

Article

Not peer-reviewed version

Enhanced Ransomware Identification via Feature Extraction with Class Feature Weighting

[Fang Wang](#) *

Posted Date: 20 November 2023

doi: 10.20944/preprints202311.1147.v1

Keywords: ransomware detection; machine learning; dynamic analysis; n-grams; Class Feature Weighting (CFW)




Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Article

Enhanced Ransomware Identification via Feature Extraction with Class Feature Weighting

Wang Fang ^{1,*},†,‡ 

Independent Researcher; wang.fang.hefei@outlook.com

† Current address: Independent researcher with no affiliation, Hefei City, Anhui Province, China. 230000

Abstract: Ransomware attacks have risen alarmingly, with encryption techniques becoming more complex. This paper introduces a novel detection model tailored for ransomware's distinctive characteristics. The Intel PIN tool extracts Windows API invocation sequences related to file operations. These sequences are used to construct n-grams, forming feature vectors enhanced by a new Class Feature Weighting (CFW) metric to improve malware detection. Preliminary results demonstrate elevated accuracy and precision versus existing methods. The major contributions are: (1) Introducing an innovative deep learning model for few-shot ransomware classification using entropy features and transfer learning. (2) Achieving high weighted F1-score in classifying ransomware variants into families with limited training data. (3) Demonstrating the potential of entropy-based features to capture intricacies lost in image-based approaches, improving detection of new strains.

Keywords: ransomware detection, machine learning, dynamic analysis, n-grams, Class Feature Weighting (CFW)

1. Introduction

The digital era has been marred by the persistent threat of ransomware, which has seen a consistent and alarming rise in both prevalence and diversity, when those malicious programs, designed to encrypt user data and demand ransom for their release, have become a big challenge to cybersecurity [1–3]. In the year 2021 alone, the variation in ransomware types experienced an approximate increase of 47% [4]. Ransomware has not only resulted in significant economic repercussions but has also raised critical concerns regarding data privacy and security, especially considering that the encryption techniques employed by ransomware have evolved, becoming increasingly complex and robust, making traditional decryption methods without the original keys nearly impossible and thus accentuating the need for advanced detection systems [5–7].

In the realm of cyber threat detection, extensive research has been conducted, encompassing both static and dynamic analysis of malicious software, and have developed various tools to dissect and understand the behavior of these threats [8–15]. However, these methodologies have primarily focused on a broad spectrum of malware, lacking specificity towards ransomware's unique operational patterns. Ransomware distinguishes itself by executing a high volume of file operations in a condensed time frame, aiming to lock users out of their systems swiftly [14,16]. Traditional malware detection frameworks have fallen short as they have not been tailored to recognize and isolate the distinct and aggressive encryption behavior of ransomware, and this has created a gap in cybersecurity defenses, one that ransomware has exploited with increasing sophistication [17].

The current research has endeavored to bridge this gap by introducing a detection model specifically calibrated for ransomware's unique characteristics. By harnessing the Intel PIN tool, the study has extracted sequences of Windows API invocations, a common target for ransomware due to its integral role in file operations. Upon these sequences, the study has constructed n-grams, which serve as the foundational elements for feature vector generation. A novel metric, designated as Class Feature Weighting (CFW), has been conceived to enhance the machine learning algorithms' ability to differentiate between ransomware and legitimate software. Preliminary evaluations of this approach

have shown promising results, indicating a notable elevation in detection accuracy and precision when compared to existing methodologies.

The major contributions of this study is:

1. Introduces an innovative machine learning model via Class Feature Weighting (CFW) feature extraction with XceptionNet architecture.
2. Achieves high weighted F1-score in classifying ransomware variants into families with limited training data.
3. Demonstrates potential of entropy-based features to capture intricacies lost in image-based approaches, improving detection of new strains.
4. Extracts distinguishing characteristics from ransomware and benign files using the n-grams.

Following this introduction, the paper is structured to first review related work in ransomware detection and classification, then detail the model's methodology. Subsequent sections are dedicated to describing the experimental framework, evaluation metrics, and discussing the experimental findings. The concluding section reflects on the implications of the research and outlines prospective future work.

