

Article

A Holistic Approach to Ransomware Classification: Leveraging Static and Dynamic Analysis with Visualization

Bahaa Yamany ¹, Mahmoud Said Elsayed ^{2,*} , Anca D. Jurcut ² , Nashwa Abdelbaki ¹  and Marianne A. Azer ^{1,3} 

¹ School of Information Technology and Computer Science, Nile University, Cairo 12566, Egypt; b.yamany@nu.edu.eg (B.Y.); nabdelbaki@nu.edu.eg (N.A.)

² School of Computer Science, University College Dublin, Belfield, D04 V1W8 Dublin, Ireland; anca.jurcut@ucd.ie

³ Computers and Systems Department, National Telecommunication Institute, Cairo 11768, Egypt; mazer@nu.edu.eg

* Correspondence: mahmoud.abdallah@ucdconnect.ie

Abstract: Ransomware is a type of malicious software that encrypts a victim's files and demands payment in exchange for the decryption key. It is a rapidly growing and evolving threat that has caused significant damage and disruption to individuals and organizations around the world. In this paper, we propose a comprehensive ransomware classification approach based on the comparison of similarity matrices derived from static, dynamic analysis, and visualization. Our approach involves the use of multiple analysis techniques to extract features from ransomware samples and to generate similarity matrices based on these features. These matrices are then compared using a variety of comparison algorithms to identify similarities and differences between the samples. The resulting similarity scores are then used to classify the samples into different categories, such as families, variants, and versions. We evaluate our approach using a dataset of ransomware samples and demonstrate that it can accurately classify the samples with a high degree of accuracy. One advantage of our approach is the use of visualization, which allows us to classify and cluster large datasets of ransomware in a more intuitive and effective way. In addition, static analysis has the advantage of being fast and accurate, while dynamic analysis allows us to classify and cluster packed ransomware samples. We also compare our approach to other classification approaches based on single analysis techniques and show that our approach outperforms these approaches in terms of classification accuracy. Overall, our study demonstrates the potential of using a comprehensive approach based on the comparison of multiple analysis techniques, including static analysis, dynamic analysis, and visualization, for the accurate and efficient classification of ransomware. It also highlights the importance of considering multiple analysis techniques in the development of effective ransomware classification methods, especially when dealing with large datasets and packed samples.

Keywords: dynamic analysis; encryption; honeypot; Jaccard index; malware; machine learning; ransomware; similarity matrix; shared code analysis; static analysis



Citation: Yamany, B.; Elsayed, M.S.; Jurcut, A.D.; Abdelbaki, N.; Azer, M.A. A Holistic Approach to Ransomware Classification: Leveraging Static and Dynamic Analysis with Visualization. *Information* **2024**, *15*, 46. <https://doi.org/10.3390/info15010046>

Academic Editor: Ruggero Lanotte

Received: 11 December 2023

Revised: 30 December 2023

Accepted: 5 January 2024

Published: 14 January 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Malware analysis is the act of finding, comprehending, and minimizing the potential damage caused by malicious software, such as ransomware in ref. [1]. It is a crucial component of cybersecurity since it enables organizations and individuals to defend themselves against the numerous types of malware that might infect their systems and data. Malware analysis employs a variety of tools and methodologies, including static analysis, dynamic analysis, sandbox analysis, and reverse engineering. These methods can be used to analyze the code and behavior of malware and to identify indicators of compromise (IOCs) that can be used to detect and categorize malware. Malware analysis is an important part of defending against ransomware as it allows organizations and individuals to identify and mitigate the potential harm caused by ransomware before it can cause significant

damage or disruption. It can also help to identify and track the activities of ransomware operators, which can provide valuable intelligence for law enforcement and other cybersecurity professionals. In addition to traditional malware analysis techniques, there are also automated malware analysis tools and platforms that can be used to automate and streamline the analysis process. These tools can help to reduce the time and resources required for manual analysis, as well as increase the speed and accuracy of the analysis process in ref. [2]. However, it is important to carefully consider the benefits and limitations of automated malware analysis as it may not always provide the same level of depth and detail as manual analysis. Static analysis and dynamic analysis are two approaches that can be used to analyze and classify ransomware in ref. [3]. Both approaches have their own benefits and limitations, and they can be used in combination or separately depending on the specific needs of the analysis.

Ransomware represents a form of malicious software that encrypts a victim's files and subsequently demands a ransom in exchange for the decryption key. This perilous threat is marked by its rapid proliferation and constant evolution, resulting in significant harm and disruption to individuals and entities worldwide [4]. Ransomware deployment encompasses various techniques, including exploit kits, drive-by downloads, and social engineering strategies. Common vectors for its transmission include email attachments, compromised websites, and software vulnerabilities. Upon infiltration, ransomware typically encrypts a wide array of files, ranging from documents to images, holding them hostage. Subsequently, victims are confronted with ransom demands, often presented through on-screen messages, or concealed notes within their systems. These demands typically include a stipulated payment deadline and a menacing ultimatum to delete the victim's data should the ransom go unpaid. The repercussions of a ransomware attack can be profound, resulting in operational disruption, critical data loss, and substantial financial setbacks. Victims facing such attacks may find themselves at a crossroads, compelled to either pay the ransom for data recovery or explore alternative avenues, such as data restoration from backups or decryption techniques. Importantly, even when a ransom is paid, there is no guarantee that the ransomware operators will honor their promise to provide the decryption key [5]. The escalating prevalence and sophistication of ransomware assaults pose a global threat to both individuals and businesses. Being prepared to respond to and recover from such attacks, as well as proactively recognizing the threat and implementing precautionary measures, assumes paramount importance for safeguarding against this formidable adversary [6]. Figure 1 offers a comprehensive overview of the various phases involved in a ransomware attack, spanning from its inception to the extortion phase.

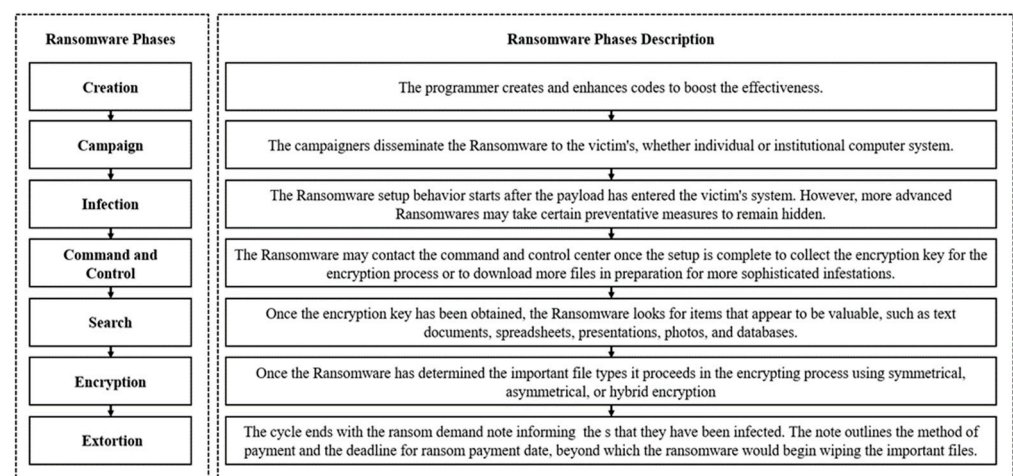


Figure 1. Ransomware lifecycle from creation to extortion.

Paper Contribution

In this paper, our primary focus was a meticulous examination of a substantial dataset containing ransomware samples, embedded within a broader corpus. This extensive analysis led to the identification of a subset of ransomware samples sharing notable similarities. Subsequently, we conducted a rigorous assessment using a similarity matrix-based analysis, incorporating both static and dynamic features, with the overarching goal of offering a comprehensive evaluation that highlights the respective merits and limitations of each analytical approach.

Beyond our innovative methodological approach, we conducted a thorough survey and comparative examination of established ransomware detection methodologies. Our study presents an expansive exploration of the ransomware detection ecosystem, covering various dimensions, including the detection environment, data analysis techniques, machine learning methodologies, outcomes, evaluation criteria, and a range of available detection tools.

Within the context of this research, our contributions encompass a multifaceted exploration of the following key aspects:

- Comparative analysis of infection behaviors across various ransomware families.
- Utilization of data visualization methods for the identification of similar ransomware samples within extensive datasets.
- Employing a similarity matrix approach for the analysis of static and dynamic features in ransomware samples.
- Assessment of the merits and limitations associated with static and dynamic feature analysis.
- Comprehensive survey and comparative evaluation of varied ransomware detection approaches, alongside an in-depth exploration of the ransomware detection ecosystem.
- Development and proposal of an automated methodology for extracting diverse feature sets from ransomware samples.

The remainder of this paper is organized as follows. In Section 2, we present an overview of the efforts that have been made in the literature to develop ransomware detection approaches. We survey the different techniques that have been proposed and analyze the criteria, parameters, and tools used in the ransomware detection ecosystem. In Section 3, we provide a background on the different static and dynamic features that have been used in ransomware tracking systems as well as the visualization techniques that have been proposed for ransomware classification. In Section 4, we describe our system setup and present the results and analysis of our proposed approach for extracting the malware's static features and classifying ransomware samples. We also compare our results to those of other approaches proposed in the literature. Finally, in Section 5, we provide conclusions and discuss future work in the field of ransomware detection. We outline the challenges and opportunities that exist for improving the accuracy and effectiveness of ransomware detection and highlight the potential impacts of these advances on cybersecurity.

