Open Journal of Physical Science (OJPS)
ISSN: 2734-2123
Article Details:
DOI: 10.52417/ojps.v3i2.382
Article Ref. No.: OJPS0302002-382
Volume: 3; Issue: 2, Pages: 23- 34 (2022)
Accepted Date: September 8, 2022

Open Journals Nigeria (OJN)
Open Access | Bi-annual | Peer-reviewed
www.openjournalsnigeria.org.ng
editorial@openjournalsnigeria.org.ng

RESEARCH ARTICLE **OJPS0302002-382**

EVALUATION OF CLASSIFICATION ALGORITHMS ON LOCKY RANSOMWARE USING WEKA TOOL

Peter, F., *George, G., Mohammed, K. & Abubakar, U. B.

Department of Computer Science, Faculty of Computing and Applied Sciences, Baze University, Jabi Abuja, Nigeria

***Corresponding Author Email:** gilbert.george@bazeuniversity.edu.ng*

ABSTRACT

The ongoing danger of ransomware has led to a struggle between creating and identifying novel approaches. Although detection and mitigation systems have been created and are used widely, they are always evolving and being updated due to their reactive nature. This is because harmful code and its behavior can frequently be altered to evade detection methods. In this study, we present a classification method that combines static and dynamic data to improve the precision of locky ransomware detection and classification. We trained supervised machine learning algorithms using cross-validation and used a confusion matrix to observe accuracy, enabling a systematic comparison of each algorithm. In this work, supervised algorithms such as the decision tree algorithm resulted in an accuracy of 97%, naïve baiyes 95%, random tree 63%, and ZeorR 50%.

Keywords: *Ransomware, Malware, Machine Learning, Naïve baiyes, Random tree, ZeorR*

INTRODUCTION

The first known ransomware attack was recorded to have been created by Joseph Popp in 1986 (Shijo & Salim, 2015). It was distributed via a floppy disk, which was handed to participants at a conference on AIDS disease. The malicious codes encrypted just the filenames and demanded a ransom to be paid to an account in Panama, but the fix was thwarted by a quick fix, thereby rendering the attack unsuccessful. However, with time, the malware programs evolved and found ways to carry out zero-day exploits. The idea of file encryption ransomware was developed by Moti Yung and A. Young was presented at the 1996 IEEE security and privacy conference(Nath & Mehtre, 2014). It was called "crytoviral extortion." The key feature of ransomware is the ability to retrieve the ransom successfully and leave no trace for forensic investigation. Young and Yung introduced the notion of using public key cryptography for encrypting ransomed data(Adamu et al., 2019). They criticized the failed AIDS ransomware attack and suggested the reason for the failure was its use of symmetric cryptography alone. This idea has led to an evolution process in which malware developers develop ransomware(Sgandurra *et al.*, 2016).

Most ransomware is distributed via emails, by embedding the malicious codes in the authentic files (Egunjobi *et al.*, 2019a), the unsuspecting victim opens the malicious files, triggering the Autoexec function,(Cabaj et al., 2016.) and the malicious codes then generate a random symmetric key and encrypt the victim's files. It uses a public key embedded in the malware to encrypt the symmetric key (Egunjobi *et al.*, 2019a). This results in asymmetric and symmetric cipher text of the victim's files. It zeroes the symmetric key value to avoid data recovery. It sets up a message to the user with the asymmetric cipher text and how to pay the money. If successful, the victim has no choice but to send the E-money by sending the asymmetric cipher text alongside the e-money. The attacker receives the payment and the asymmetric cipher text and then deciphers it with a private key, sends the symmetric key to the victim, and uses it to decrypt the files. Many governmental and private organizations have been attacked by the use of ransomware. Most recently was the "WannaCry" ransomware that hit over 20 countries in May 2017(Egunjobi *et al.*, 2019b) and just when the world thought it was over, came "Petya" on June 27, 2017. Patches to counteract these malicious programs are not adequately deployed, leaving a lot of room for damage (Egunjobi *et al.*, 2019b).

The current detection solutions are insufficient to tackle the rising security threats and challenges(Egunjobi *et al.*, 2019b). This brings up the big question; how do we develop or improve the current systems to detect these malicious files and prevent them from being executed in the first place?

