Obfuscated Ransomware Family Classification Using Machine Learning

William Cassel, Nahid Ebrahimi Majd
Department of Computer Science and Information Systems
California State University San Marcos, United States
Casse017@csusm.edu, nmajd@csusm.edu

***Abstract*— The recent rise of ransomware attacks, average ransom demands, average ransom payments, and average ransomware recovery time has made ransomware a serious threat for businesses and individuals. Obfuscated ransomware is a more threatening variation that is more complicated to detect. Designing accurate ransomware detection systems is essential to protect networks from harmful consequences of a ransomware attack. In this research, we propose a machine learning based ransomware classification framework and study five machine learning algorithms and four feature selection techniques to detect the class of an obfuscated ransomware vs. benign. We studied different feature selection techniques that remove noise and highly correlated features to get the most efficient model. We also studied the impacts of different techniques to combat data imbalance. Our results indicate that Random Forest with LightGBM feature selection technique outperforms other models with 89.4% accuracy.**

Keywords*— *Obfuscated Ransomware Classification; Network security; Feature Selection; Machine Learning;

I. Introduction

Ransomware is a type of malware cyber-attack that gets the control of the machine's file system, encrypts the files, and blocks the user's access to the files [1]. Then, it provides instructions for the user to pay a ransom in cryptocurrency to provide the key to decrypt the files. In most cases, even if the victim pays the ransom, the attacker will not provide the decryption key to recover the files, and there is no way that the victim can track the attacker as cryptocurrency accounts are anonymous, and their owners could not be tracked or blocked.

Since COVID pandemic where a large portion of Internet users needed to work virtually, the volume of data storage on business and individual machines has significantly increased. Data is the most vulnerable asset for most businesses and individuals, and ransomware attacks directly target the victims' file systems [2]. Thus, ransomware has increasingly become a profitable attack for cyber attackers. In the first quarter of 2023, not only the number of ransomware attacks but also the amount of demanded ransom has significantly increased comparing to the last year. Besides that, with the aid of new AI technologies, new types of ransomware attacks have been created by cyber-attackers and are being circulated in networks, which require longer recovery times. All of these impose severe financial costs and data losses to businesses and individuals [3].

With the continuing rise of ransom demands, average ransom payments, average recovery times, and the number of ransomware-related lawsuits in 2023, there is an immediate need to detect ransomware attacks, block them from infecting the network or mitigate their threats [4]. Machine Learning (ML) is a promising solution to combat ransomware [5]. In this paper, we propose a ransomware family classification framework that can effectively detect benign vs. the type of ransomware attack.

Different classes of ransomware conduct the attack in different ways. They mainly encrypt the files, block the user's access to the files, and provide instructions to pay a ransom. Each class of ransomware follows a different procedure to encrypts the files and limit the access to files. Some ransomwares first copy the files to memory, then encrypt them, then store the encrypted files on the disk, and then delete the original files; some other ransomwares, delete the original files before they store the encrypted files on the disk; etc. Detecting the type of attack helps network administrators take appropriate precautionary measures to prevent the attack or mitigate its disruptive impacts [6].

Malware obfuscation is the act of reforming the malware code in a way that it is harder for anti-malware systems to be detected, but it still successfully conducts the attack. The goal is to hide the malware operation from anti-malwares. Some of the most common malware obfuscation techniques are the code compression, encryption, and encoding. *Ransomware obfuscation* is usually done by compiling after delivery. In this technique, the victim receives the ransomware source code in a spam email. The victim undesirably will activate the code, which will call a local compiler such as csc.exe to compile its payload on the machine. This will bypass firewalls and intrusion detection systems that are usually at the network edge. Then, the complied ransomware conducts the attack on the machine, gets the control of file system, and encrypts the files [7]. Detecting obfuscated malwares is more complex than regular malwares, and requires a system specifically designed to detect these types of attacks. Using machine learning in obfuscated malware detection is relatively new as there are not many obfuscated malware datasets available to build ML models [8]. Recently, CIC-MalMem2022 dataset has been created using real network scenarios. This dataset includes five different obfuscated classes of ransomware, and it is a perfect choice to design an obfuscated ransomware classifier. We use this dataset in our research.