2. Related Work

In this section, we aim to delve deeper into the related work, refine the problem statement by addressing its limitations, and provide additional context regarding the categorization of ransomware. Within the scope of this paper, our objective is to conduct a comprehensive survey of the diverse spectrum of ransomware detection methodologies and techniques as delineated in the existing literature. Beyond this survey, we undertake a meticulous analysis of the varied criteria, parameters, and tools employed within the broader ransomware detection ecosystem. The overarching goal is to furnish readers with an in-depth understanding of the contemporary landscape in ransomware detection, including both its advancements and inherent challenges. However, it is imperative to acknowledge certain limitations in this pursuit. Firstly, the rapidly evolving nature of ransomware demands a continuous update of detection methodologies, and as such, some state-of-the-art techniques may not be covered if they have emerged after our knowledge

cutoff date. Secondly, the effectiveness of ransomware detection can be context-dependent, varying based on factors such as the specific ransomware variant, its obfuscation techniques, and the target environment. These contextual variations pose challenges in proposing a one-size-fits-all solution.

Categorizing ransomware is a crucial aspect of understanding the threat landscape. Ransomware can be classified into various categories based on its characteristics, propagation methods, and behavior. Some common categories include:

- **Encrypting Ransomware:** This category involves ransomware that encrypts files on the victim's system, rendering them inaccessible until a ransom is paid.
- **Locker Ransomware:** Locker ransomware locks the victim out of their entire system, preventing access until a ransom is provided.
- **Doxware or Leakware:** This type threatens to release sensitive information unless a ransom is paid, often compromising privacy.
- **Scareware:** Scareware displays false warnings or claims of malware infections, extorting money for their removal.
- **Mobile Ransomware:** Designed for mobile devices, this category targets smartphones and tablets, encrypting data or locking the device.
- **Ransomware-as-a-Service (RaaS):** RaaS platforms allow cybercriminals to easily create and distribute ransomware, contributing to its proliferation.
- **Targeted Ransomware:** Some ransomware attacks are highly targeted, focusing on specific organizations or individuals, often with higher ransom demands.
- **Cryptojacking:** While it is not traditional ransomware, cryptojacking malware hijacks computer resources to mine cryptocurrencies, often without the victim's consent.

The proliferation of computers, the Internet, and applications has introduced threats, including malicious software or malware. One study in ref. [7] focused on ransomware, a type of malware that encrypts user files, demanding a ransom for their release. Despite advisories against paying ransoms, victims commonly resort to this measure. The paper emphasized the need for advanced protection measures against ransomware, highlighting the importance of understanding its nature for effective defense. While existing surveys touch on technical aspects, there is a dearth of comprehensive reviews dedicated to exploring ransomware research. This paper seeks to fill this gap by providing a detailed survey and introducing a new ransomware taxonomy. The survey covers ransomware threat factors, taxonomy, and existing research, offering valuable insights for future endeavors in this domain.

Categorization aids in understanding the *modus operandi* of different ransomware variants and tailoring detection and mitigation strategies accordingly. As we proceed with our analysis, it is essential to consider these categories and their implications on detection and prevention strategies. Furthermore, we acknowledge that the ransomware landscape is dynamic, and new categories or variants may emerge over time, necessitating ongoing research and adaptation of security measures.

Ransomware detection through reverse engineering entails shared code analysis to identify analyzable sample groupings, aiding in developer attribution and variant identification. Shared code analysis enables swift determination of code commonality between new and previously analyzed samples. In a standard malware detection system, the primary components encompass feature extraction, feature selection, classification/clustering, and decision-making. Raw data are processed through feature extraction, yielding relevant features. Feature selection reduces complexity by identifying feature correlations. The resulting feature vector undergoes classification or clustering, with the decision module distinguishing between malicious and benign samples in ref. [8]. In ref. [9], the authors conducted a comparative study between static and dynamic malware analysis techniques. Both static and dynamic analysis approaches hold significant value in the realm of ransomware analysis and classification. The choice between these methods hinges on the specific requirements, available resources, and characteristics of the ransomware samples under scrutiny. Often, a synergistic combination of static and dynamic analysis proves to

be the most effective approach. Identifying the malware family to which a new sample belongs is a common necessity in malware analysis. One prevalent approach involves subjecting the sample to a multi-engine antivirus scanner, such as VirusTotal. However, outcomes from these scanners can sometimes lack clarity and accuracy as malware is often tagged with generic labels like “generic”, offering little meaningful information. Additionally, malware creators may actively monitor the VirusTotal database, modifying their code or functions to evade detection. An alternative method for malware analysis involves executing the sample within a controlled sandbox environment, such as CuckooBox, to gather insights into the malware’s behavior and communication with callback servers. While this approach can yield valuable insights, it can be time-consuming and less efficient when dealing with extensive malware datasets. A distinctive and automated approach to malware analysis, as introduced in ref. [10], is shared code analysis or similarity check analysis. This technique compares two malware samples by quantifying the proportion of the recompiled source code they share. Unlike shared attribute analysis, which relies on external characteristics, shared code analysis swiftly and accurately classifies malware, particularly within large datasets. Nevertheless, it is crucial to assess the limitations of this method and utilize it in conjunction with other analysis techniques as needed. In the context of malware analysis and ransomware, ref. [11] offers a comparative exploration of various analysis approaches and ransomware typologies, shedding light on their respective behaviors and characteristics; Section 2.1, “Ransomware Detection Approaches”, outlines various techniques. Machine learning leverages known ransomware datasets for classification. Behavioral analysis observes malware execution, analyzing network activity, file operations, and system resource usage.

2.1. Ransomware Detection Approaches and Techniques

In the Machine Learning approach, machine learning algorithms analyze and categorize ransomware behavior. Trained on datasets of both known ransomware and benign samples, these algorithms identify new ransomware based on learned characteristics. Machine learning techniques, such as Decision Trees, Support Vector Machines, and Artificial Neural Networks, are applied. Advantages include adaptability to new ransomware variations and scalability for handling large datasets. However, accuracy hinges on dataset quality, diversity, and algorithm complexity. The Honeypot approach entails establishing networks or systems designed to attract and ensnare ransomware. These systems simulate vulnerability to lure ransomware attackers and monitor their activities. Benefits encompass real-time collection and analysis of new ransomware samples and the ability to discern attacker behavior trends and patterns. Nonetheless, Honeypots require substantial resources and maintenance and may not detect all ransomware types. The Statistical Analysis approach scrutinizes the statistical attributes of ransomware samples to uncover common patterns and features. Techniques like frequency analysis, entropy analysis, and n-gram analysis are employed. Advantages include rapid analysis of large datasets and the identification of shared patterns across diverse ransomware types. However, it may struggle with sophisticated or novel ransomware and could yield false positives if benign samples exhibit similar statistical characteristics. Each approach possesses its own merits and drawbacks, making them suitable for specific ransomware detection scenarios. The choice of approach should align with the requirements and constraints of the detection system. Careful consideration is vital when selecting the appropriate methodology.

2.1.1. Machine Learning

Machine Learning leverages algorithms grouped into categories like Bayesian, decision tree, dimension reduction, instance-based, clustering, deep learning, ensemble, neural network, regularization, rule system, and regression. These algorithms are utilized for ransomware detection by analyzing and classifying behaviors. Bayesian algorithms, rooted in Bayesian statistics, employ probabilistic models for event likelihood predictions, commonly applied in spam filters and malware detection systems. Decision tree algorithms

employ tree-like structures to make decisions based on predefined conditions or rules, often used for classifying malware. Dimension reduction reduces dataset features for easier analysis, aiding in identifying malware patterns and characteristics. Instance-based algorithms make predictions based on stored instances or examples, useful in recognizing malware patterns. Clustering algorithms group similar data points, employed to identify malware features. Deep learning utilizes artificial neural networks for pattern recognition. Ensemble algorithms combine multiple models to enhance accuracy, while neural network algorithms employ artificial neural networks for pattern detection. Regularization algorithms prevent overfitting in complex models. In ref. [12], a machine learning-based model distinguished ransomware from normal files and other malware, with an automatic detection model enabling the identification of new ransomware samples. Ref. [13] explored research projects employing machine learning and deep learning for ransomware detection. Ref. [14] utilized a digital DNA sequencing engine and AI machine learning network to classify ransomware into distinct families based on their “digital genomes”. Researchers in [15] employed hybrid multi-level profiling for a comprehensive forensic investigation of crypto ransomware. They introduce the concept of “behavioral chaining” and employ tools for mining associative rules and AI. Profiling ransomware behavior based on its chain ratio introduces a novel approach to creating unique ransomware signatures.