The approach taken to solve the problem is based on a previous application of machine learning on malware features to improve malware detection. This study aims to detect threats before they can get executed. This will be done by using specific machine learning algorithms to decide based on the data passed to them. We propose to explore the possibility of increasing the accuracy of detection and classification by using both static and dynamic features of the Locky ransomware family, training machine learning algorithms, and introducing a test set with the aim to achieve a higher percentage of classification.

The study is organized as follows: First, we examine relevant work, and then we describe the suggested technique in the next section. The research process for obtaining empirical insight is then presented. The presentation of the results

is then followed by a description of the samples utilized and the categorization process. Finally, a summary and suggestions for other research areas are given.

BACKGROUND OF LITERATURE

In this section, enlightenment is provided to the reader about different ways of malware analysis, the advancements and limitations have done in the area of study (Shijo & Salim, 2015), explanations will be given for the different existing types of malware analysis, and point out their limitations. We will go further to explain the need for our research in the field.

Static Analysis

Static analysis is the use of certain features extracted from malicious software, these features include program header information, binary strings, and source codes extracted from the malware (Belaoued & Mazouzi, 2015)-(Nath & Mehtre, 2014). The particular software can extract the features, it decompiles the application back to codes. These can be done without executing the codes (Nath & Mehtre, 2014). When the features are extracted, they will have to be examined by forensic investigators that will keep an eye out for abnormalities in the codes(Milosevic *et al.*, 2017). This analysis is effective due to the careful examination done(Milosevic *et al.*, 2017), however, it is time-consuming (Milosevic *et al.*, 2017), and obfuscation of the malicious programs by malware developers makes it hard to detect abnormality (Chen *et al.*, 2017; Sharma & Sahay, 2016) and leaves room for human error. Because of these problems mentioned above, static analysis is rarely used for analysis.

Dynamic analysis

Dynamic analysis involves the investigation of malware performed by running the malicious code in a sandbox (Shijo & Salim, 2015) (virtual environment). This process allows the examiners to monitor the API calls made by the malicious application to identify the features and behaviours of the malware. During these analyses, procedures are put into place to study the behaviour of executed malware codes containing the binary occurrences and function calls (Sahay & Sharma, 2016).

The limitation of dynamic analysis is that most times malware programmers sometimes modify codes to detect sandboxes and this will result in an inaccurate analysis of the malware characteristics (Afonso *et al.*, 2015). Feature extraction can be a whole lot wearisome and tiring and a large amount of time can be spent on analysing just one malware, which makes it a very unsuitable method for analyzing a large data set of malware (Milosevic *et al.*, 2017).

Machine Learning Practices

There have been several similar works done using machine language to detect malware. The first of this goes back to 2001, when a group of academic investigators (Schultz *et al.*, 2001.), proposed that it is a better method for malware detection, rather than the old signature-based detection techniques. In the effort to prove it, 4301 applications were examined, 3301 of these applications were deemed malicious and 1000 of them were benign files. Binary profile files, string sequences, and hex dumps were used in the experiments. Many more works followed after this, such as a group of researchers going on to use the n-gram attributes obtained from the binary files added to the classifier, which yielded

a high success ratio of 98% (Gardiner & Nagaraja, 2016). Over time different features have been used by these classifiers to yield greater accuracy (Gardiner & Nagaraja, 2016). The classifiers require a number of samples to yield higher accuracy (Gardiner & Nagaraja, 2016). Generating all the data is time-consuming. Within the machine learning techniques, several suggestions have been brought forward to fix the issue, which involves the use of multi classifiers approach (Sajedi & Allameh, 2017). This aimed at getting the best true positive ratio.

