Classifying Ransomware Using Machine Learning Algorithms

Chapter · October 2019

DOI: 10.1007/978-3-030-33617-2_5

Simon Parkinson
University of Huddersfield

Classifying Ransomware Using Machine Learning Algorithms

Samuel Egunjobi[1], Simon Parkinson[1][0000−0002−1747−9914], and Andrew Crampton[1][0000−0002−4635−9102]

Department of Computer Science, School of Computing and Engineering, University of Huddersfield, Queensgate, Huddersfield HD1 3DH, UK
{firstname.lastname}@hud.ac.uk

Abstract. Ransomware is a continuing threat and has resulted in the battle between the development and detection of new techniques. Detection and mitigation systems have been developed and are in wide-scale use; however, their reactive nature has resulted in a continuing evolution and updating process. This is largely because detection mechanisms can often be circumvented by introducing changes in the malicious code and its behaviour. In this paper, we demonstrate a classification technique of integrating both static and dynamic features to increase the accuracy of detection and classification of ransomware. We train supervised machine learning algorithms using a test set and use a confusion matrix to observe accuracy, enabling a systematic comparison of each algorithm. In this work, supervised algorithms such as the Naïve Bayes algorithm resulted in an accuracy of 96% with the test set result, SVM 99.5%, random forest 99.5%, and 96%. We also use Youdens index to determine sensitivity and specificity.

Keywords: Ransomware · Malware · Machine Learning

1 Introduction

This wide-scale use of computers has resulted in computing systems being used abusively for illegal activities [17]. Ransomware is a type of malicious software (malware), which when deployed on the computer encrypts or locks a computer or files, requesting that a ransom to be paid to the author of the ransomware for the successful decryption and release of the users data and system. It is intended to compromise the availability, confidentiality and integrity of the victims data, demanding payment from the user in order to have their files unencrypted and accessible [18].

There are two main types of ransomware this paper will be focusing on: the first one, Locker Ransomware, is designed to deny access to the victim's computer, to prevent them from using the system as a whole or services on the computer. The second is the Crypto Locker Ransomware which encrypts personal files to make them inaccessible to its victims [19]. In the case of Lockers Ransomware, the victim loses the ability to use the whole computer or a particular piece software on the computer. Crypto Locker Ransomware selects document, pictures or target files that have a favourable type. This causes a lot inconvenience and panic to victims as threats (e.g., blackmail, extortion) are made to facilitate quick response and payment by victims [18].

Typical techniques for detecting and classifying malwares often involve the use of static or dynamic features for analysis and detection [9]. One challenge with the methods of classification is the obfuscation of features and is explained in a recent paper [19]. Furthermore, this process is reactive, meaning that once a paper discussing malware detection is published, malware developers upgrade their form of attack so as to reduce or nullify the detection rates when run against anti-malware software and successfully carry out the intended action [9]. The proposed solution to this problem is increasing the alertness of anti-malware software to features of the malware software that increased reliable and detection accuracy. This includes utilising both static and dynamic features. Static features are those observed without executing the software, e.g file size, whereas dynamic features are those related to have the malware interacts with the system, e.g requests to operating system functionality or file interaction [6].

The second problem is the detection of false positives [9]. The introduction of malware that functions in a different way to those used during training of the machine learning [11] often results in mis-classification. These reduce the efficiency of the machine learning algorithm and its ability to accurately distinguish ransomware. In recent research, the authors propose a solution to increase the number of samples for both normal (goodwares) and malicious (malwares) software to cover a wide range of possible feature being taught to the machine [12].

In this paper, we explore the possibility of increasing the accuracy of detection and classification by integrating both static and dynamic features of ransomware, train machine learning algorithms and introduce test set with the aim to achieve a higher percentage of classification. A confusion matrix is used to observe the best algorithm with the least margin of error. The paper is structured as follows: First we discuss related work, and in the next section we explain the proposed technique. Following on, we present the research methodology for acquiring empirical understanding. We then give details about the samples used and how the classification was performed, followed on by the presentation of the results. Finally, a conclusion and recommendations as to future areas of research are provided.