2.1.2. Honeypots

Honeypots are valuable tools for gathering information about attacks, including the identification of users and the extent of their activities, aiding in informed decision-making for defense strategies. The primary objective of deploying honeypots is to acquire insights into ongoing attacks and utilize that intelligence to bolster security measures. To enhance user awareness, email notifications are sent, occasionally advising users to disconnect network cables as a precautionary measure. This user training aspect adds an extra layer of security awareness, making honeypots an effective means to detect ransomware attacks. In ref. [16], the authors employed a combination of methods, including machine learning for grouping cases and Honeypots to capture potentially malicious packages. Classification tasks utilize Decision Trees and Support Vector Machine (SVM). The study suggests the potential of architectural solutions for malware detection. Ref. [17] introduced an Intrusion Detection Honeypot (IDH), comprising Honeyfolder, Audit Watch, and Complex Event Processing (CEP). IDH is designed to mimic vulnerability while also functioning as an early warning system, notifying users of suspicious file activity. Ref. [18] presented a deception method involving Honeyfiles and Honeytokens, designed to access compromised private files and detect hacking or ransomware attempts. The hypothesis explores the use of honeypots combined with machine learning for malware detection. In ref. [19], data from an Internet of Things (IoT) honeypot were effectively employed to train a dynamic machine learning model. This highlights the dynamic nature of honeypot-driven machine-learning techniques. Ref. [20] suggested a framework utilizing an Intrusion Prevention System (IPS) gateway, an analytical system, and honeypots to detect and identify ransomware. The framework encompasses six elements: IPS, gateway, static detector, dynamic detector, honeynet, and a notification component, collectively contributing to effective ransomware detection and user notification. These studies underscore the versatility and potential of honeypot-driven approaches, often combined with machine learning techniques, for enhancing ransomware detection and overall cybersecurity.

2.1.3. Statistics

To better understand the characteristics of ransomware, it may be possible to employ statistical analysis. One prominent method of detecting ransomware is using statistical analyses, which can identify unpredictable behavior and be used to flag the presence of encryption. Based on the frequency of opcodes in the portable executable file, the authors in ref. [21] proposed an approach for detecting malware. The study used a machine learning system to detect false positives, false negatives, true positives, and true negatives in

malware. While the authors in ref. [22] proposed a method for finding malware. This research employed a machine learning algorithm to identify malware with varying degrees of accuracy. The method of malware detection was developed by the authors using a similarity measurement algorithm. The proposed method was meant to boost malware detection times and throughput. This methodology has various advantages over others, including increased speed by using opcodes directly and improved detection outcomes from being immune to obfuscation and disassembly methods in ref. [23]. Another approach for malware was classification presented in ref. [24] inspired by the aesthetic similarities across viruses in the same family, this work proposes binary texture analysis over greyscale photos generated directly from malware executables. This technique provides statistical texture features of the second order over the graphical representation of malware. This strategy cannot be fooled by common methods of concealment (e.g., packing, code relocation, and encryption). Five malware detection metrics were assessed in the absence of ground truth, a real-world scenario that poses various technical challenges, the end goal was to develop fully automated, principled methods to assess these indications with the highest possible precision. Estimators of statistical significance were provided for the five measures used to identify malware. These statistical estimators were shown to be accurate by comparison to the known truth and fictional data. This large dataset was obtained from VirusTotal, and the estimators were then utilized to measure and quantify five metrics in ref. [25]. Several methods proposed in the literature make use of multiple strategies. The benefits and drawbacks of various ransomware detection strategies are summarized in Table 1.

Table 1. Comparison between ransomware detection approaches.

Ransomware Detection Approach	Ref.	Description	Advantages	Disadvantages
Machine Learning	[12–15]	<p>The most used machine learning techniques in ransomware detection include supervised learning, unsupervised learning, and semi-supervised learning. Supervised learning involves training a model on labeled data, where the input and output are both known. This allows the model to make predictions based on the relationships learned from the training data.</p> <p>Unsupervised learning involves training a model on data where the output is not known, and the model must find patterns and relationships within the data on its own. Semi-supervised learning is a combination of supervised and unsupervised learning, where the model is trained on a mix of labeled and unlabeled data.</p>	<p>One of the main advantages of using machine learning for ransomware detection is that it allows for the automatic identification of patterns and relationships within large datasets. This can be particularly useful for identifying new and emerging threats, as the model can learn from past data to identify patterns and make predictions about future threats.</p> <p>Machine learning algorithms can also be trained on a wide variety of data types, including text, images, and audio, which makes them useful for detecting ransomware in different formats.</p>	<p>Machine learning algorithms can be vulnerable to bias and can produce inaccurate results if the training data are not representative of the real-world data. They also require frequent retraining to ensure that they continue to perform well as the data distribution changes.</p>

Table 1. Cont.

Ransomware Detection Approach	Ref.	Description	Advantages	Disadvantages
Honeypot	[16–20]	Honeypots are a type of decoy system that is designed to attract and detect malware or cyber-attacks. They are used to lure attackers into a controlled and isolated environment, where their actions can be observed and studied. By setting up a honeypot, it is possible to monitor and track ransomware activity and identify new strains or variants of the malware.	One advantage of using a honeypot is that it allows researchers to gather valuable data and intelligence about the tactics, techniques, and procedures (TTPs) used by attackers. This information can be used to improve the effectiveness of ransomware detection and prevention measures. Additionally, honeypots can help mitigate the impact of ransomware attacks by preventing the malware from reaching the target system or data.	There are also some disadvantages to using honeypots. One potential issue is the risk of false positives, where legitimate activity is mistaken for malicious activity. Another issue is the cost and resources required to maintain and operate a honeypot, as well as the potential legal and ethical considerations. Additionally, honeypots may not be suitable for all types of environments or organizations and may not provide comprehensive protection against all types of ransomware attacks.
Statistical	[21–25]	The statistical analysis approach involves collecting and analyzing data about ransomware behavior to identify patterns and trends. This can be done through various methods, such as collecting data about the frequency and types of ransom demands, the types of files targeted, and the tactics used by ransomware operators.	The advantage of using statistical analysis is that it allows researchers to gain a deeper understanding of ransomware behavior and identify key trends that can inform prevention and detection efforts.	The disadvantage of this approach is that it relies on the availability of accurate and comprehensive data, which may be difficult to obtain in some cases. Additionally, statistical analysis may not be able to identify specific instances of ransomware in real time, making it less effective for immediate detection and response.

3. Background

In this section, we define and present the features that affect ransomware tracking and introduce the different static and dynamic features that have been used for ransomware tracking. In Section 3.1, we introduce the different types of ransoms and provide a brief history of ransomware. We also compare the key features, spreading techniques, exploitation, and ransomware families of different ransomware types, such as crypto worm, Ransomware-as-a-Service (RaaS), and Automated Active Adversary ransomware. We also discuss the role of APT attacks, such as the Shamoon data wiper malware, in ransomware infections. In Section 3.2, we discuss visualization techniques that are used to represent and analyze data in a visual form. In the context of ransomware classification, visualization techniques can be utilized to graphically represent the relationships and similarities between different ransomware samples. These techniques can provide a more intuitive and comprehensive understanding of the data, allowing analysts to identify patterns and trends that may not be immediately apparent through traditional methods of analysis. Some common visualization techniques that may be used in ransomware classification include scatter plots, heat maps, and network graphs. By using these techniques, analysts can effectively classify, and cluster ransomware samples based on their features and characteristics, enabling more accurate and efficient detection and analysis of these threats. Finally, in Section 3.3, we discuss the use of static and dynamic features in ransomware tracking systems and the challenges and opportunities that these features present. Overall, this section provides a comprehensive overview of the key features and techniques that are

used in ransomware tracking and classification as well as the challenges and opportunities that these approaches present.

3.1. Ransomware Types and History

Ransomware, classified as a type of malware, operates by encrypting a victim's files and subsequently demanding a ransom in exchange for restoring access to these files in ref. [26]. Notably, various categories of ransomware exist, each with unique characteristics. These categories encompass crypto worms in ref. [27], Human-operated Ransomware in ref. [28], Ransomware-as-a-Service (RaaS) in ref. [29], and Automated Active Adversary ransomware in ref. [30]. Table 2 encapsulates the essential features, propagation methods, exploitation strategies, and ransomware families associated with these diverse ransomware types. A specific subtype within the RaaS ransomware category is Advanced Persistent Threat (APT) attacks, typified by instances like the Shamoon data wiper malware in ref. [31]. APT-33, for instance, has employed such attacks in the Middle East and Europe, often driven by commercial or military motives. Notably, ransomware infections can originate from various sources in ref. [32], with the distribution percentages elucidated in ref. [33]. Figure 2 visually represents the primary sources of infection for most ransomware, which may include phishing emails, APT attacks, system vulnerabilities, drive-by downloads, and exploit kits. An in-depth exploration of the history of ransomware has been undertaken by the authors in ref. [34]. In Table 3, a chronological account of significant ransomware attacks is summarized, including the attack date, the responsible ransomware family, and the resultant damage. Broadly, ransomware can be categorized into two principal subgroups: locker ransomware in ref. [35] and crypto ransomware in ref. [36]. Locker ransomware restricts access to a device, often by imposing an additional password requirement to access the device. In contrast, crypto ransomware identifies and encrypts valuable data located on the victim's device.

Table 2. Comparison between ransomware malware behavior types.