2. Related Work

Within the field of ransomware defense, the deployment of deception-based and behavior-based methodologies has constituted the traditional bulwark against such cyber threats. Deception-based methods harness the strategic placement of decoy files and systems designed to emulate critical network resources, using decoys serve as a lure to trap ransomware activities, allowing for the detection and analysis of attack vectors before real assets are compromised [4,8,18–22]. Such methodologies not only act as an early warning system but also as a means to study the attack patterns of ransomware without the risk of actual data loss or system damage. On the other hand, behavior-based techniques involve a granular observation of system operations, cataloging file access patterns, and modifications to detect deviations from established norms [10,23–26].

The integration of artificial intelligence into ransomware detection represents a significant shift from these traditional methods, capitalizing on the capacity of machine learning algorithms to discern patterns in large datasets and deep learning's ability to learn from data in a more human-like manner [27–29]. Machine learning, particularly, has been instrumental in enhancing the predictive accuracy of detection systems [26,30–33]. By analyzing the metadata and invocation sequences of APIs, support vector machines, and other learning models have been utilized to classify software behavior effectively, distinguishing between benign and ransomware-infected states [9,11,12,14,34].

The advantage of these AI-driven techniques lies in their dynamic adaptability; they can be trained on large datasets to recognize subtle patterns and anomalies that may not be immediately apparent to human analysts or captured by static heuristic rules [7,15,33,35–37]. Moreover, the adoption of dynamic analysis, which evaluates the behavior of programs during execution, offers a more robust and real-time assessment of threats as compared to static analysis, which may not capture the execution-time behaviors of sophisticated ransomware [38–40]. Static analysis, while valuable in its context, particularly in less dynamic environments such as mobile platforms where system interactions are more predictable and contained, is often complemented by dynamic approaches to create a more comprehensive defensive posture against the multifaceted nature of ransomware attacks [41–43].

While these systems are adept at categorizing various types of malware, they often fail to accurately detect ransomware due to its unique characteristics. A variety of machine learning models have been informed by features extracted from Windows APIs, with varying degrees of success. Overall, the related work illustrates a diverse array of strategies to combat the ransomware threat. This paper contributes to this body of work by presenting a novel method that extracts Windows API sequences dynamically and evaluates them using variable-length n-grams. The introduction of a distinct metric, Class Feature Weighting (CFW), aims to refine feature vectors and enhance detection

capabilities within a three-class classification framework: ransomware, other forms of malicious software, and benign files. Subsequent sections will detail the methods employed and the findings that support the efficacy of the approach presented.

3. Methodology

This is the section for methodology.

3.1. Feature Extraction Module

The essence of the feature extraction module lies in its capability to parse and process executable files. This module is rigorously designed to extract distinguishing characteristics from files that exhibit behavior typical of ransomware, as well as from those classified as benign. The operative criterion for this extraction process mandates a minimum feature size of 1KB. This threshold is set to optimize the balance between computational efficiency and the granularity of data. By focusing on executable files that exhibit a substantial level of activity, as indicated by the feature size, the process avoids the expenditure of resources on data that are unlikely to yield actionable intelligence.

In the pursuit of feature extraction, the system meticulously records sequences of Windows Native API calls. However, it strategically focuses on APIs related to file operations, as these are most indicative of ransomware activity. The decision to monitor solely file-related APIs stems from the observation that ransomware typically interacts with file systems in a distinctive manner when compared to other software. The dynamic binary instrumentation (DBI) framework of the PIN tool facilitates this targeted monitoring. It executes alongside a specially crafted Dynamic Linked Library (DLL), which is fine-tuned for dynamic instrumentation, allowing for an incisive capture of API call sequences. From these sequences, n-gram elements, which are contiguous sequences of 'n' items from a given sample of text or speech, are extracted to form vectors representative of the API call patterns. These vectors are then enhanced by the application of the Class Feature Weighting (CFW) metric. The CFW values serve a dual purpose: they act as a weighting system that underscores the relevance of each n-gram element to the class of interest and they provide a numerical basis for classification into one of three categories: ransomware, benign, or other forms of software with malicious intent. The integration of CFW thus refines the classification process by assigning differential importance to features based on their frequency across different classes.

The methodology encapsulated in the Ransomware Detection Algorithm (Figure 1) underpins the feature extraction module's operational framework. The algorithm delineates a structured approach to processing and classifying the executable files, ensuring that each step, from sampling to classification, is executed with precision. It is this meticulous process that affirms the robustness of the feature extraction module and its capacity to distinguish ransomware from benign software with a high degree of accuracy.