2 Related Works

In this section, we review current research literature related to malware and ransomware analysis, with a particular focus on their research and analysis on the identification and classification of ransomwares.

2.1 Malware Analysis

An extensive review of the static feature analysis is provided in [11]. Static feature analysis is done using static features such as Application Programming

Interface (API) calling, binary code, control graph, etc. to find suspicious behavioural patterns of the malware. There are three different types of features used for the extraction process as mentioned in [9]: Byte sequence n-grams, string features, and portable executable. The byte sequence n-grams technique extracts n bytes sequences from an executable file; The string features refer to encoded text from the program file. The portable executable approach extracts from the dynamic link library located within the Win32 portable executable binaries [4]. Furthermore, some antivirus and anti-malware tools are detecting malware based on their signature. However, their accuracy is reduced due to obfuscation techniques which are easily implemented to prevent matching [16].

Despite the use of the static feature extraction to distinguish between malicious and benign software, there are some downsides to this technique. It is largely restricted due to the obfuscation method being carried out by adversaries [11, 12]. Furthermore, the entire process is reactive as it requires extensive manual human effort, time and expertise to formally describe unwanted behaviour. This can slow down the whole process of analysis especially when dealing with high volumes of malware variants to analyse [15].

An alternative approach to static analysis is dynamic feature analysis, which is performed while the malware is being executed. This focuses on system call sequences to provide information about the program at run-time, which is hard to obfuscate and uses various techniques like dynamic binary instrumentation, virtual machine inspection, information flow tracking and function call monitoring [16]. It involves the use of behavioural pattern during the period of the program execution to distinguish the malware and it is most often utilized to provide fast analysis on the malwares behavioural pattern [19].

Although it has been concluded that the dynamic feature extraction technique is harder to obfuscate, research has demonstrated that it does have some drawbacks [10]. Firstly, when analysing a program using the dynamic analysis, the behavioural characteristics vary on the condition which it was executed in therefore making a single execution of reveal a section of the programs behaviour [15]. Furthermore, while using the dynamic feature extraction technique, the program is identified as being malicious software based on sensitive APIs; however, many benign applications may also invoke this APIs which could lead to the issue of false positives in the result of the analysis [11].

Shijo et al. [19] proposed using both static and the dynamic feature analysis to produce an improved result accuracy compared to using each method individually. The paper states that by reason of the continuous increase in malware samples, security vendors depend on automated malware tools. It points out that the main advantage of static analysis is that the binary code contains useful information about the malware behaviour and the main advantage of the dynamic analysis is that it analyses the runtime behaviour of a malware which is hard to obfuscate and gives a clearer result [11]. Unlike other research work which used static feature or dynamic feature for their analysis [3, 8, 19], we explore the possibility of increasing the accuracy of detection and classification by integrating both static and dynamic features of ransomware, analyse these features using machine learning algorithms aiming to achieve a higher percentage of classification while also taking note of the elements in the confusion matrix (True positive, True Negative, False positive and False negative) to observe the best algorithm with the least margin of error.

2.2 Machine Learning Techniques

Machine learning is a widely used mechanism for malware detection which is heavily reliant on the selection of features to makeup data for analysis [18]. In this research, we utilise the machine learning procedure to enable the machine to make further predictions on distinguishing malicious software from benign software [7]. In this work, the algorithms that will be used to carry out the machine learning procedure include: instance-based (IB1) [13], Random Forest (RF) [16], Naïve Bayes and Support Vector Machine (SVM) [1] [2]. Based on our research [21] [8], these algorithms help produce accurate detection.