	Crypto Worm	Human-Operated Ransomware	Ransomware-as-a-Service (RaaS)	Automated Active Adversary
Key Features	Self-propagating	Targeted attacks	Ransomware-as-a-Service model	Advanced evasion tactics
Spreading techniques	Wormhole	Social engineering	Email attachments, web links	Customized attack vectors
Exploitation techniques	Vulnerabilities in systems	Targeted vulnerabilities	Vulnerabilities in systems	Customized exploits
Detection modules	Antivirus	User awareness, network monitoring	Antivirus, network monitoring	Network monitoring, user awareness
Ransomware Family Example	WannaCry	Ryuk	REvil	SolarWinds

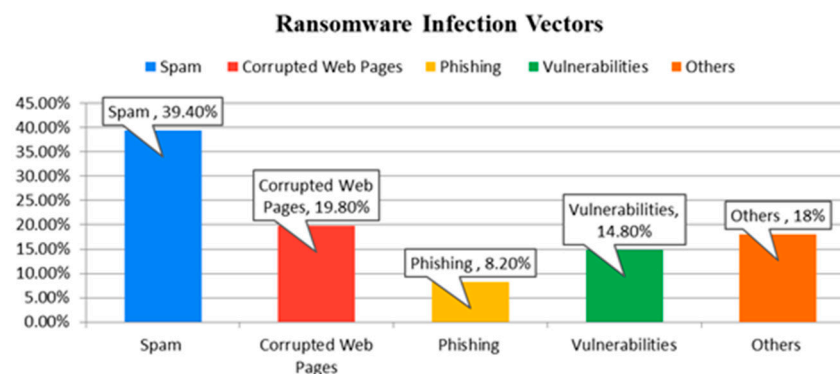


Figure 2. Ransomware infection vectors.

Table 3. Ransomware history timeline.

Date	Ransomware Family	Event Description
1989	AIDS	First ransomware, called “AIDS” or “PC Cyborg”, is released.
1991	PC Cyborg	It displays a message on the infected computer’s screen demanding payment.
2005	Gpcode	Gpcode ransomware uses strong encryption to lock users’ files, demanding payment to decrypt them.
2013	Cryptolocker	Cryptolocker ransomware is released, using encryption to hold users’ files hostage and demanding payment for the decryption key.
2014	Cryptowall	Cryptowall ransomware is released, using encryption to hold users’ files hostage and demanding payment for the decryption key.
2015	TeslaCrypt	TeslaCrypt ransomware has been released, targeting video game files and demanding payment for the decryption key.
2016	Locky	Locky ransomware is released, using encryption to hold users’ files hostage and demanding payment for the decryption key.
2017	NotPetya	NotPetya ransomware attack causes widespread damage, affecting thousands of computers and causing disruptions to various industries.
2018	LockerGoga	LockerGoga ransomware attack targets industrial control systems, causing disruptions to manufacturing and other industries.
2019	Ryuk	Ryuk ransomware targets government and healthcare organizations for large ransoms, causing widespread damage.
2020	REvil (Sodinokibi)	REvil (also known as Sodinokibi) ransomware attack causes widespread damage, affecting thousands of users and organizations.
2021	Babuk	Babuk ransomware attack targets government agencies and high-profile companies, threatening to release stolen data if a ransom is not paid.
2022	Egregor	Egregor ransomware attack causes widespread damage, affecting thousands of users and organizations.
2023	Black Cat	BlackCat ransomware is a type of malicious software that encrypts a victim’s files and demands a ransom for the decryption key.

3.2. Ransomware Classification with Visualization Techniques

Visualization techniques play a pivotal role in the realm of cybersecurity, offering valuable support in the classification and analysis of ransomware. Ransomware classification entails identifying and categorizing diverse ransomware types, relying on their distinctive characteristics and behaviors. Visualization methods, in this context, emerge as powerful tools for rendering large datasets of ransomware samples in a manner that is both intuitive and highly effective in ref. [37]. There are several different visualization techniques that can be used for ransomware classification, including scatter plots, heat maps, and tree maps. Scatter plots are a type of graph that plots data points on a two-dimensional grid, with one variable on the *x*-axis and the other on the *y*-axis. Scatter plots can be used to visualize the relationships between different features of ransomware samples, such as their encryption algorithms and file types, and can help analysts identify patterns and trends in the data. Heat maps are another type of visualization that uses color-coded scales to represent data values, with higher values being represented by warmer colors and lower values being represented by cooler colors in ref. [38]. Heat maps can be used to visualize the distribution of different features of ransomware samples and can help analysts identify clusters or outliers in the data. Treemaps are a type of visualization that uses nested rectangles to represent data values, with the size of the rectangles representing the value and the color representing the category in ref. [39]. Treemaps can be used to visualize the relationships between different categories of ransomware samples and can help analysts identify patterns and trends in the data. Visualization techniques are particularly useful for ransomware classification because they allow analysts to identify patterns and trends quickly and easily in large datasets and can help them identify similarities and differences between different ransomware families. By visualizing the data in this way, analysts can more easily identify clusters and outliers, and can use these insights to better understand the TTPs in ref. [40].

3.3. Ransomware's Features Tracking System

Our proposed ransomware classification, clustering, and detection system aims to provide a comprehensive approach to analyzing and classifying different types of ransomware. By using a combination of static analysis, dynamic analysis, and visualization techniques, our system can extract a wide range of features from ransomware samples and generate similarity matrices based on these features. These matrices can then be compared using a variety of comparison algorithms to identify similarities and differences between the samples, and the resulting similarity scores can be used to classify the samples into different categories, such as families, variants, and versions. One of the key features of our system is the ability to identify the amount of code shared by two malicious ransomware binaries before they are assembled. This can be especially useful for ransomware analysts and reverse engineers as it can help them better understand the TTPs of different ransomware families and identify common patterns and trends in ref. [41]. By providing a joint collaborative analysis platform, our system allows analysts to avoid having to redo tedious tasks that have already been done by others and enables them to work together more efficiently and effectively in ref. [42]. Overall, our proposed ransomware classification, clustering, and detection system offers a powerful and comprehensive approach to analyzing and classifying different types of ransomware. By providing a joint collaborative analysis platform and the ability to identify the amount of code shared by two malicious ransomware binaries, it helps analysts and reverse-engineers better understand the TTPs of different ransomware families and develop more effective defense and response strategies. Ransomware features tracking refers to the process of identifying and tracking the characteristics and behavior of different types of ransomware over time. This is an important task for cybersecurity professionals as it allows them to better understand the TTPs of different ransomware families and to develop more effective defense and response strategies in ref. [43]. There are several different approaches that can be used for ransomware feature tracking, including the Jaccard index, N-grams, and shared feature analysis. We classified malware samples into “bags of features” before comparing them; features could be strings, hashes, export and import tables, and API calls. Shared features between two malware samples are shown in Figure 3. Shared feature analysis involves identifying common characteristics or behaviors shared by different malware samples. This can be done using techniques such as static analysis, dynamic analysis, and machine learning, and can be particularly useful for tracking the evolution of ransomware families by analyzing the shared features of different ransomware samples in ref. [44].

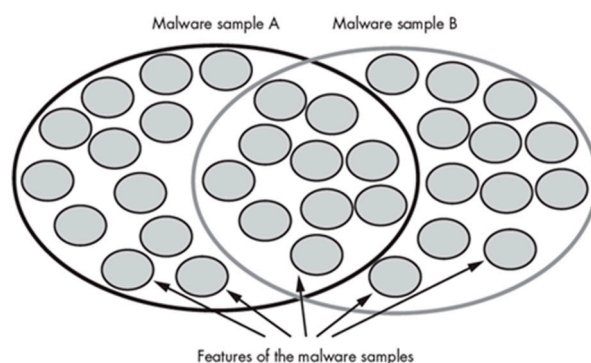


Figure 3. Shared features between two malware samples [45].

The Jaccard index is a measure of the similarity between two sets of data in ref. [46]. It is calculated by dividing the size of the intersection of the two sets by the size of the union of the two sets. The Jaccard index is often used in cybersecurity to measure the similarity between different malware samples, and it can be particularly useful for tracking the evolution of different ransomware families over time. By calculating the Jaccard index for different pairs of ransomware samples, analysts can identify how similar or dissimilar

they are and can use this information to better understand the TTPs of the different families. The Jaccard index has emerged as the most generally adopted—and with good reason. It quantifies the degree of overlap between two sets of malware features simply and sensibly, providing us with the percentage of unique features common to both sets normalized by the percentage of unique features in each group in ref. [47] (JI = intersection length/union length). The Jaccard Index explanation is shown in Figure 4.

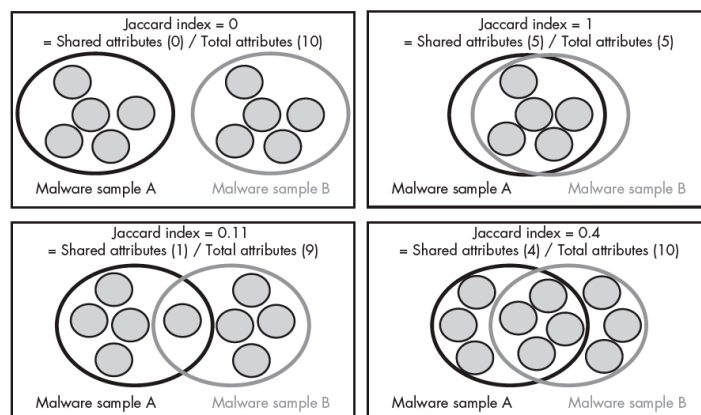


Figure 4. Jaccard Index between two malware samples.