The main contributions of this research are we applied four feature selection techniques, namely ANOVA, Light GBM, ReliefF, and Chi-square, to reduce the noise and remove highly correlated features to achieve the most accurate classifiers. We applied five machine learning algorithms to the reduced datasets. We used Naive Bayes (NB), Linear Discriminant Analysis (LDA), Extreme Gradient Boosting (XGB), Extra Tree (ET), and Random Forest (RF) machine learning algorithms. We studied these models in extensive experiments with different numbers of features. The results indicated our RF model with

2769-5654/23/$31.00 ©2023 IEEE
DOI 10.1109/CSCI62032.2023.00134

LightGBM feature selection outperforms the other models in terms of accuracy and train and test times.

The rest of this paper is organized as the following. Section 2 describes the related work. Section 3 explains the methodology, including the dataset and preprocessing. Section 4 describes the six studied feature selection techniques. Section 5 explains hyperparameter tunning. Section 6 presents the results and discussion. Section 7 draws the conclusion.

II. Related Work

Most research on ML-based ransomware classification has studied binary classification. We review the most recent research in this field. [9] proposed a ransomware detection method that extracts features from raw bytes of executable file, and then feeds the extracted data to a random forest binary classifier. They achieved a high accuracy of (97.74%), high ROC of 99.6%, low FPR (around 0.04), and low FNR (around 0.002) in only 1.37 second time of detection. [10] proposed multi-level ML models to analyze different sections of the ransomware code. Their proposed models achieved accuracies ranging from 76% to 97%. [11] and [12] also proposed ML binary classifiers to detect ransomware vs. benign files. [13] proposed a different approach to select dynamic features that are most relevant in identifying Encryptor and Locker ransomware classes. Their approach achieved an accuracy of 99%. [14] proposed DeepRan, a deep learning model that used a bi-directional Long Short-Term Memory (BiLSTM) model and a fully connected (FC) layer. They used this model to identify ransomware anomalies. Their model achieved a high accuracy of 99.87% and F1-score of 99.02%. [15] studied the impact of filter feature selection on a variety of ML binary classifiers to detect ransomware. Their results indicated that their random forest classifier outperformed the other methods in terms of accuracy, F-beta, and precision scores.

III. Methodology

A. Dataset

We used the CIC-MalMem-2022 dataset [16] for our research because it is the most recent dataset that contains data captured from obfuscated malware in real scenarios. This dataset is publicly available and has a substantial number of instances for different ransomware attacks as well as well-defined features to effectively train and test models. This dataset is large and contains data on three types of malwares: Ransomware, Spyware and Trojan attacks. In this research we focus on ransomware and use only ransomware and benign instances. This dataset has been created by extracting features from memory dumps using VolMemLyzer tool [17]. In this dataset, each record has 56 features: 54 numerical features, and 2 nominal features that describe the record's class and subclass. There are five subclasses of Ransomware: (1) Shade, (2) Ako, (3) Conti, (4) Maze, and (5) Pysa. *Our problem* is to develop a family classification model that classifies a dataset record to either benign or one of the five ransomware classes. We split the dataset to 80% training and 20% test sets. Table 1 presents the breakdown of the number of instances for benign and each ransomware attack in our dataset.

Table 1: the dataset breakdown

Class	Count	Percentage
Benign	29,298	75%
Ransomware-Shade	2,128	5.44%
Ransomware-Ako	2,000	5.12%
Ransomware-Conti	1,988	5.09%
Ransomware-Maze	1,958	5.01%
Ransomware-Pysa	1,717	4.39%
Total	39,089	100%

B. Data Standardization

We applied the standard scholar method from scikit learn library to standardize data.

IV. Feature Selection Technique

We employed 4 different feature selection techniques on the dataset and studied their impact on the models' accuracies.

A. ANOVA

Analysis of Variance (ANOVA) is a feature selection algorithm that identifies which features have the most significant impact on the target variable via comparing the variance values. If the variance of a certain feature is high within a certain class; then it is likely that feature is useful for identifying the differences between records in that class. ANOVA then calculates the mean of the variance for each given feature and uses this value to compute the features F-values. We sorted the features based on their ANOVA F-values and selected the features with highest F-values.