In machine learning, there are metrics used to evaluate the effectiveness of each algorithm [14]. In this work, we use a confusion matrix [20] and Youdens index the weighted mean of these metrics are also taken into consideration since it finds the mean by assigning the weight of each element such that each element contributes to the final result based on how much importance it carries [5]. True positive (TP), false positive (FP), true negative (TN), and false negative values (FN) are used to calculate the following performance measures:

1. True Positive Rate/recall/sensitivity (*tpr*): the fraction of malware samples correctly identified as ransomware;
2. False Positive Rate (*fpr = 1 - tnr*): the fraction of goodware samples incorrectly identified as being malware;
3. True Negative Rate/specificity (*tnr*): the fraction of goodware samples correctly identified as goodware;
4. False Negative Rate (*fnr = 1 - tpr*): the fraction of ransomware samples incorrectly classified as goodware; and
5. *Accuracy* is reported as the fraction of all samples correctly identified. More specifically, $Accuracy = \frac{tpr+tnr}{tpr+tnr+fpr+fnr}$;
6. *Precision* is calculated as $precision = \frac{tp}{tp+fp}$; and
7. *Youdens index* is calculated as $Y = tpr + tnr - 1$

2.3 The Data Set

Virus Share (virusshare.com) was used to access a ransomware repository and goodware were acquired from Portable Apps (portableapps.com). Kali Linux was used for analysing of the malware samples.

In total, 400 executable files (200 malwares and 200 goodware) are used. First, the executables were put through an online intelligence platform (virustotal.com) as shown in Figure 1. The use of virtual machine was a necessity to be able to see the features of the malware and goodware properly without the risk of damaging or condemning the host machine being used for the analysis.

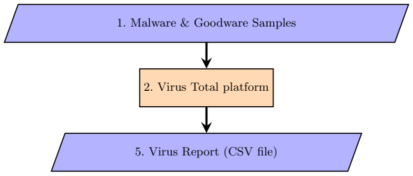

Fig. 1. Data Extraction Process overview

The virtual machine was running a default installation of Windows 10 and the networking was disabled [19]. The features extracted from the virus total platform which make up the dataset for the machine learning algorithms contained values such as detection rate, hash value, file size, dll calls, mutexes, pe info etc. which included both static and dynamic features.

Features were extracted using a script written by Didier Stevens[1] to automate the extraction of feature data from the virus total platform.

3 Experimental Analysis

The classification of the dataset was done using the WEKA which is a dedicated tool for machine learning [14]. Machine learning algorithms were selected involving the selection of appropriate classifier for the classification of the dataset into goodware and malwares, the classifiers used for this experiment are support vector machine (SVM) by sequential minimal optimization, instance-based (IB1), Naïve Bayes and Random Forest. The analysis carried out on the ransomware dataset classification was in two stages of training and testing, where each stage makes use of the above-mentioned classifiers. Normalization is a feature of WEKA which is applied to the attributes in the data pre-processed stage.

The training data set include 100 ransomware samples that are variants of Locker and Crypto Locker and also 100 goodware. The training of the machine in this stage involves using labelled data, whereas in the test stage, new and previously unseen dataset are provided for classification. The result is checked to see if it was classified accurately. The sections below show the result of the analysis in both training stage and test stage.

The training set was analysed using the WEKA classifiers which showed a perfect classification with confusion matrix and an accuracy of 100 percent which is shown in the Table 1. Furthermore, the table provides the fine-grained accuracy results necessary to understand the algorithm's capability. The confusion matrix of this analysis shows that 100 of the ransomware instances were correctly classified under the ransomware class and 100 of the goodware instances were correctly classified under the goodware class. This demonstrate the algorithms capability to correctly learn key characteristics that differentiate goodware and malware.

[1] Didier Steven's script available at: https://blog.didierstevens.com/programs/virustotal-tools/

Classifier	TP	TN	FP	FN	Youdens index	Accuracy		Precision		Recall		F-Measure	
						Training	Test	Training	Test	Training	Test	Training	Test
SVM	100	99	1	0	0.99	100	99.5	1.00	0.99	1.00	0.99	1.00	0.99
IB1	97	95	4	4	0.91	100	96	1.00	0.96	1.00	0.96	1.00	0.96
RF	100	99	1	0	0.99	100	99.5	1.00	0.99	1.00	0.99	1.00	0.99
NB	100	92	1	7	0.91	100	96	1.00	0.96	1.00	0.96	1.00	0.96