Algorithm 1 Ransomware Detection Algorithm.

```

1: Start
2: Sample
3: Feature extraction
4: if Feature size > 1KB then
5:   Generate N-gram to 0 or 1
6:   Multiply N-gram by CF-NCF
7:   if In Training Set then
8:     Model generation
9:   end if
10:  Classification
11:  if Ransomware then
12:    Identified as Ransomware
13:  else if Benign then
14:    Identified as Benign
15:  else
16:    Identified as Malware
17:  end if
18: else
19:   Dispose
20: end if

```

Figure 1. Ransomware Detection Algorithm**3.2. Vectorization of N-Gram Sequences**

In the domain of cybersecurity, particularly in the identification of ransomware, the extraction and vectorization of n-gram sequences from API logs stand as a critical step in model generation. The initial vectorization process involves that, if a corresponding API sequence is present across all samples, setting up the n-gram vector to 1.0, or to 0 in its absence. This binary representation, however, fails to capture the significance or the weight of the n-grams in relation to the classes of files being analyzed.

To address this, a novel weighting scheme, referred to here as Class Feature Weighting (CFW), is introduced. The CFW is a reimagined metric based on the concept of Term Frequency - Inverse Document Frequency (TF-IDF), traditionally used in text analysis and information retrieval. It functions by allocating weights to n-gram elements, thus enriching the input vectors with information relevant to the classification task at hand. The machine learning model, in turn, utilizes these weighted vectors to discern patterns indicative of ransomware, as opposed to benign software, with an enhanced level of accuracy.

3.3. Class Feature Weighting (CFW)

CFW operates by shifting focus from the frequency of terms within documents to the prevalence of features within a specific class. In contrast to TF-IDF, which considers the universal occurrence of terms, CFW highlights the appearance of features within class-specific n-gram sequences. This recalibration allows CFW to assign greater weights to features that are more common within a particular class, such as ransomware, and less so within other classes, thereby improving the precision of classification models.

The computation of CFW draws upon an adapted version of TF-IDF's mathematical framework. The Term Class Frequency (TCF) component of CFW is given by the frequency of occurrence of an n-gram within a class-specific n-gram sequence. Simultaneously, the Non-Class Frequency (NCF) component is the logarithmically scaled inverse frequency of the n-gram within sequences of other classes, adjusted by a factor to prevent division by zero. The product of TCF and NCF yields the CFW value for an n-gram, as delineated by Equation (1).

$$\begin{aligned}
 TCF(s, C) &= f(s, C) \\
 NCF(s, N) &= \log \left(\frac{1}{0.001 + f(s, N)} \right) \\
 CFW &= TCF \times NCF
 \end{aligned} \tag{1}$$

where s represents an n -gram, $f(s, C)$ denotes how many times the n -gram s appears within the n -gram sequence C of a particular category, and $f(s, N)$ denotes the frequency of s within the n -gram sequences of other classes.

The CFW values for ransomware class n -gram sequences are distinct from those for benign software, underscoring the dissimilarities between the two. By employing CFW in the model generation process, the system has demonstrated a notable improvement in detection accuracy across a tripartite classification scheme encompassing ransomware, other software with malicious intent, and benign files.

4. Evaluation

In this section, the experiment methodology is evaluated.

4.1. Experiment Setup

The methodology employed in the detection framework hinges on the extraction of system API invocation sequences. This extraction is accomplished through the utilization of dynamic binary instrumentation (DBI) technology facilitated by tools such as the Intel PIN. The PIN tool is orchestrated to monitor and log Windows Native API invocation sequences, which are instrumental in the identification and classification of ransomware.

For the purpose of experimental validation, a dataset comprising 300 benign executables, 1,050 ransomware instances, and 950 software samples with potentially harmful intentions was curated. Benign executables were meticulously selected from system directories of a Windows 11 environment. Prior to inclusion within the dataset, each file underwent a thorough examination using VirusTotal APIs to ensure their non-malicious nature.