B. LGBM

LightGBM (LGBM) is a feature selection method that utilizes a vertical ensemble decision tree based algorithm. LGBM distributes the data in a histogram-based distribution using GOSS (Gradient-based One Side Sampling) to subsample the dataset. The benefit of using GOSS over random sampling is the fact that it removes samples with smaller gradients, meaning the model is less susceptible to issues such as overfitting. Categorical data must be converted to numerical data for this algorithm, so in our case, the classes were converted to integers using the label_encoder method from sklearn.

C. ReliefF

ReliefF performs feature selection by calculating a value for each feature that is then used as an estimator for that features ability to describe a class. The algorithm runs in cycles working with one record at a time where it finds the record's nearest neighbors and then calculates the distance between them using the feature values for that record. Then, the feature weights are updated and the cycle restarts from another record until all records have been iterated through.

D. Chi-square

The Chi-Square test measures the independence between two features using a contingency table which contains the distribution of two variables. This table is then used to calculate the Chi-Square value for each feature, which will be used to select the most relevant features.

We applied the above four feature selection techniques to train the dataset and fed the transformed data to five ML algorithms, consisting of (1) Naive Bayes (NB), Linear Discriminant Analysis (LDA), Extreme Gradient Boosting (XGB), Extra Tree (ET), and Random Forest (RF). Then, we trained the models with different numbers of features for each feature selection method. Our studies showed that the highest accuracy is obtained when the number of features is in range [20-30]. For most feature selection techniques, increasing the number of features in range [20-30] did not significantly improve the accuracy, but increased the training and prediction times, which is not desired. Selecting less than 20 features declined the accuracies. According to our results, we concluded 20 is the best number of features for almost all models and feature selection techniques for this dataset, which results in highest accuracy and lowest training and prediction times. In this paper, we present the results of models trained by *20 features*.

Table 2 lists the top 20 features selected by each feature selection technique. We observe that these techniques share some features. For instance, features 4, 8, 9, 15, 18, 20 are selected by all these techniques, meaning they all have high associations with the label, which is benign or ransomware class.

V. Hyperparameter Tunning

We used five supervised ML algorithms to create five series of models, one for each of the four studied feature selection methods and one for no feature selection. We tunned hyperparameters for each model when no feature selection method was employed. We used grid search and 5-fold cross-validation. The tunned hyperparameters are listed in Table 3. We used the same tunned hyperparameters for the models that use feature selection methods as well.

Table 2: Top 20 features selected by each feature selection method.

Feature selection	Selected feature numbers
ANOVA	3, 4, 7, 8, 13, 15, 16, 18, 19, 20, 21, 22, 24, 28, 29, 30, 31, 47, 50, 51
LGBM	4, 6, 7, 8, 10, 12, 15, 16, 18, 20, 21, 25, 26, 28, 29, 31, 47, 48, 51, 54
ReliefF	3, 4, 6, 7, 8, 10, 13, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 48, 50, 51
Chi-square	4, 7, 8, 10, 14, 15, 17, 18, 20, 21, 22, 23, 24, 25, 26, 30, 31, 32, 33, 36

Table 3: ML models with their Hyperparameters.

No	Model	Hyperparameter
1	NB	priors = none, var_smoothing = 0.000000001
2	LDA	solver= lsqr, shrinkage = None
3	XGB	n_estimators = 150, learning_rate = 0.1, max_depth =3, min_samples_leaf = 3
4	ET	n_estimators = 300, criterion = "entropy", min_samples_leaf = 1
5	RF	n_estimators = 100, criterion = "entropy", min_samples_leaf = 1, mine_samples_split =2

VI. Results and Discussion

As discussed earlier, we studied different number of features, and our models presented the highest accuracies and lowest training and prediction times with 20 features. Table 4 illustrates the results for the five studied machine learning algorithms and the four studied feature selection techniques where the models are built by 20 features. It also presents the results with no feature selection, which we use as a basis to investigate the efficiency of feature selection employment on this dataset. Fig. 1 presents a summary of accuracies for all models.

Among the five ML algorithms, RF and ET present the highest accuracies, and then XGB, LDA, and NB, in order, present less accuracies for almost all feature selection methods. Between RF and ET, RF models present higher or comparable accuracies while requiring less time for training and prediction. Especially, RF models require significantly less prediction times comparing to ET models, which makes RF a suitable model for time-sensitive and large-scale classification applications.