Using N-grams to track the evolution of ransomware families can be a powerful tool for cybersecurity professionals in ref. [48]. By extracting subsequences of specific lengths from sequential data and comparing them using a similarity function, it is possible to determine the level of code commonality between two malware samples. This can be especially useful for identifying patterns and trends in the TTPs of different ransomware families and can help analysts develop more effective defense and response strategies. The similarity function used in this process should have certain properties to ensure accurate and reliable results. It should produce a normalized value that allows all similarity comparisons to be made on the same scale, and it should be able to accurately estimate the amount of code sharing between two samples. Additionally, it should be able to provide insight into why it performs well in modeling code similarities. Overall, the use of N-grams and a similarity function can be an effective way to track the evolution of ransomware families and better understand their TTPs. By extracting and comparing subsequences of specific lengths, analysts can identify common patterns and trends and can use this information to develop more effective defense and response strategies in ref. [49]. We employ a similarity function with the following properties to determine the level of code commonality between two malware samples shown in Figure 5. In the provided figure, each number corresponds to a distinct malware sample included in the analysis. The purpose of these numbers is to uniquely identify and label each malware instance for clarity. The arrows in the figure represent the presence of similar n-gram features between different malware samples. Specifically, the direction of the arrows indicates the connection from the source malware sample to the target sample, demonstrating a shared set of n-gram features. This visual representation highlights the commonalities in the n-gram patterns found in the corresponding malware instances. By examining the arrows and associated numbers, one can gain insights into the relationships and similarities among the various malware samples based on their n-gram features. This analysis aids in understanding the potential connections and patterns within the dataset, contributing to a more comprehensive comprehension of the malware landscape under investigation.

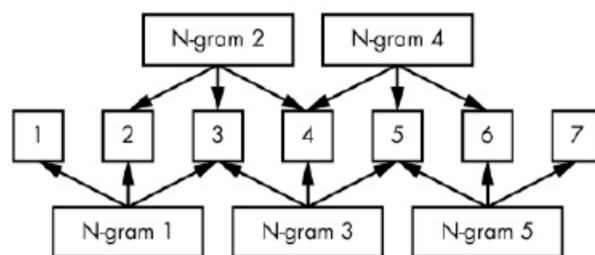


Figure 5. N-gram extracted from ransomware samples.

4. Experimental Work and Detection Scheme

In this section, we present the experimental work done to study ransomware visualization techniques and shared static and dynamic features between different ransomware samples. Ransomware visualization techniques are presented in Section 4.1, while shared static and dynamic features are presented in Sections 4.2 and 4.3 respectively. In Section 4.4, our lab setup is presented. Time complexity is presented in Section 4.5. Finally, in Section 4.6, we present the results from static and dynamic analyzers.

4.1. Visualization Techniques

In our approach to using visualization techniques to classify and analyze ransomware samples, we started by selecting a dataset of ransomware samples (most matched ones) and then applied a similarity matrix using a static and dynamic analyzer to find a fast and suitable way to use it in our final approach. To identify the most similar samples, we used a cluster engine to analyze the data and report the samples with the highest level of similarity. We then used static and dynamic analysis techniques to generate a similarity matrix for each group of samples. This matrix allowed us to visualize the relationships between the different samples and identify patterns and trends in the data. Once we had generated the similarity matrix, we used it to validate the query-sample similarity with the matched samples. This helped us to confirm that the samples in the first group were indeed the most similar ones and allowed us to identify any discrepancies or errors in the data.

Constructing nodes and connections between them helps to view and graph the data's connections. In other words, each sample is a node, and we may connect them and declare they are comparable if they have similar DLL imports.

- The cluster engine reported the most similar samples from the set.
- There is a need to validate the query-sample similarity with the matched samples.
- It is also important to reveal intelligence from the data and discover the patterns.

The graphical representation in Figure 6 elucidates the Vendors Detection for a collection of ransomware samples. It is worth emphasizing that not all security vendors have uniformly detected every sample within this dataset. This observation underscores the inherent variability in ransomware detection rates across different security solutions, thereby emphasizing the critical need for robust and comprehensive cybersecurity strategies. In the ensuing discussion, we will delve deeper into the implications of these detections. The samples characterized by a consistent segment count are indicative of non-packed samples, reflecting their unaltered and original nature within the dataset. This differentiation is instrumental in our analysis of the dataset's composition and assists in identifying potential trends or variations among the samples. The numerical results for visualization techniques can be found in Table 4.

The data depicted in Figure 7 reveals a noteworthy observation concerning the sample sizes utilized within the context of this study. It is evident from the graphical representation that a predominant portion of the collected samples exhibited uniformity in their respective sizes.

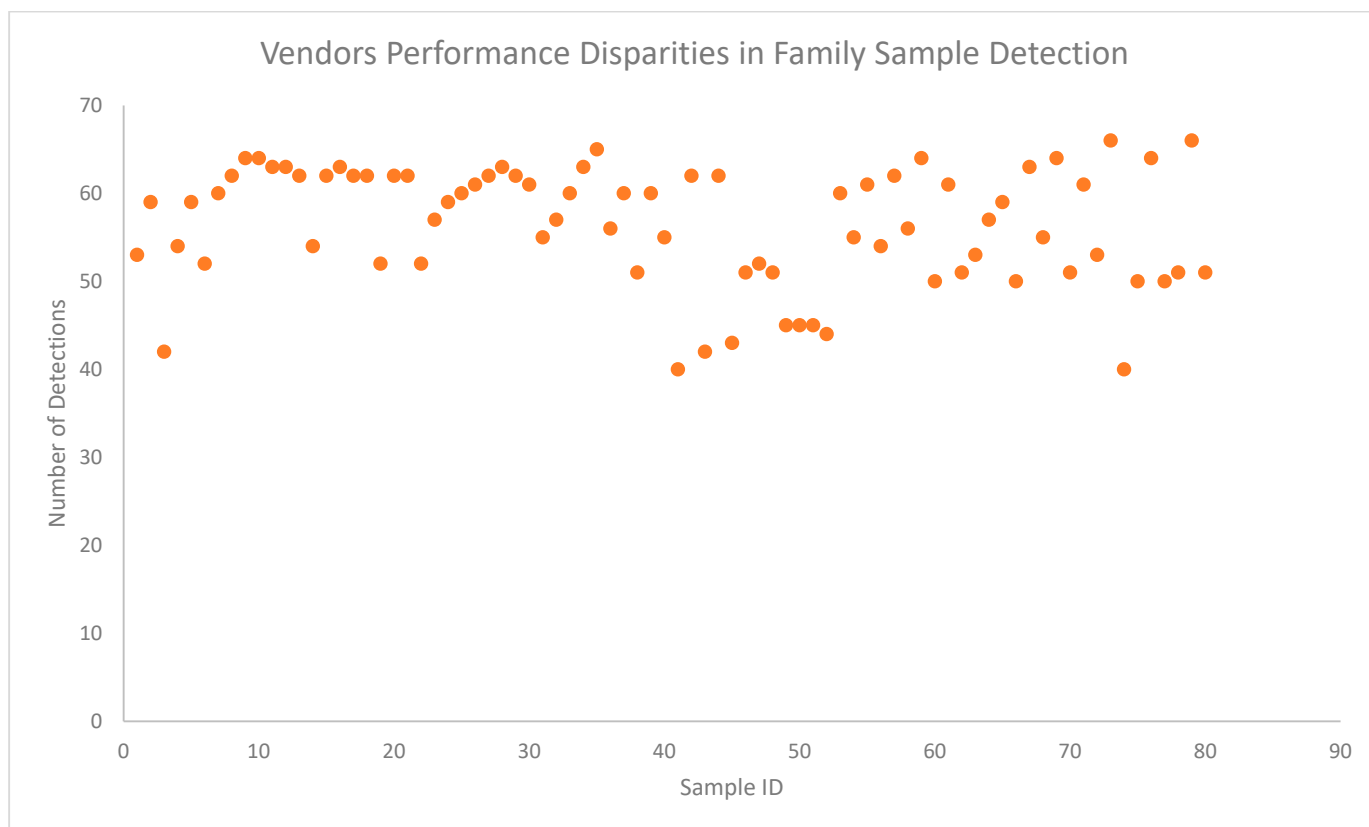


Figure 6. Vendors Performance for Samples Detection.

Table 4. Visualization of numerical results.

Number of Samples	80 Samples
Vendor Detection	100% (40 + vendors) detectability
Sample Size	57 samples with 55,200 bytes 23 samples with 55,300 bytes
IAT Count	111 imports
Internal Disassembled Functions Count	185 functions
Sample Segments	5 sections

Figure 8, which illustrates the counts of the Import Address Table (IAT) across various samples, unequivocally highlights a remarkable consistency in these counts across all the analyzed samples. This compelling consistency within the IAT counts underscores the robustness of this static feature as a key determinant for effectively classifying and clustering ransomware samples within our laboratory experiments.