Related works

Egunjobi *et al.* (2019b) used WEKA, similarly to our method, the authors present a classification method that combines static and dynamic information to improve the precision of ransomware detection and classification. They utilize a test set to train supervised machine learning algorithms and used a confusion matrix to track accuracy. This allows the authors to compare different methods in-depth. The supervised algorithms they used included the I Bayes algorithm which had a test set accuracy of 96%, the SVM algorithm with an accuracy of 99.5%, random forest accuracy of 99.5%, and an accuracy of 96%. To calculate sensitivity and specificity, we also employ Youden's index.(Al-rimy et al., 2018) provided a thorough overview of various ransomware studies. The report made some recommendations for preventing ransomware outbreaks. An early-stage ransomware attack could have been prevented with the help of a situational awareness model that was presented. The ransomware literature that is currently available was categorized based on the review, proposal, testing, detection, and preventive research. It gave a summary by taking into account a few variables, a dataset, and the technologies used in the ransomware investigations.

Khammas (2022) reported different machine-learning techniques were evaluated to analyse their ability in the detection of ransomware attacks. The top one thousand features were extracted from raw bytes with the use of a gain ratio as a feature selection method. Three different classifiers (decision tree J48), random forest, radial basis function (RBF network) available in Waikato Environment for Knowledge Analysis (WEKA) based machine learning tool are evaluated to achieve significant detection accuracy of ransomware. The result shows that random forest gave the best detection accuracy almost around 98%.

Singh (2022) collected some malicious and self-generated benign PCAP's and then applied a suitable machine learning classification algorithm to build a traffic classifier. The proposed classifier classifies the malicious HTTPs traffic. The experimental results show the average accuracy (90%) and false-positive (0.030) for Random Forest (RF) classifier.

Hairil *et al.* (2021) proposed a design that was built in the form of a ransomware detection system based on available bitcoin heist data to minimize hacking attacks against cryptocurrency in the future. The ransomware detection system was built using the backpropagation artificial neural network method using Weka software. The best results in data testing are using the parameter number of the hidden layer with 9 neurons; a learning rate of 0.1; and the number of iterations of 5000 yielded an accuracy rate of 97%.

MATERIALS AND METHODS

Table 1 shows a breakdown of the dataset used in this study. It also shows the processes taken to obtain the results. We used a dataset that comprises 50 benign ware and 50 locky ransomware samples. These samples were obtained

from ransomtracker.com and portableapps.com. We then ran the malware samples against the virus total intelligence platform to retrieve relevant data on the behaviors exhibited by these malwares.

The benign hashes which were obtained from portableapps.com were passed through the virus total intelligence framework which yielded no detection whatsoever from any antivirus software hosted on the virus total framework.

The machine models were simulated on our data set using WEKA on an OSX operating system, 8.00GB RAM, and a Dual Core i5 2.67GHz processor.

Table 1: Details of the Malware Dataset.

Class	Type	Total
Locky(0)	Win32	50
Benignware(1)	Win32	50

Static features extractions

As mentioned above, the malicious hashes were run individually on the virus total intelligence platform and analysis reports gotten from the site were used to determine the relevant features needed in this study. The classification of the features was based on the results obtained from static analysis, such as the strings found in the file binary codes. For all the samples in the dataset, we examined the extracted strings and identified the recurring strings in the majority of the hashes both in malware and benign ware.

However, there was a string only found in malware hashes which was the string ".rdata" while in the benign ware the ".ndata". The information gotten from this was passed to the machine learning classifiers for automated classification.

Table 2: Portable Execution Section.

Static Features	Found
.reloc	True
.rdata	True
.data	True
Ndata	False
.rsrc	True
.Text	True
.Tex2t	True

Dynamic Feature Extraction

There were also some dynamic features extracted from the malware hashes and used for the classification process, they include specific (Dynamic Link Library) dll files called by the ransomware in a live environment. Most of all the hashes examined had the following dll files called and sometimes executed.

Table 3: Dynamic Link Libraries Files Called by the Malware in Runtime.

DLL	No. of Appearance
ADVAPI32.dll	55/100
GDI32.dll	86/100
KERNRL32.dll	28/100
PSAPI.dll	45/100
SHELL32.dll	90/100
SHLWAPI.dll	55/100
USER32.dll	66/100
VERSION.dll	61/100
WTSAPI32.dll	52/100
msvcrt.dll	86/100
ole32.dll	91/100

Integrated Features

This section involves combining the features from both dynamic and static analysis(Shijo & Salim, 2015). It provides explanations behind the combination of the features selected for the research. Using the unique strings in both malware and benign hashes, we were able to train the machine to differentiate between malware and benign.