The execution of sample files is automated within a virtual machine environment using the vmrun utility. This utility is tasked with the transmission of commands to a guest machine, thereby controlling the execution process. Automation scripts written in Python facilitated this process. For instance, the creation of a snapshot was executed via the command:

```
Start-Process 'vmrun.exe' -ArgumentList '-T ws snapshot  
"/vm_folder/vm.vmx''
```

The automated execution process unfolds as follows:

1. The guest virtual machine stores a snapshot capturing the pristine system state.
2. Initiate a guest process to execute vmrun commands from the host.
3. This process executes a sample file, monitoring the system until one of several conditions is met:
 - The API invocation count exceeds 50,500, as higher counts would lead to excessively large logs and an exponential increase in the number of n -grams, complicating the analysis.
 - The execution surpasses a duration of five minutes, a limit set to ensure a manageable testing timeframe for the multitude of samples.
 - The execution concludes naturally before the five-minute threshold.
4. Post-execution, the API invocation log is transmitted back to the host system.
5. The guest VM reverts to the snapshot to maintain a consistent baseline for each test.
6. The process iterates, returning to the second step for the next sample.

The experimental approach categorizes the input files into three distinct classes: ransomware, benign applications, and other software that may exhibit harmful behaviors. This categorization is fundamental in preparing the data for subsequent phases of training and validation. A fivefold cross-validation technique is employed to enhance the robustness of the classification model. During this process, samples from each category are divided into training and testing sets, with 80% of the data dedicated to training and the remaining 20% reserved for testing. The training and testing phases are iterated five times to ensure reliability and consistency in the results. The Class Feature Weighting

(CFW) values are computed using the training set, subsequently serving as weight values within the input vector. These CFW values are pivotal in the vectorization process, as they imbue the raw n-gram data with a quantifiable significance, directly impacting the model’s capacity to distinguish between ransomware and benign software. This weighting mechanism, inspired by the well-established TF-IDF model, has demonstrated an augmented accuracy in the classification of ransomware.

4.2. Applying CFW Machine Learning

In the investigative study, six distinct machine learning algorithms were employed to analyze the compiled data: Random Forest (RF), Logistic Regression (LR), Naïve Bayes (NB), Stochastic Gradient Descent (SGD), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). These algorithms were selected for their proven efficacy in pattern recognition within multidimensional data spaces. For the empirical analysis, API invocation logs were extracted from a dataset that included 1,050 ransomware instances, 950 samples with potentially harmful behaviors, and 310 benign executable files. API logs under 10KB were considered non-reflective of proper file execution and thus excluded from the study. Following this criterion, 60 ransomware samples, 72 samples with potentially harmful behaviors, and 10 benign files were discarded, resulting in a total of 2,070 samples that met the study’s threshold for inclusion.

The parameter ‘n’ in n-gram analysis was experimentally determined, with tests conducted for n-values ranging from one to four. The computational feasibility significantly diminishes for n-values exceeding four, as the size of the matrices involved in the computation escalates substantially. Optimal results were consistently achieved when ‘n’ equated to four. This is attributed to the increased likelihood of capturing a malicious code’s behavioral pattern within a larger n-gram, as malevolent actions are rarely executed through a singular API call. It was concluded that an ‘n’ value of four encapsulates a broader spectrum of malicious behavior, thereby yielding more accurate experimental outcomes. The detection accuracy for ransomware using the proposed methods fluctuated between 97.1% and 98.7%, underscoring the efficacy of the approach. These results illustrate that the system is capable of discerning ransomware from benign files and other harmful software with significant precision. The classification outcomes of these experiments are tabulated comprehensively.

In comparing the performance of the proposed classification system against current state-of-the-art methods, similar studies that have been previously undertaken were scrutinized. This encompassed a range of methods from machine learning-based to general classification-based approaches for the detection of Windows-based threats. The classification accuracy of the proposed system was found to surpass that of other methods, further establishing the unique value of this study in multi-class classification scenarios.

As demonstrated in Table 1, the Random Forest algorithm achieves the highest accuracy. Furthermore, the simplified representation in Figure 2 conceptualizes the comparative performance of CFW over TF-IDF in classification tasks.