The best result is achieved by RF with LGBM feature selection with 89.4% accuracy. RF with ANOVA (89.3% accuracy) and Chi-square (89.2% accuracy) are the next best models for this dataset. These three have the same prediction time and comparable training times.

Among different feature selection methods, LGBM, ANOVA, and Chi-square present the highest accuracies, while LGBM presents slightly higher accuracy than ANOVA and Chi-square for almost all models. ReliefF and no feature selection present less but comparable accuracies.

For RF, the accuracies of models with LGBM, ANOVA, Chi-square, ReliefF, and no feature selection are 89.4%, 89.3%, 89.2%, 88.7%, and 88.3% respectively. We observe that LGBM, the best feature selection method, improves the accuracy by 1.1% comparing to the model that uses no feature selection technique. However, it should be considered that the feature selection techniques reduce the number of features from 54 to 20 features. That significantly reduces the training and prediction times. For RF, the training time with no feature selection is 69.78 sec, which is almost 12 times more than LGBM feature selection with 4.56 sec training time.

Overall, the feature selection techniques on this dataset improve the accuracy and significantly reduce the training and prediction times. This shows the importance of employing appropriate feature selection techniques in ML models.

Fig. 2 demonstrates the ROC charts for the five models using LGBM feature selection technique. This chart indicate that RF and ET present the best performances with AUCs of 0.969 and 0.967.

Our RF with LGBM presents the best result. We investigated the top 20 features selected by LGBM with more details to figure out what features have the highest impact on classifying obfuscated ransomware. Table 5 presents the top 20 features that were selected by LGBM to create the models. In this table, the features are sorted based on the scores that LGBM calculated for each feature.

Table 4: The results of the proposed models with 20 features

Feature selection	No	Model	Accuracy	Training time	Prediction time
ANOVA	1	NB	0.804	0.08	0.01
	2	LDA	0.832	0.24	0.01
	3	XGB	0.877	82.77	0.08
	4	ET	0.890	5.99	0.48
	5	RF	0.893	3.59	0.12
LGBM	6	NB	0.809	0.09	0.01
	7	LDA	0.825	0.24	0.01
	8	XGB	0.878	109.05	0.08
	9	ET	0.891	6.1	0.46
	10	**RF**	**0.894**	**4.56**	**0.12**
ReliefF	11	NB	0.796	0.07	0.01
	12	LDA	0.821	0.2	0.01
	13	XGB	0.870	107.74	0.08
	14	ET	0.887	5.85	0.44
	15	RF	0.885	4.85	0.12
Chi-square	16	NB	0.826	0.07	0.01
	17	LDA	0.821	0.25	0.01
	18	XGB	0.877	104.97	0.08
	19	ET	0.889	6.09	0.47
	20	RF	0.892	4.93	0.12
None	26	NB	0.802	0.37	0.02
	27	LDA	0.830	1.36	0.02
	28	XGB	0.874	938.98	0.15
	29	ET	0.879	59.96	0.75
	30	RF	0.883	69.78	0.22