Machine Learning Techniques

The choice of features to include in the data that will be subjected to analysis when using machine learning to detect malware is crucial (Singh, 2022). In this study, we use machine learning to help the computer make more predictions about how to tell harmful software from benign software. In this study, the Decision Tree (Che *et al.*, 2011), Random Forest (RF) (Egunjobi *et al.*, 2019a), Nave Bayes, and ZeroR (Egunjobi *et al.*, 2019b; Mitchell, 2010) algorithms will be employed to carry out the machine learning procedure. Our study suggests that these methods aid in accurate detection.

Cross Validation

Ten "folds were randomly selected from the training dataset of the chosen images. This prevented bias or overfitting while performing 10-fold cross-validation on the model training." A "validation set that was completely different from the other training folds was chosen to assess the training state throughout training." Once one model training phase was complete, the other independent fold was utilized as a validation set, and the previous validation set was recycled as part of the training set to evaluate the model training.

RESULTS AND DISCUSSION

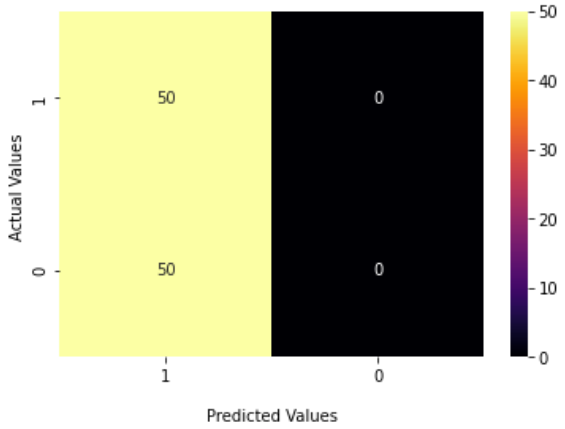

Figure 1: Confusion Matrix for Random Tree Classifier

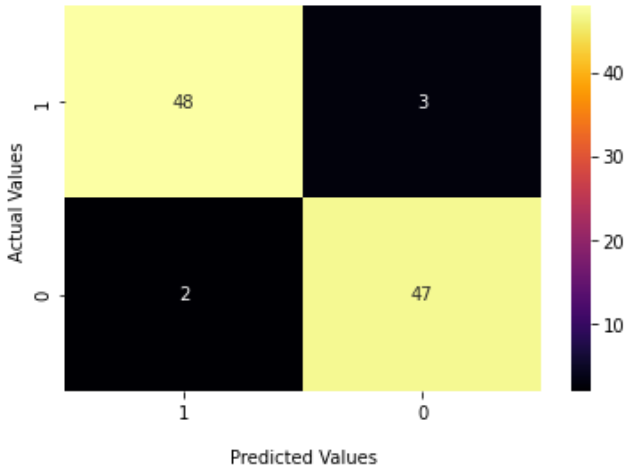

Figure 2: Confusion Matrix for Random Naïve Baiyes

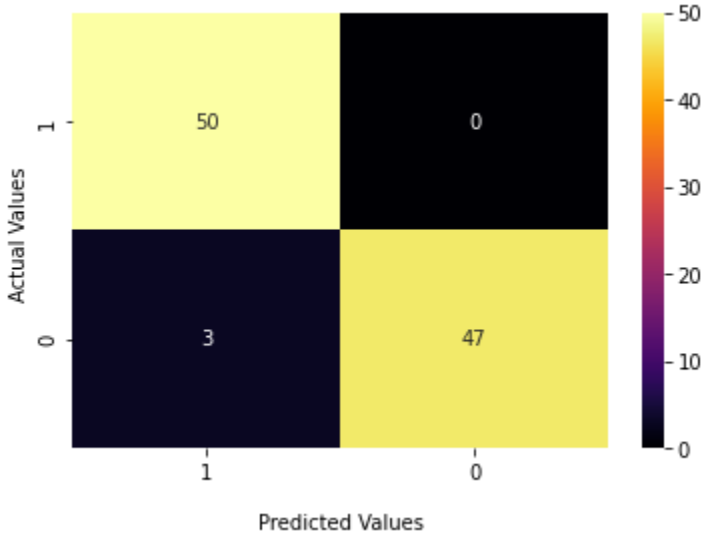

Figure 3: Confusion Matrix for Decision Tree

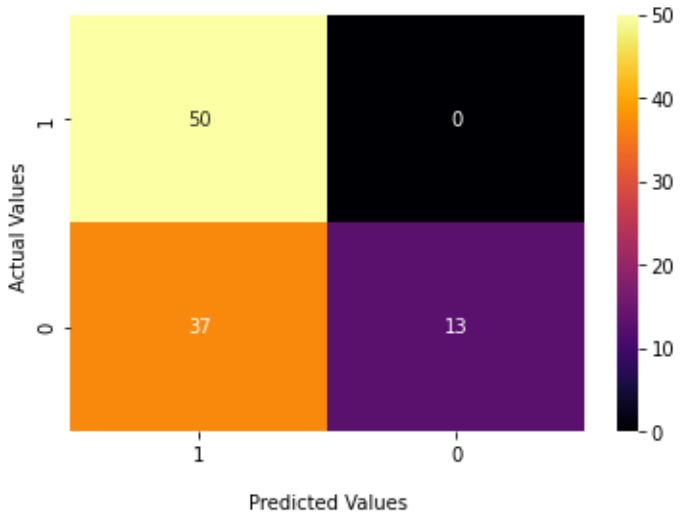

Figure 4: Confusion Matrix for Random Tree

Figure 5: Accuracy Graph for ZeroR classifier

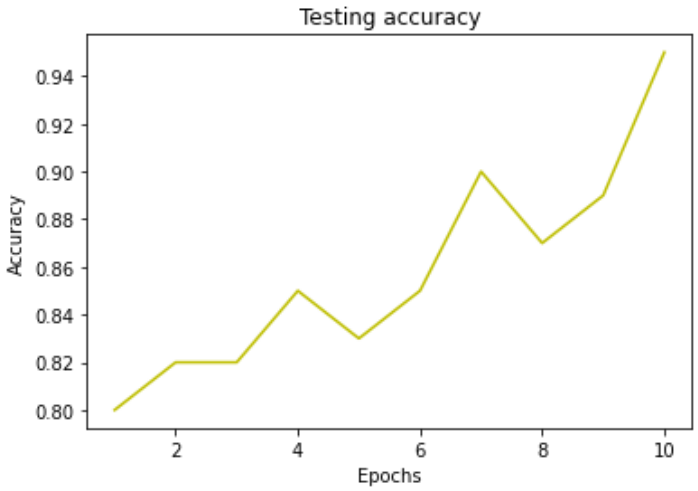

Figure 6: Accuracy Graph for Naïve Baiyes classifier

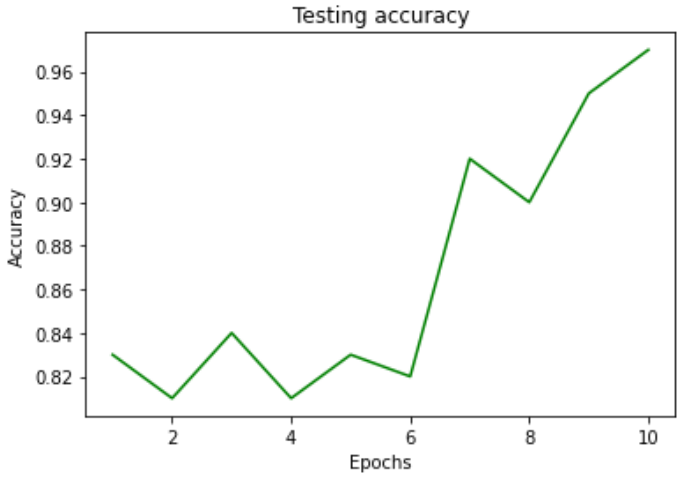

Figure 7: Accuracy Graph for Decision Trees classifier

Figure 8: Accuracy Graph for Random Tree classifier

Prediction Conference

Table 3: Metric Table

	Accuracy	**Recall**	**Specificity**	**Precsion**	**F1**
Naïve Baiyes	0.95	0.95	0.96	0.95	0.95
Decision Tree	0.97	0.97	0.94	0.97	0.97
Random Tree	0.63	0.63	0.26	0.79	0.57
ZeroR	0.50	1.0	0.00	0.50	0.67