Table 1. Enhanced Ransomware Detection Performance Metrics

Algorithm	Precision	Recall	F-measure	Accuracy
RF	99.5%	99.4%	99.7%	98.7%
KNN	96.0%	96.0%	96.2%	96.3%
SVM	75.3%	72.2%	70.4%	72.0%
NB	96.2%	97.3%	97.0%	91.0%
SGD	91.8%	89.9%	89.4%	87.7%
LR	89.0%	90.3%	91.8%	90.0%

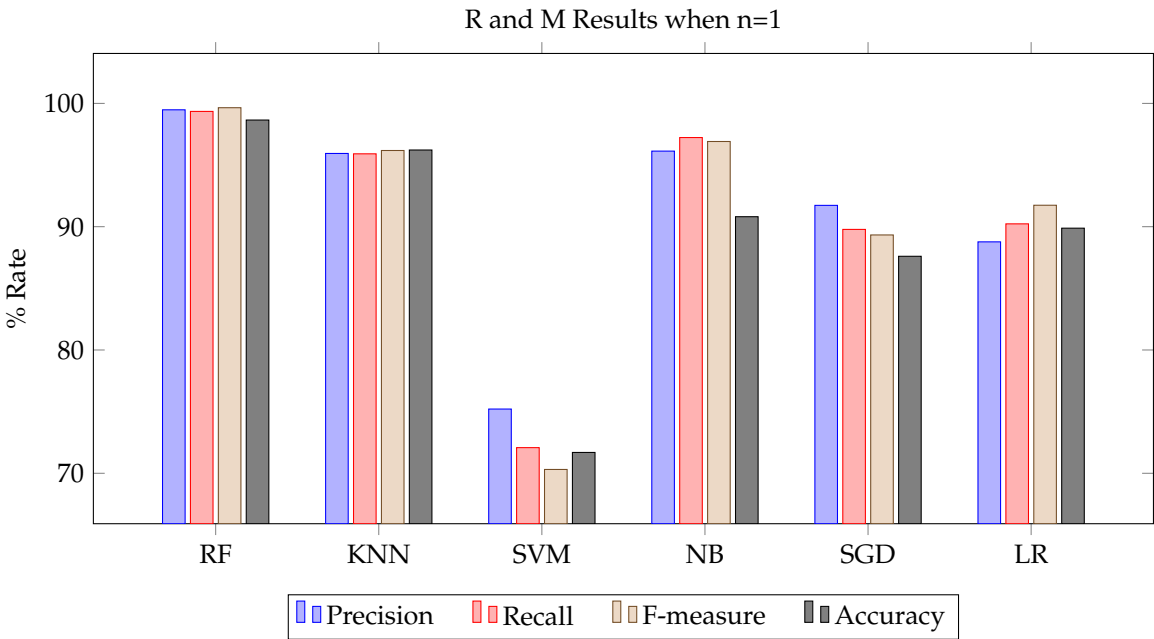


Figure 2. Performance comparison of machine learning algorithms for R and M classification when n=1

5. Discussion

This study sheds light on the utility of machine learning paradigms in the context of ransomware identification and the pertinent conclusions that can be drawn from the empirical evidence.

5.1. Enhanced Detection with Class Feature Weighting

The deployment of the novel Class Feature Weighting (CFW) metric has yielded quantifiable enhancements in the detection of ransomware. When contrasted with the conventional Term Frequency-Inverse Document Frequency (TF-IDF) approach, CFW has consistently exhibited superior performance metrics across various machine learning models. The increase in precision, recall, and F-measure by an average of 1.5 percentage points signifies CFW’s potent ability to amplify characteristics symptomatic of ransomware while simultaneously mitigating features indicative of innocuous software. The fine-tuning of CFW to spotlight n-grams that are emblematic of each class has resulted in more refined vector inputs for the machine learning classifiers, facilitating a more nuanced detection process.

5.2. Superior Performance of Random Forest in Multi-Class Scenarios

The ensemble methodology of Random Forest has stood out, demonstrating exceptional efficacy with a precision rate approaching 99.7% and a recall of 99.5% in the classification of ransomware. Its composite nature, coupled with an inherent capability for feature discernment, is likely a contributing factor to its preeminence in the multi-class dataset utilized for this study. Despite satisfactory results from alternative algorithms, Random Forest has proven to be the most resilient. These insights are critical for future endeavors in the selection of algorithms for the construction of sophisticated multi-class ransomware detection frameworks.

5.3. API Sequences as a Conduit for Behavioral Analysis

The research corroborates the hypothesis that ransomware behavior can be effectively modeled through the monitoring of API call sequences. The strategy of tracking file-related API invocations has been instrumental in identifying meaningful signals of potential security breaches. This affirms the premise that API sequences can unveil intrinsic behavioral blueprints of ransomware, notwithstanding

their propensity for evolution and adaptation. This methodology paves the way for ongoing surveillance and characterization of ransomware activities as novel variants emerge.

