and Network Technologies.

[12] Wei Yin, Hongjian ZHOU, Mingyang WANG, Zhiwen JIN, Jun XU. (2018). A Dynamic Malware Detection Mechanism Based on Deep Learning. IJCSNS International Journal of Computer Science and Network Security, VOL.18 No.7, July 2018.

[13] Ye, Y., Li, T., Adjeroh, D., & Iyengar, S. S. (2017). A survey on malware detection using data mining techniques. ACM Computing Surveys (CSUR), 50(3), 1-40.

[14] Yunan Zhang, Qingjia Huang, Xinjian Ma, Zeming Yang and Jianguo Jiang (2016). Using Multi-features and Ensemble Learning Method for Imbalanced Malware Classification. 2016 IEEE TrustCom/BigDataSE/ISPA

[15] Title: Weka 3: Machine Learning Software in Java. Url:https://www.cs.waikato.ac.nz/ml/weka/. Accessed: 20-10-2018.

[16] Title: The Jupyter Notebook. Url:https://jupyter.org/.

[17] Title: Malware dataset. Url https://www.kaggle.com/c/malware-classification.

[18] Title: Understanding Random Forests Classifiers in Python. Url: https://www.datacamp.com/community/tutorials/random-forestsclassifierpython. Accessed: 20-10-2018.

[19] Kovachev, D., Cao, Y., & Klamma, R. (2011). Mobile cloud computing: a comparison of application models. arXiv preprint arXiv:1107.4940.

[20] Mutanga, M. B. (2020). Service Discovery in Mobile Ad-Hoc Environments: A Solution Space Analysis. International Journal, 8(7).