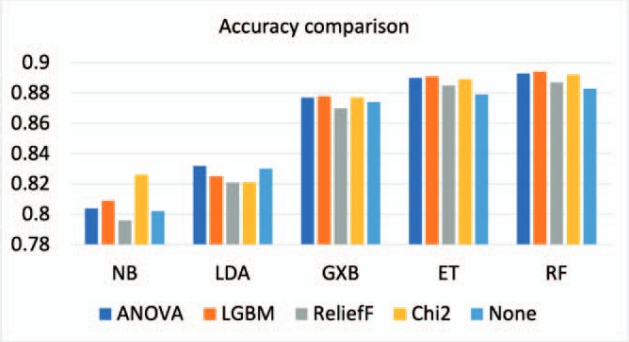

Fig. 1. Accuracy comparison of the proposed models

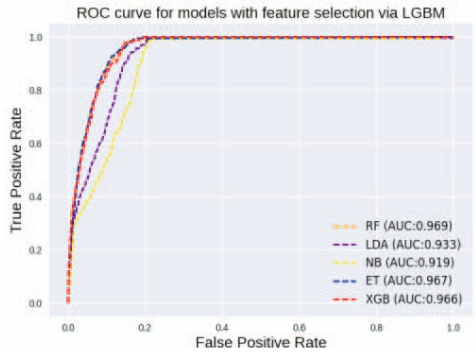

Fig. 2. ROC chart for the proposed models with LGBM

The initial VolMemLyzer tool [17] extracted basic malware features. The authors of [16] developed an extended version of this tool that extracts features related to obfuscated malwares. Table 5 shows that 10 out of the selected 20 features represent obfuscated features of the malware, which are highlighted in orange. Most of these features are aggregated data on handles, including number of file handles, key handles, section handles, mutant handles, semaphore handles, and thread handles. These features give information about obfuscated ransomware activities. Although an obfuscated ransomware intends to hide its activities from anti-malware systems, it appears that its activities on the operating system's file management and handlers follow specific patterns that could be used to help the ML models to detect this type of attacks. The above-mentioned features are highly related as the purpose of ransomware attacks is to get the control of the file system and encrypt the files, which involves a rise on the number of file handlers generated during a short time that the ransomware is actively encrypting the files and changing their access permissions and owners.

The other obfuscation-related features in table 5 are also highly associated with obfuscated ransomware activities, e.g., the average amount of modules missing from the initialization list and the load list. To encrypt the files, ransomwares initially load the files to the memory, then encrypt them, and then store the encrypted files to the disk. However, an obfuscated ransomware changes these steps and manipulates the list of initial files to hide its malicious activities from anti-malwares. However, it appears that using these features the ML models could detect the patterns of such file system manipulations made by obfuscated ransomware. There are also two malfind obfuscation-related features in tables 5, which refer to the hidden code injections. Although obfuscated ransomware tries to hide its code from anti-malwares, the data on hidden code injection captured by the operating system could be used to train the ML models to detect obfuscated ransomwares.

VII. Conclusion

With the rise of ransomware attacks, it is essential to develop a system that can effectively detect these attacks at the network and block them before they spread in the network and perform their malicious actions on the network machines. In this paper, we proposed a family classification framework that uses CIC-MalMem2022 dataset to classify benign vs five different classes of obfuscated ransomware attacks. We performed extensive experiments to study different combinations of feature selection techniques, number of features, and hyperparameter values tunned for five ML algorithms. Our results indicated that RF algorithm with LGBM feature selection technique is the most accurate model for this dataset. The results indicate that employing no feature selection can create models with relatively high accuracies, however, the time to build and test a model with a large number of features is significantly more than a model with selected features. We tunned models for different numbers of features, and 20 features presented the highest accuracy and least training and prediction times for this dataset. We also investigated the features that are highly associated with obfuscated ransomware attacks. Although obfuscated ransomwares hide their malicious activities from anti-malwares, these features collected from the operating system can help ML detect this type of ransomwares.

Table 5: The top 20 features selected by LGBM feature selection method.

No	Feature number	Feature name	LGBM score	Obfuscated	Description
1	47	svcscan.nservices	65,254		Total number of services
2	51	svcscan.shared_process_services	24,509		Total number of Windows 32 shared processes
3	8	dlllist.avg_dlls_per_proc	20,726		Average number of loaded libraries per process
4	12	handles.nfile	11,027	Yes	Total number of file handles
5	15	handles.nkey	5,815	Yes	Total number of key handles
6	29	malfind.commitCharge	5,460	Yes	Total number of Commit Charges
7	48	svcscan.kernel_drivers	5,090		Total number of kernel drivers
8	10	handles.avg_handles_per_proc	4,701		avg handles per proc
9	20	handles.nsection	4,475	Yes	Total number of section handles
10	21	handles.nmutant	4,400	Yes	Total number of mutant handles
11	26	ldrmodules.not_in_init_avg	4,358	Yes	The average amount of modules missing from the initialization list
12	31	malfind.uniqueInjections	3,893	Yes	Total number of unique injections
13	4	pslist.avg_threads	3,556		Average number of threads for the processes
14	25	ldrmodules.not_in_load_avg	3,503	Yes	The average amount of modules missing from the load list
15	18	handles.nsemaphore	3,220	Yes	Total number of semaphore handles
16	16	handles.nthread	2,332	Yes	Total number of thread handles
17	6	pslist.avg_handlers	2,120		Average number of handlers
18	28	malfind.ninjections	2,109		Total number of hidden code injections
19	7	dlllist.ndlls	1,965		Total number of loaded libraries for every process
20	54	callbacks.ncallbacks	1,949		Total number of callbacks

REFERENCES

[1] F. Noorbehbahani, F. Rasouli, and M. Saberi, "Analysis of machine learning techniques for ransomware detection," IEEE ISCISC, 2019. pp. 128–133, doi: 10.1109/ISCISC48546.2019.8985139.