Fifty (50) malware and 50 benign samples were used for the classification using WEKA classifier detection and classification precision was empirically assessed to determine the machine learning method with the lowest false discovery rate and highest detection accuracy. From table 3 we can see the displayed precision outcome and the accuracy rate for each classifier, which includes the specificity and F1 score. From this, it clearly shows that the decision tree outperforms the rest of the classifiers with an accuracy of 97%. Because of the reputation of the K-fold cross-validation approach, this test analysis was performed using the 10-fold cross-validation method, which allows the dataset to be iteratively studied and increases the likelihood that the algorithms would be less biased as seen in the graphs figures 5-8 above. The weighted average of the recall, precision, and f-measure for the various methods while utilizing the testing set is compared in Table 1. Additionally, it shows that DT and NB have the maximum accuracy, f-measure, and recall.

CONCLUSION

There is a critical need for an increased detection rate because ransomware is being developed and used more frequently. This paper aims to contribute to achieving this objective. To lower the rate of a false positive detection, ransomware samples were gathered, examined using a variety of machine learning methods, and then tested in WEKA. To improve the classification model's ability to detect the ransomware family of malware, notably the locky family, algorithms were employed to train and test it. The decision tree learning technique provided a 97% detection accuracy, which is a respectable level. We have determined, however, that the number of instances used in the classification and detection is insufficient to provide an overall reliable result, as the algorithms must be tested on a larger scale to expose the training set to a variety of ransomware for analysis, as well as the inclusion of other ransomware samples to improve the detection rate as the ransomware samples used are not all equally dangerous. Future research will examine the use of additional ransomware sample attributes, such as the ssdeep hash, dlls, and opcodes, with unsupervised machine learning techniques for categorization.

CONFLICT-OF-INTEREST

There was no conflict of interest from all the authors contributing to this article

REFERENCE

Adamu, Umaru & Awan, Irfan. (2019). Ransomware Prediction Using Supervised Learning Algorithms. 57-63. https://doi.org/10.1109/FiCloud.2019.00016.

Afonso, V. M., Favero De Amorim, M., André, ·, Abed, R., Glauco, G. ·, Junquera, B., Lício De Geus, P., Afonso, V. M., De Amorim, M. F., Grégio, A. R. A., De Geus, P. L., & Junquera, G. B. (2015). Identifying Android malware using dynamically obtained features. *Journal of Computer Virology and Hacking Techniques* **11**: 9–17. https://doi.org/10.1007/s11416-014-0226-7

Al-rimy, B. A. S., Maarof, M. A., & Shaid, S. Z. M. (2018). Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions. *Computers & Security*, **74:**144–166. https://doi.org/10.1016/j.cose.2018.01.001

Belaoued, M., & Mazouzi, S. (2015). A real-time pe-malware detection system based on chi-square test and pe-file features. *IFIP International Conference on Computer Science and Its Applications*, 416–425.

Cabaj, K., Gregorczyk, M., & Mazurczyk, W. (2018). Software-Defined Networking-based Crypto Ransomware Detection Using HTTP Traffic Characteristics. Comput. Electr. Eng., **66**:353-368.

Che, D., Liu, Q., Rasheed, K., & Tao, X. (2011). Decision Tree and Ensemble Learning Algorithms with Their Applications in Bioinformatics. *Advances in Experimental Medicine and Biology*, **696**:191–199. https://doi.org/10.1007/978-1-4419-7046-6_19.

Chen, R., Niu, W., Zhang, X., Zhuo, Z., & Lv, F. (2017). An Effective Conversation-Based Botnet Detection Method. *Mathematical Problems in Engineering*, 1–9. https://doi.org/10.1155/2017/4934082.