Figure 9 provides an insightful depiction of the counts of internal functions across the examined samples. Notably, a striking similarity becomes evident as one observes the distribution of these internal function counts. This remarkable uniformity among the samples in terms of internal function counts further solidifies the findings from our laboratory experiments, affirming the robustness of our research results.

In Figure 10, we observe the counts of segments within portable executable (PE) files. This analysis allows us to discern between packed and non-packed samples in our dataset.

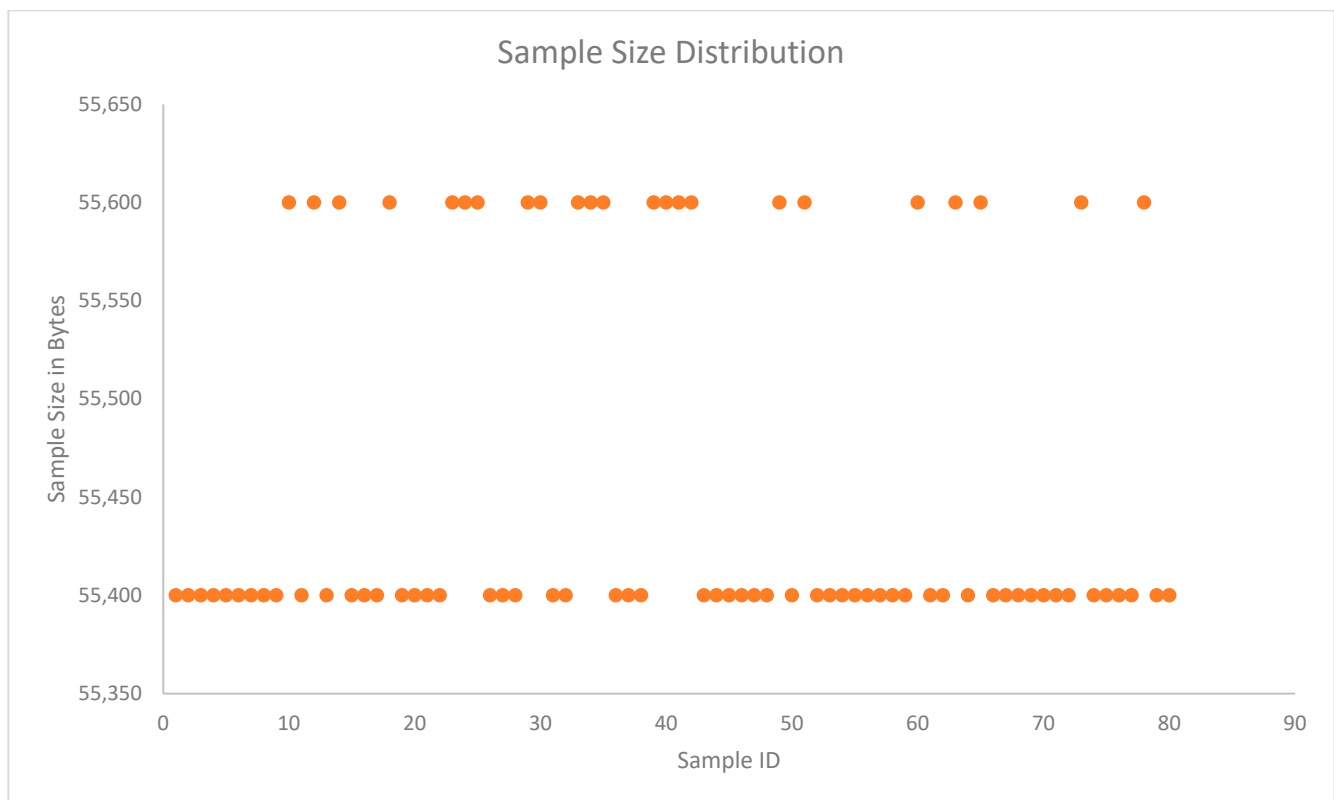


Figure 7. Sample Size Distribution.

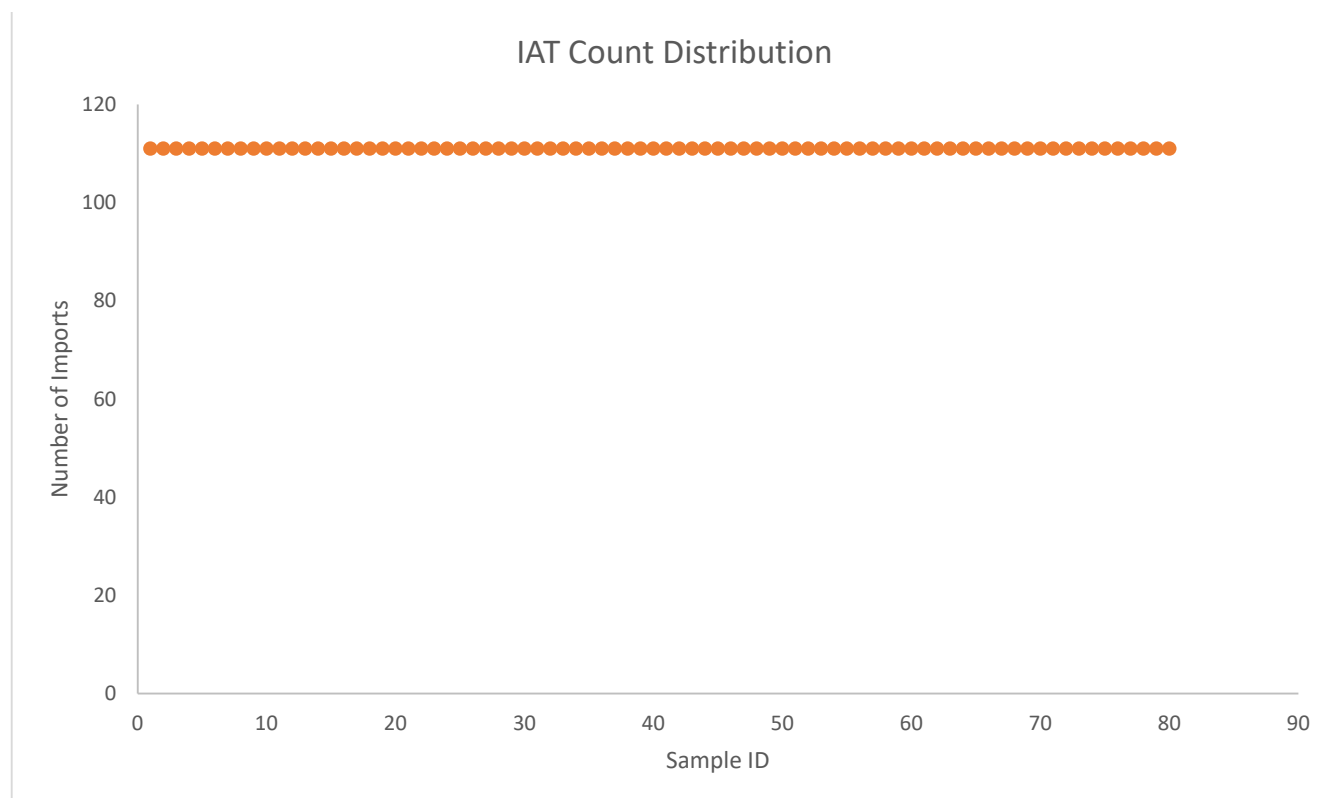


Figure 8. IAT Count Distribution.