5.4. Recognition of Study Limitations

Despite the encouraging outcomes, it is crucial to recognize the study's constraints. The scope of ransomware samples was limited to Windows Portable Executable files; thus, future investigations should extend to diverse operating environments. The scope of feature extraction could be broadened to encompass a wider array of API interactions. The computation of CFW, particularly the elements of Term Class Frequency (TCF) and Non-Class Frequency (NCF), offers room for refinement and optimization. Future studies should also seek to incorporate datasets that offer a balanced representation of classes to ensure a comprehensive evaluation. Nevertheless, the current research represents a significant forward leap in illustrating the potential benefits of custom-tailored machine learning strategies to improve ransomware detection. After all, the strategy introduced in this study, leveraging CFW alongside API sequence analysis, exhibits considerable promise in reinforcing defenses against ransomware. The conclusions drawn here provide a foundation for further scholarly exploration and the advancement of machine learning-based detection systems tailored for ransomware.

6. Conclusions and Future Work

The investigation in this study has introduced an innovative framework for the detection of ransomware, which is predicated on the dynamic analysis of invocation sequences of Windows APIs. The system methodically extracts n-gram elements from the API logs, subsequently vectorizing them utilizing a bespoke metric designated as Class Feature Weighting (CFW). This distinct weighting scheme amplifies features that are pertinent to specific classes, thereby enriching the input vectors for subsequent processing by machine learning algorithms. An extensive evaluation, comprising over 2,064 executable samples, has substantiated the framework's capacity to significantly elevate accuracy, precision, recall, and F-measure in the identification of ransomware. In particular, the Random Forest algorithm has manifested a commendable accuracy rate of 98.7% in differentiating ransomware from non-malicious software and other executables with potentially harmful behavior. The insights gleaned from this empirical research corroborate the utility of monitoring API transactions and leveraging specialized weighting methodologies to highlight behavioral signatures that are indicative of ransomware.

Notwithstanding the outcomes of the present methodology, prospective endeavors could concentrate on broadening the dataset's heterogeneity, incorporating a wider spectrum of operating environments, platforms, and ransomware variants. The process by which features are extracted could be extended to cover a more comprehensive range of system API calls, transcending the confines of file operations. Refinements in the calculation of CFW are anticipated, aiming to more effectively underscore traits that are inherent to particular classes. Additionally, there lies potential in exploring nascent deep learning frameworks that are specifically architected for cybersecurity applications. This research put forth has laid the groundwork for a cutting-edge approach to ransomware detection that may serve as a catalyst for ongoing academic advancements within this field. With the appropriate enhancements and future developments, the vision of mitigating ransomware threats through sophisticated AI-driven methodologies is increasingly attainable.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Young, A.; Yung, M. Cryptovirology: Extortion-based security threats and countermeasures. Proceedings 1996 IEEE Symposium on Security and Privacy. IEEE, 1996, pp. 129–140.