Table 1. Classifier performance on both training and test sets

After analysing the training set using each algorithm, we go on to analyse the test set using each algorithm with which the training set was analysed in the sections below. It is important to highlight that in this section we use the previously unseen 100 malware and goodware samples. The detection and classification accuracy of the WEKA classifiers were empirically evaluated to choose the machine learning algorithm with the highest detection accuracy and lowest false positive rate and the result is shown in Table 1. This test analysis was carried out using the 10-fold cross validation for its test analysis due the reputation of the K-fold cross validation method, this method allows the dataset to be iteratively analysed which gives the algorithms a better chance to be less biased, the results demonstration that the Naïve Bayes algorithm has an accuracy of 96% with the test dataset. The IB1 produced an accuracy of 96%, demonstrating that it is not the best machine learning algorithm to use for classifying this type of dataset. This is due to 4 false positive instances and 4 false negative instances from the dataset. SVM and the random forest both gave an accuracy of 99.5% which means these algorithms performed well in comparison with Naïve Bayes and IB1. The training set demonstrates a 100% detection rate and accuracy, hence it can be deduced that a larger dataset produced a better result since a larger percentage of dataseta are used for the training while the remaining was used for the test set. Furthermore, the algorithms demonstrate that regardless of the available number of datasets, they result in a higher detection rate since it performs analysis regardless of the position of the data in the dataset.

Table 1 provides a comparison of the weighted average of the recall, precision and f-measure for the different algorithms when using the testing set. It also illustrates that RF and SVM with the highest recall and f-measure and a joint highest in accuracy. This research also considered the samples that were incorrectly classified in the WEKA result in relations to the other metrics which was why we calculated the informedness of the algorithms used. Informedness also refers to the Youdens index which takes into consideration the sensitivity and specificity, which is shown in Table 1. The table illustrates that SVM and RF both had the joint highest value of 0.99 and IB1 and the Naïve Bayes both had 0.91, which when compared to the accuracy clearly gives a better representation of how the algorithms performed. It was noted that considering the correctly classified class does not give a clear picture of the efficiency of detection and classification compared to a metric which factors all elements of the confusion matrix. Algorithms with high Youdens index are considered more efficient which is why we have selected SVM and RF to be better classifiers for this research.

A number of factors could be responsible for the error in classification ranging from use of wrong classifiers to error in dataset; however, this research has been considered to have minimal error in classification because the false positive and false negative rate is below 10% of the entire class. It could be that the malware and the goodware samples are of the same sample type which means that there is a likelihood of the classifier detecting a goodware as a malware or vice-versa due to the sample size. It could also be the case that the 100 goodware and malware samples for the training set was too restrictive, reducing the classifier's ability to learn the characteristics of the training set. There is every likelihood that the classifiers need more samples to cover enough characteristics before the test sample is introduced.

4 Conclusions

Due to the increasing development and use of ransomware, there is a strong requirement for improved detection rate. This paper sets about trying to contribute to achieve this goal. This research aimed to reduce the rate of false positive in detection, which is why samples of ransomware were collected and analysed using multiple machine learning algorithms, and finally tested in WEKA. The algorithms were used to train and test the dataset to be able to better detect the ransomware family of malwares.

The random forest learning algorithm gave a relatively high detection accuracy of 99.5%, but its analysis is not reliable because it chooses the modal class and if the modal class was of the wrong class it would still have given a relatively high detection accuracy. The support vector machine globally replaces every missing value and changes nominal characteristics into a binary representation meaning it is also not reliable. The aim of this research was achieved by the random forest and SVM algorithm and the analysis carried out has a very low false positive and false negative classification. We have however concluded that the number of instances used in the classification and detection is not enough to give an overall reliable result as there is need for the algorithms use to be tested on a larger scale so as to expose the training set to a wide variant of ransomware for analysis to be able to give an accurate and reliable analysis, also, there should be inclusion of older ransomware samples so as to improve the detection rate as the ransomware samples used are from a recent collection. Future works would look into using of more features from the ransomware samples like ssdeep hash, dlls, opcodes etc. with unsupervised machine learning algorithms for classification.