[2] N. Shah and M. Farik, "Ransomware-Threats, Vulnerabilities And Recommendations," International Journal of Scientific and Technology Research, 2017, pp. 307-309.

[3] Available online, https://www.scmagazine.com/news/ransom-demands-recovery-times-payments-and-breach-lawsuits-rise, accessed on 08/15/2023.

[4] S. Kamil S, H.S. Norul, A. Firdaus, O.L. Usman, "The rise of ransomware: A review of attacks, detection techniques, and future challenges," International Conference on Business Analytics for Technology and Security, IEEE ICBATS, 2022, pp. 1-7, doi: 10.1109/ICBATS54253.2022.9759000

[5] U. Adamu and I. Awan, "Ransomware prediction using supervised learning algorithms," International Conference on Future Internet of Things and Cloud, IEEE FiCloud 2019, pp. 57–63. doi: 10.1109/FiCloud.2019.00016.

[6] Kaspersky Security Bulletin, 2023.

[7] SophosLabs 2018 Malware Forecast, 2023.

[8] H. J. Chittooparambil, B. Shanmugam, S. Azam, K. Kannoorpatti, M. Jonkman, and G. N. Samy, G. N. (2019). "A Review of ransomware families and detection methods," International Conference of Reliable Information and Communication Technology (IRICT), Springer, 2018, pp. 588-597, doi: 10.1007/978-3-319-99007-1_55

[9] B. M. Khammas, (2020). "Ransomware detection using random forest technique," ICT Express, Elsevier, 325-331, doi: 10.1016/j.icte.2020.11.001

[10] S. Poudyal, K. P. Subedi, and D. Dasgupta, "A Framework for Analyzing Ransomware using Machine Learning," Symposium Series on Computational Intelligence, IEEE SSCI, 2018, pp. 1692– 1699, doi: 10.1109/SSCI.2018.8628743.

[11] S. I. Bae, G. B. Lee, and E. G. Im, (2020). "Ransomware detection using machine learning algorithms," Concurrency and Computation: Practice and Experience, 32(18), e5422, 2020, doi: 10.1002/cpe.5422

[12] V. G. Ganta, G. V. Harish, V. P. Kumar, and G. R. K. Rao, "Ransomware Detection in Executable Files Using Machine Learning," International Conference on Recent Trends on Electronics, Information, Communication & Technology, IEEE RTEICT, pp. 282–286, 2020, doi: 10.1109/RTEICT49044.2020.9315672.

[13] J.A. Herrera-Silva, M. Hernández-Álvarez, "Dynamic Feature Dataset for Ransomware Detection Using Machine Learning Algorithms," Sensors 2023, doi: 10.3390/s23031053.

[14] K.C. Roy, Q. Chen, "DeepRan: Attention-based BiLSTM and CRF for Ransomware Early Detection and Classification," Information Systems Frontiers, pp. 299–315, 2021, doi:10.1007/s10796-020-10017-4

[15] M. Masum, M.D.J.H. Faruk, H. Shahriar, K. Qian, D. Lo, and M. I. Adnan, "Ransomware classification and detection with machine learning algorithms," In 2022 IEEE Computing and Communication Workshop and Conference (CCWC), 2022, doi: 10.1109/CCWC54503.2022.9720869

[16] T. Carrier, P. Victor, A. Tekeoglu, and A.H. Lashkari, "Detecting Obfuscated Malware using Memory Feature Engineering," ICISSP, 2022. doi: 10.5220/0010908200003120.

[17] A.H. Lashkari, B. Li, T.L. Carrier, and G. Kaur, "Volmemlyzer: Volatile memory analyzer for malware classification using feature engineering," 2021 IEEE Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS), 2021. doi: 10.1109/RDAAPS48126.2021.9452028.