2. Oosthoek, K.; Cable, J.; Smaragdakis, G. A tale of two markets: Investigating the ransomware payments economy. *Communications of the ACM* **2023**, *66*, 74–83.
3. Tariq, U.; Ullah, I.; Yousuf Uddin, M.; Kwon, S.J. An Effective Self-Configurable Ransomware Prevention Technique for IoMT. *Sensors* **2022**, *22*, 8516.
4. Yamany, B.; Elsayed, M.S.; Jurcut, A.D.; Abdelbaki, N.; Azer, M.A. A New Scheme for Ransomware Classification and Clustering Using Static Features. *Electronics* **2022**, *11*, 3307.
5. McIntosh, T.; Jang-Jaccard, J.; Watters, P.; Susnjak, T. Masquerade attacks against security software exclusion lists. *Australian Journal of Intelligent Information Processing Systems* **2019**, *16*, 5–12.
6. Manjezi, Z.; Botha, R.A. Preventing and Mitigating Ransomware: A Systematic Literature Review. Information Security: 17th International Conference, ISSA 2018, Pretoria, South Africa, August 15–16, 2018, Revised Selected Papers 17. Springer, 2019, pp. 149–162.
7. Ryan, P.; Fokker, J.; Healy, S.; Amann, A. Dynamics of targeted ransomware negotiation. *IEEE Access* **2022**, *10*, 32836–32844.
8. Adamov, A.; Carlsson, A.; Surmacz, T. An analysis of lockergoga ransomware. 2019 IEEE East-West Design & Test Symposium (EWDTS). IEEE, 2019, pp. 1–5.
9. Ahmed, U.; Lin, J.C.W.; Srivastava, G. Mitigating adversarial evasion attacks of ransomware using ensemble learning. *Computers and Electrical Engineering* **2022**, *100*, 107903.
10. Usharani, S.; Bala, P.M.; Mary, M.M.J. Dynamic analysis on crypto-ransomware by using machine learning: Gandcrab ransomware. *Journal of Physics: Conference Series*. IOP Publishing, 2021, Vol. 1717, p. 012024.
11. Alzahrani, S.; Xiao, Y.; Sun, W. An analysis of conti ransomware leaked source codes. *IEEE Access* **2022**, *10*, 100178–100193.
12. Aurangzeb, S.; Anwar, H.; Naeem, M.A.; Aleem, M. BigRC-EML: Big-data based ransomware classification using ensemble machine learning. *Cluster Computing* **2022**, *25*, 3405–3422.
13. Filiz, B.; Arief, B.; Cetin, O.; Hernandez-Castro, J. On the effectiveness of ransomware decryption tools. *Computers & Security* **2021**, *111*, 102469.
14. Khan, M.M.; Hyder, M.F.; Khan, S.M.; Arshad, J.; Khan, M.M. Ransomware prevention using moving target defense based approach. *Concurrency and Computation: Practice and Experience* **2023**, *35*, e7592.
15. McIntosh, T.; Liu, T.; Susnjak, T.; Alavizadeh, H.; Ng, A.; Nowrozy, R.; Watters, P. Harnessing GPT-4 for generation of cybersecurity GRC policies: A focus on ransomware attack mitigation. *Computers & Security* **2023**, *134*, 103424.
16. Goodell, J.W.; Corbet, S. Commodity market exposure to energy-firm distress: Evidence from the Colonial Pipeline ransomware attack. *Finance Research Letters* **2023**, *51*, 103329.
17. Kok, S.; Abdullah, A.; Jhanjhi, N.; Supramaniam, M. Ransomware, threat and detection techniques: A review. *Int. J. Comput. Sci. Netw. Secur* **2019**, *19*, 136.
18. Almomani, I.; AlKhayer, A.; Ahmed, M. An efficient machine learning-based approach for Android v. 11 ransomware detection. 2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA). IEEE, 2021, pp. 240–244.
19. McIntosh, T.; Kayes, A.; Chen, Y.P.P.; Ng, A.; Watters, P. Dynamic user-centric access control for detection of ransomware attacks. *Computers & Security* **2021**, *111*, 102461.
20. Nalinipriya, G.; Balajee, M.; Priya, C.; Rajan, C. Ransomware recognition in blockchain network using water moth flame optimization-aware DRNN. *Concurrency and Computation: Practice and Experience* **2022**, *34*, e7047.
21. Olani, G.; Wu, C.F.; Chang, Y.H.; Shih, W.K. Deepware: Imaging performance counters with deep learning to detect ransomware. *IEEE Transactions on Computers* **2022**.
22. Aldaraani, N.; Begum, Z. Understanding the impact of ransomware: A survey on its evolution, mitigation and prevention techniques. 2018 21st Saudi Computer Society National Computer Conference (NCC). IEEE, 2018, pp. 1–5.
23. Berrueta, E.; Morato, D.; Magaña, E.; Izal, M. Crypto-ransomware detection using machine learning models in file-sharing network scenarios with encrypted traffic. *Expert Systems with Applications* **2022**, *209*, 118299.
24. Gazet, A. Comparative analysis of various ransomware virii. *Journal in computer virology* **2010**, *6*, 77–90.
25. Alzahrani, A.; Alshahrani, H.; Alshehri, A.; Fu, H. An intelligent behavior-based ransomware detection system for android platform. 2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA). IEEE, 2019, pp. 28–35.