Egunjobi, S., Parkinson, S., & Crampton, A. (2019a). Classifying Ransomware Using Machine Learning Algorithms. In H. Yin, D. Camacho, P. Tino, A. J. Tallón-Ballesteros, R. Menezes, & R. Allmendinger (Eds.), *Intelligent Data Engineering and Automated Learning – IDEAL 2019* (pp. 45–52). Springer International Publishing. https://doi.org/10.1007/978-3-030-33617-2_5.

Egunjobi, S., Parkinson, S., & Crampton, A. (2019b). Classifying Ransomware Using Machine Learning Algorithms. In H. Yin, D. Camacho, P. Tino, A. J. Tallón-Ballesteros, R. Menezes, & R. Allmendinger (Eds.), *Intelligent Data Engineering and Automated Learning – IDEAL 2019* (pp. 45–52). Springer International Publishing. https://doi.org/10.1007/978-3-030-33617-2_5.

Gardiner, J., & Nagaraja, S. (2016). On the Security of Machine Learning in Malware C8C Detection. *ACM Computing Surveys*, **49**(3):1–39. https://doi.org/10.1145/3003816.

Hairil, Cahyani, N. D. W., & Nuha, H. H. (2021). Ransomware Detection on Bitcoin Transactions Using Artificial Neural Network Methods. *2021 9th International Conference on Information and Communication Technology (ICoICT)*, 1–5. https://doi.org/10.1109/ICoICT52021.2021.9527414.

Khammas, B. M. (2022). Comparative analysis of various machine learning algorithms for ransomware detection. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, **20**(1):43–51. https://doi.org/10.12928/telkomnika.v20i1.18812.

Milosevic, N., Dehghantanha, A., & Choo, K.-K. R. (2017). Machine learning aided Android malware classification. *Computers & Electrical Engineering*. https://doi.org/10.1016/j.compeleceng.2017.02.013.

Mitchell, T. M. (2010). CHAPTER 1 GENERATIVE AND DISCRIMINATIVE CLASSIFIERS : NAIVE BAYES AND LOGISTIC REGRESSION Learning Classifiers based on Bayes Rule. *Machine Learning*, *1*(Pt 1-2), 1–17.

Nath, H. V., & Mehtre, B. M. (2014). *Static Malware Analysis Using Machine Learning Methods* (pp. 440–450). https://doi.org/10.1007/978-3-642-54525-2_39.

Sahay, S. K., & Sharma, A. (2016). Grouping the Executables to Detect Malwares with High Accuracy. *Procedia Computer Science*, *78*, 667–674. https://doi.org/10.1016/j.procs.2016.02.115.

Sajedi, H., & Allameh, F. (2017). Detection of malicious web pages by evolutionary ensemble learning. *International Journal of Hybrid Intelligent Systems*, *13*(3–4), 151–159. https://doi.org/10.3233/HIS-160232.

Schultz, M. G., Eskin, E., Zadok, F., & Stolfo, S. J. (2001). Data mining methods for detection of new malicious executables. *Proceedings 2001 IEEE Symposium on Security and Privacy. S&P*, 38–49. https://doi.org/10.1109/SECPRI.2001.924286.

Sgandurra, D., Muñoz-González, L., arXiv … R. M., & undefined 2016. (2016). Automated dynamic analysis of ransomware: Benefits, limitations and use for detection. *Arxiv.Org*. https://arxiv.org/abs/1609.03020.

Sharma, A., & Sahay, S. K. (2016). An Effective Approach for Classification of Advanced Malware with High Accuracy. *International Journal of Security and Its Applications*, **10**(4):249–266. https://doi.org/10.14257/ijsia.2016.10.4.24

Shijo, P. V., & Salim, A. (2015). Integrated Static and Dynamic Analysis for Malware Detection. *Procedia Computer Science*, 46:804–811. https://doi.org/10.1016/j.procs.2015.02.149

Singh, A. (2022). Classification of Malware in HTTPs Traffic Using Machine Learning Approach. *El-Cezeri*, *9*(2), 644–655. https://doi.org/10.31202/ecjse.990318