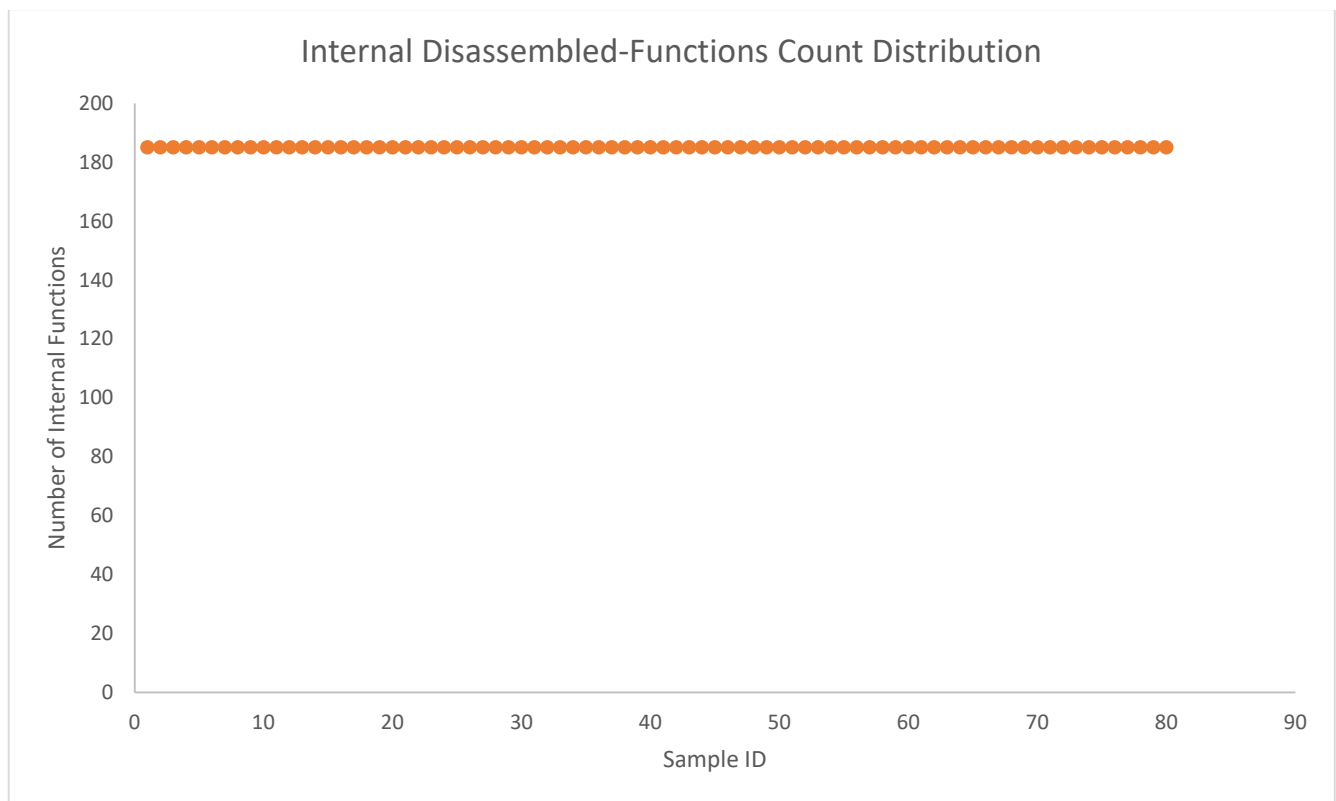


Figure 9. Internal Disassembled-Functions Count Distribution.

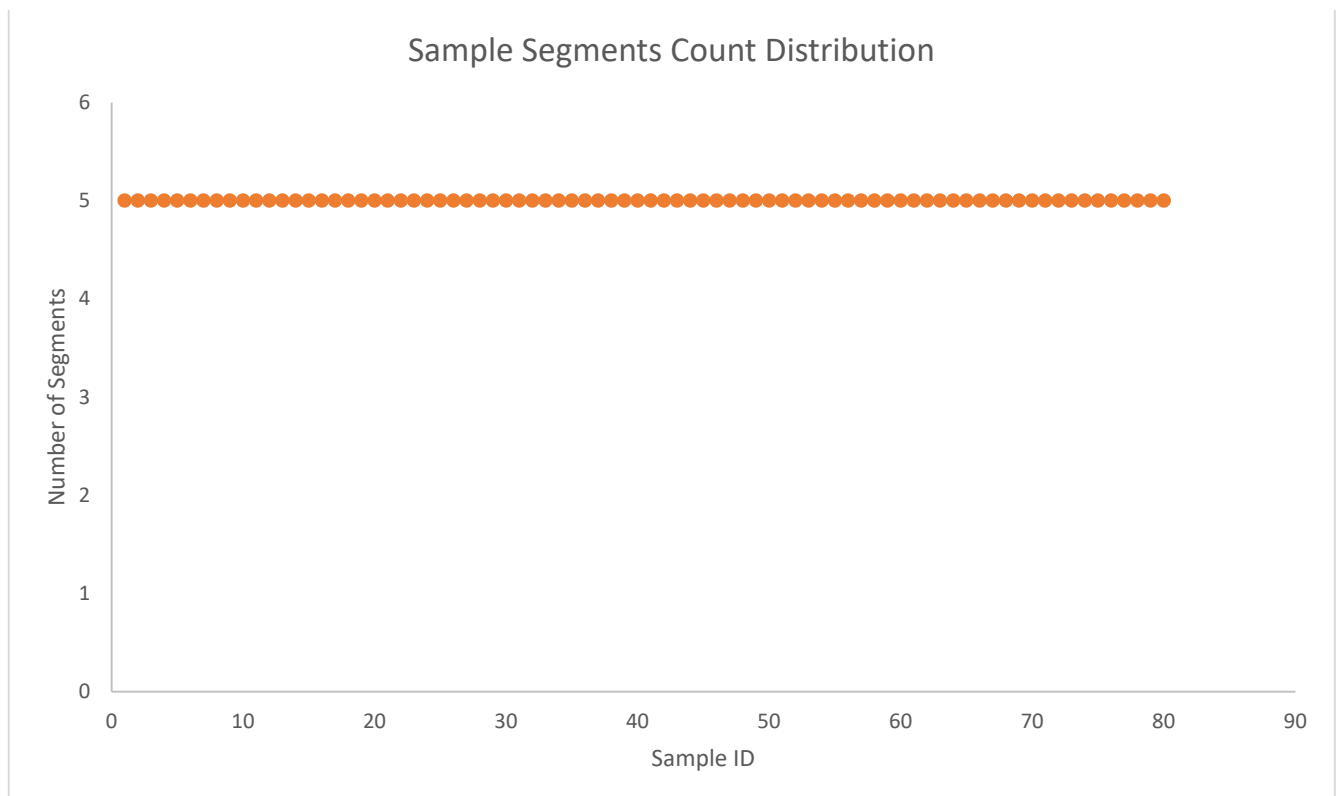


Figure 10. Sample Segments Count Distribution.

Visualization comparison is a powerful tool for analyzing and classifying different types of ransomware. By visualizing the relationships between different samples, it is possible to identify common patterns and trends and to better understand the TTPs of different ransomware families. In our research, we found that the Import Address Table and Internal Function count were the most effective features for finding the similarity between different ransomware samples. The Import Address Table is a data structure in a Windows executable file that contains the addresses of imported functions from other dynamic link libraries (DLLs). The Internal Function count is the number of functions defined in the sample. By analyzing these features, we were able to identify common patterns and trends in the data and to better understand the TTPs of different ransomware families. We also found that obfuscated or packed samples could be identified by comparing the PE File Segments of different samples. The PE File Segments are the different parts of a Windows executable file that contain code, data, and other information. By comparing the Segment count of different samples, we were able to identify those that had been packed or obfuscated as the default Segment count for a sample is typically around five.

4.2. Static Ransomware Classification System

The ransomware classification and detection system that we have proposed is designed to analyze and classify different types of ransomware using a combination of static and dynamic analysis techniques. It works by submitting samples to the system through a Python API and then applying a classification and clustering algorithm using disassembled binaries to extract various features and generate a mnemonic N-gram. The system then calculates the Jaccard similarity between different samples and performs clustering on those samples to group them into classified clusters. This allows analysts to identify common patterns and trends in the TTPs of different ransomware families and to better understand the relationships between different samples. The proposed static ransomware classification and detection system includes a System Controller with APIs for submitting ransomware samples to the static analysis server and querying the MongoDB NoSQL database for various properties. The Analyzer server on Windows retrieves the static characteristics and attributes from the given samples through the disassembler process. Overall, the proposed ransomware classification and detection system provides a comprehensive approach to analyzing and classifying different types of ransomware. By using a combination of static and dynamic analysis techniques, it can extract a wide range of features from ransomware samples and generate a mnemonic N-gram to identify common patterns and trends in the TTPs of different ransomware families. By performing clustering on the samples, it also allows analysts to better understand the relationships between different samples and to develop more effective defense and response strategies. The proposed static ransomware classification and detection system diagram is illustrated in Figure 11.

4.3. Dynamic Ransomware Classification System

In this paper, we have presented a novel and efficient malware indexing system that provides a range of search and analysis capabilities for ransomware analysts and reverse engineers. The system is designed to analyze native binaries and to identify common patterns and trends in the TTPs of different ransomware families. One of the key features of the system is its ability to perform similarity checks between different samples and to classify and cluster them based on their features and attributes. This allows analysts to identify commonalities and differences between different ransomware families and to better understand the relationships between different samples. One limitation of the system is that it is mainly designed to analyze native binaries and may not be as effective at analyzing packed or obfuscated samples. Many malware authors use packing techniques to hide and obscure their code, making it more difficult to analyze and classify. However, the system is still able to provide useful insights and intelligence for analysts working with packed or obfuscated samples, as it relies on hybrid data from static and dynamic analysis to identify common patterns and trends in the TTPs of different ransomware

families. Overall, the proposed malware indexing system is a valuable tool for analysts and reverse engineers working with ransomware samples. By providing a range of search and analysis capabilities, it helps analysts to identify common patterns and trends in the TTPs of different ransomware families and to better understand the relationships between different samples. This enables them to develop more effective defense and response strategies and to more effectively mitigate the threat posed by ransomware. The proposed dynamic ransomware classification and detection system diagram is illustrated in Figure 12.