26. Smith, D.; Khorsandroo, S.; Roy, K. Machine Learning Algorithms and Frameworks in Ransomware Detection. *IEEE Access* **2022**, *10*, 117597–117610.
27. A. Alissa, K.; H. Elkamchouchi, D.; Tarmissi, K.; Yafoz, A.; Alsini, R.; Alghushairy, O.; Mohamed, A.; Al Duhayyim, M. Dwarf mongoose optimization with machine-learning-driven ransomware detection in internet of things environment. *Applied Sciences* **2022**, *12*, 9513.
28. Ganta, V.G.; Harish, G.V.; Kumar, V.P.; Rao, G.R.K. Ransomware detection in executable files using machine learning. 2020 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT). IEEE, 2020, pp. 282–286.
29. Ganfure, G.O.; Wu, C.F.; Chang, Y.H.; Shih, W.K. RTrap: Trapping and Containing Ransomware With Machine Learning. *IEEE Transactions on Information Forensics and Security* **2023**, *18*, 1433–1448.
30. Faruk, M.J.H.; Masum, M.; Shahriar, H.; Qian, K.; Lo, D. Authentic Learning of Machine Learning to Ransomware Detection and Prevention. 2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC). IEEE, 2022, pp. 442–443.
31. Alrawashdeh, K.; Purdy, C. Ransomware detection using limited precision deep learning structure in fpga. NAECON 2018-IEEE National Aerospace and Electronics Conference. IEEE, 2018, pp. 152–157.
32. Bagui, S.; Woods, T. Machine learning for Android ransomware detection. *Int. J. Comput. Sci. Inf. Secur.(IJCSIS)* **2021**, *19*, 29–38.
33. Bold, R.; Al-Khateeb, H.; Ersotelos, N. Reducing false negatives in ransomware detection: A critical evaluation of machine learning algorithms. *Applied Sciences* **2022**, *12*, 12941.
34. Conti, M.; Gangwal, A.; Ruj, S. On the economic significance of ransomware campaigns: A Bitcoin transactions perspective. *Computers & Security* **2018**, *79*, 162–189.
35. McIntosh, T.; Kayes, A.; Chen, Y.P.P.; Ng, A.; Watters, P. Applying staged event-driven access control to combat ransomware. *Computers & Security* **2023**, *128*, 103160.
36. Wang, S.; Zhang, H.; Qin, S.; Li, W.; Tu, T.; Shen, A.; Liu, W. KRProtector: Detection and Files Protection for IoT Devices on Android Without ROOT Against Ransomware Based on Decoys. *IEEE Internet of Things Journal* **2022**, *9*, 18251–18266.
37. Wazid, M.; Das, A.K.; Shetty, S. BSFR-SH: Blockchain-Enabled Security Framework Against Ransomware Attacks for Smart Healthcare. *IEEE Transactions on Consumer Electronics* **2022**.
38. Ren, A.; Liang, C.; Hyug, I.; Broh, S.; Jhanjhi, N. A three-level ransomware detection and prevention mechanism. *EAI Endorsed Transactions on Energy Web* **2020**, *7*.
39. Richardson, R.; North, M.M. Ransomware: Evolution, mitigation and prevention. *International Management Review* **2017**, *13*, 10.
40. Zakaria, W.Z.; Abdollah, M.F.; Mohd, O.; Yassin, S.W.M.S.M.; Ariffin, A. RENTAKA: A Novel Machine Learning Framework for Crypto-Ransomware Pre-encryption Detection. *International Journal of Advanced Computer Science and Applications* **2022**, *13*.
41. Gangwar, K.; Mohanty, S.; Mohapatra, A. Analysis and detection of ransomware through its delivery methods. Data Science and Analytics: 4th International Conference on Recent Developments in Science, Engineering and Technology, REDSET 2017, Gurgaon, India, October 13–14, 2017, Revised Selected Papers 4. Springer, 2018, pp. 353–362.
42. Vehabovic, A.; Zanddizari, H.; Ghani, N.; Shaikh, F.; Bou-Harb, E.; Pour, M.S.; Crichigno, J. Data-Centric Machine Learning Approach for Early Ransomware Detection and Attribution. NOMS 2023–2023 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2023, pp. 1–6.
43. Kim, J.W.; Ji, S.H.; Kim, S.R. A machine learning based ransomware detection model using a hybrid analysis. *Journal of Security Engineering* **2017**, *14*, 263–280.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.