References

1. A. Kumar, K.S.K., Aghila, G.: A learning model to detect maliciousness of portable executable using integrated feature set. J. King Saud Univ. - Comput. Inf. Sci. (2017)

2. A. Mohaisen, O.A., Mohaisen, M.: Amal: High-fidelity, behavior-based automated malware analysis and classification. Comput. Secur. **vol. 52**, pp. 251266 (2015)
3. Alazab, M.: Profiling and classifying the behavior of malicious codes. J. Syst. Softw. **vol. 100**, pp. 91102 (2015)
4. F. Shahzad, M.S., Farooq, M.: In-execution dynamic malware analysis and detection by mining information in process control blocks of linux os. Inf. Sci. (Ny). **vol. 231**, pp. 4563 (2013)
5. Gatz, D.F., Smith, L.: The standard error of a weighted mean concentrationi. bootstrapping vs other methods. Atmospheric Environment **29**(11), 1185–1193 (1995)
6. Grant, L., Parkinson, S.: Identifying file interaction patterns in ransomware behaviour. In: Guide to Vulnerability Analysis for Computer Networks and Systems, pp. 317–335. Springer (2018)
7. H. Lu, X. Wang, B.Z.F.W., Su, J.: Endmal: An anti-obfuscation and collaborative malware detection system using syscall sequences. Math. Comput. Model. **vol. 58**(no. 5), pp. 11401154 (2013)
8. H. Zhang, X. Xiao, F.M.S.N.F.M., Sangaiah, A.K.: Classification of ransomware families with machine learning based on n-gram of opcodes. Futur. Gener. Comput. Syst. **vol. 90**, pp. 211221 (2019)
9. Islam, R., Tian, R., Batten, L.M., Versteeg, S.: Classification of malware based on integrated static and dynamic features. Journal of Network and Computer Applications **36**(2), 646–656 (2013)
10. K. Deepa, G.R., Vinod, P.: Investigation of feature selection methods for android malware analysis. Procedia Comput. Sci. **vol. 46**, pp. 841848 (2015)
11. M. Sun, X. Li, J.C.S.L.R.T.B.M., Liang, Z.: Monet: A user-oriented behavior-based malware variants detection system for android. IEEE Trans. Inf. Forensics Secur. **vol. 12**(no. 5), pp. 11031112 (May 2017)
12. N. Milosevic, A.D., Choo, K.K.R.: Machine learning aided android malware classification. Comput. Electr. Eng. (2017)
13. P. Burnap, R. French, F.T., Jones, K.: Malware classification using self organising feature maps and machine activity data. Comput. Secur. **vol. 73**, pp. 399410 (2018)
14. Patil, T.R., Sherekar, M.S.S.: Performance analysis of naive bayes and j48 classification algorithm for data classification. Int. J. Comput. Sci. Appl. **vol. 6**(no. 2) (2013)
15. Provataki, A., Katos, V.: Differential malware forensics, digit. investig. **vol. 10**(no. 4), pp. 311322 (2013)
16. S. Das, Y. Liu, W.Z., Chandramohan, M.: Semantics-based online malware detection: Towards efficient real-time protection against malware,. IEEE Trans. Inf. Forensics Secur. **vol. 11**(no. 2), pp. 289302 (2016)
17. Schultz, M.G., Eskin, E., Zadok, F., Stolfo, S.J.: Data mining methods for detection of new malicious executables. In: Proceedings 2001 IEEE Symposium on Security and Privacy. S&P 2001. pp. 38–49. IEEE (2000)
18. Sharma, A., Sahay, S.K.: An effective approach for classification of advanced malware with high accuracy. arXiv preprint arXiv:1606.06897 (2016)
19. Shijo, P.V., Salim, A.: Integrated static and dynamic analysis for malware detection,. Procedia Comput. Sci. **vol. 46**, pp. 804–811 (2015)
20. Townsend, J.T.: Theoretical analysis of an alphabetic confusion matrix* (1971)
21. Zhang, H.: The optimality of naive bayes