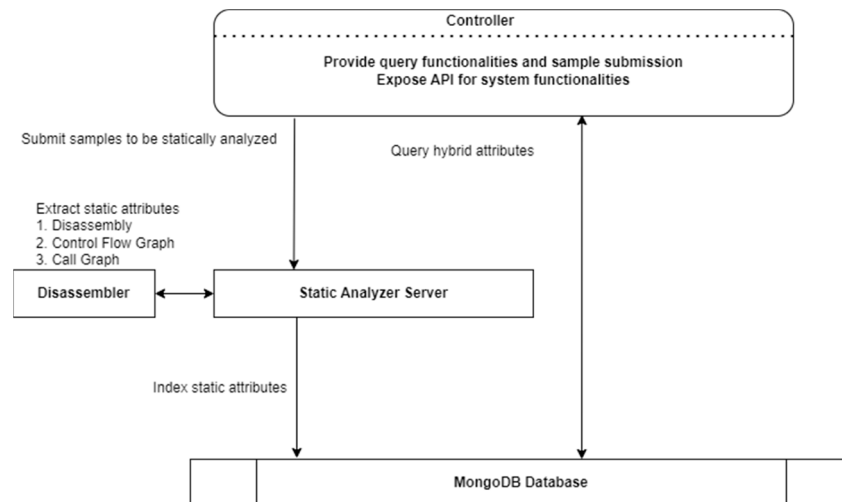


Figure 11. Static ransomware classification system block diagram.

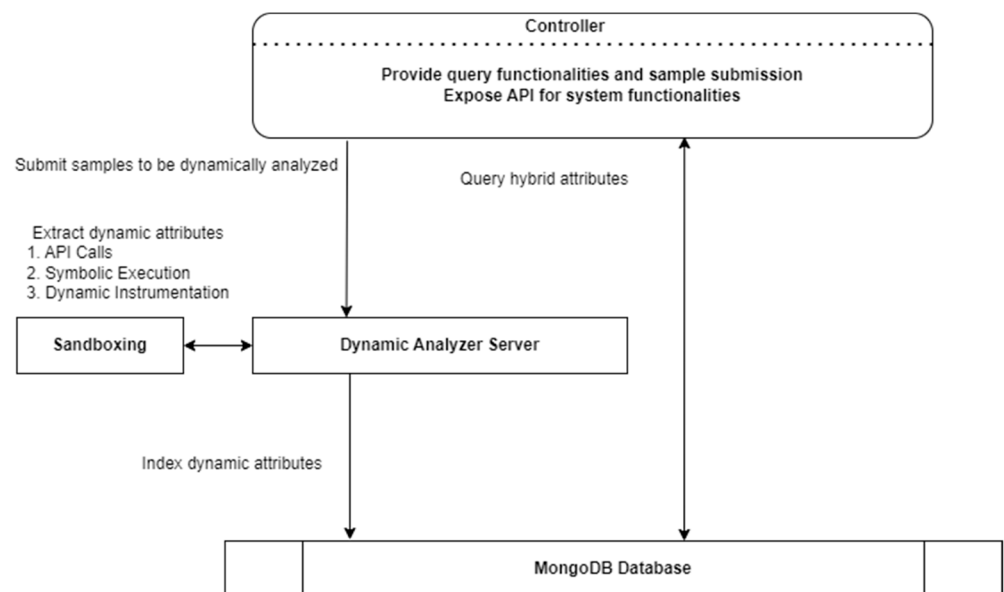


Figure 12. Dynamic ransomware classification system block diagram.

4.4. Lab Setup

The lab setup for our proposed malware indexing system includes a range of machines and tools that are designed to support the analysis and classification of different types of malware presented in Table 5, including ransomware. The controller is the central component that interacts with all the analysis engines and performs database queries to retrieve relevant samples in response to analyst requests. It also submits samples to the analyzer server for analysis. The analyzer server is responsible for disassembling executable binaries into a set of static features and extracting pertinent properties and features that the

controller uses to classify the binaries. The disassembler is an important part of the analyzer server, as it is responsible for breaking down the samples into their constituent parts and extracting the relevant features. MongoDB is used to index all the extracted features so that they can be queried and analyzed using the Jaccard Index similarity function. This allows analysts to identify common patterns and trends quickly and easily in the TTPs of different ransomware families and to better understand the relationships between different samples. VirusTotal is another important tool in our lab setup, as it provides a range of clustering and similarity-matching capabilities that allow analysts to group and classify different samples based on their features and attributes. It also includes a comprehensive graph view that enables analysts to visualize the relationships between different malware objects and to better understand how they are associated with specific campaigns. Overall, the lab setup for our proposed malware indexing system is designed to provide analysts with the tools and resources they need to effectively analyze and classify different types of malware, including ransomware. It includes a range of machines and tools that support the disassembly and analysis of executable binaries, as well as powerful indexing and querying capabilities that enable analysts to identify common patterns and trends in the TTPs of different malware families. The full-matched ransomware classification and detection system diagram is illustrated in Figure 13.

Table 5. Machines and tools used in dynamic analyzer lab setup.

Components	Purpose	Technology	Output
System Controller	The main component that interacts with all the analysis engines and performs database queries to retrieve relevant samples in response to analyst requests is the controller. This component is the central hub of the malware indexing system and is responsible for coordinating the analysis and classification of different types of malware, including ransomware. It receives requests from analysts and communicates with the various analysis engines to gather the necessary data and information. The controller also interacts with the database to retrieve relevant samples based on the analyst's requests, ensuring that the analyst has access to the most up-to-date and relevant data. Overall, the controller plays a critical role in the operation of the malware indexing system, enabling analysts to access the data and information quickly and easily in order to effectively analyze and classify different types of malware.	APIs modules	Hybrid attributes
Analyzer Engine	The analyzer server is responsible for converting executable binaries into a set of static features through disassembly to extract relevant static attributes that can be used by the controller to classify the binaries. This process involves breaking down the samples into their constituent parts and extracting the relevant features, such as function names, imported libraries, and other characteristics. The analyzer server uses a disassembler tool to perform this process, which is an important part of the overall malware indexing system. In addition to extracting static features, the analyzer server is also responsible for converting the set of binaries into dynamic attributes through sandboxing. Sandboxing involves running the samples in a controlled environment and observing their behavior to extract dynamic attributes, such as network traffic, file system changes, and other activities. These dynamic attributes can be used to supplement the static features to classify and analyze the samples more accurately. Overall, the analyzer server plays a critical role in the operation of the malware indexing system, providing the necessary data and information that enables analysts to effectively analyze and classify different types of malware.	Disassembling Decompiling	Static features: Function names Imported libraries Strings and string patterns File size and metadata Dynamic features: Network traffic and connection information API Calls File system changes Process and thread behavior Memory modifications Registry changes System and library calls

Table 5. Cont.

Components	Purpose	Technology	Output
MonogoDB	MongoDB is a document-based store for all the extracted attributes by the analysis engines. It is an NoSQL database that is designed to handle large amounts of data and to support flexible and scalable data models. In the context of the malware indexing system, MongoDB is used to store all the extracted static and dynamic features from the analyzer server, as well as any additional metadata or information about the samples. These data are then used by the controller to perform various analysis and classification tasks, such as calculating the Jaccard Index similarity between different samples or clustering the samples into different categories. By providing a centralized repository for all the extracted features, MongoDB enables analysts to access and analyze the data more easily, and to perform complex queries and searches across the entire dataset.	Receive controller queries for binaries features and attributes.	Hybrid attributes
Disassembler (Ghidra 11.0)	Ghidra used to extract static features from malware samples. Ghidra is a powerful and feature-rich tool that is developed and maintained by the National Security Agency (NSA). It provides a wide range of features and functionality that are useful in the analysis of malware, including the ability to disassemble code, view and edit assembly instructions, and perform various other tasks. In the context of the malware indexing system, Ghidra could be used to disassemble the samples to extract static features such as function names, imported libraries, and other characteristics. These features could then be stored in the MongoDB database for use in various analysis and classification tasks.	Disassembling	Static features of the samples
Sandbox (Cuckoo)	Cuckoo Sandbox is a powerful and widely used tool that is used to extract dynamic features from malware samples. It is an open-source sandboxing platform that allows users to analyze the behavior of malware in a controlled environment. By analyzing the behavior of malware in a sandbox, analysts can extract various dynamic features such as network traffic, file system changes, and other characteristics that may not be visible through static analysis alone. In the context of the malware indexing system, Cuckoo Sandbox could be used to extract dynamic features from the samples and store them in the MongoDB database. These dynamic features could be used in conjunction with the static features extracted through disassembly to provide a more complete picture of the malware's behavior and capabilities.	Sandboxing	Dynamic features of the samples
Python PyCharm (2023.3.2)	Python PyCharm is a powerful integrated development environment (IDE) that is often used in the development of Python programs. It is a popular choice among developers due to its feature-rich set of tools and capabilities, including code completion, debugging, testing, and deployment. In the context of the malware indexing system, Python PyCharm could be used to develop and maintain the various components of the system, including the analysis engines, the controller, and the database.	Python IDE	System scripts Database connections script Sample submissions script
VirusTotal	VirusTotal is a powerful tool that allows users to scan files and URLs against a vast array of antivirus engines and other security tools. This enables analysts to identify malicious content and track the evolution of malware over time quickly and easily. In our study, we used VirusTotal to collect a large dataset of ransomware samples, which we then analyzed using various techniques such as static analysis, dynamic analysis, and visualization. By leveraging the power of VirusTotal, we were able to gather a large and diverse set of ransomware samples quickly and efficiently, which allowed us to more accurately and effectively classify and cluster the samples. Overall, VirusTotal proved to be a valuable resource in our study, and it is a powerful tool that is widely used in the field of malware analysis	VirusTotal Hunting feature	Collected ransomware samples

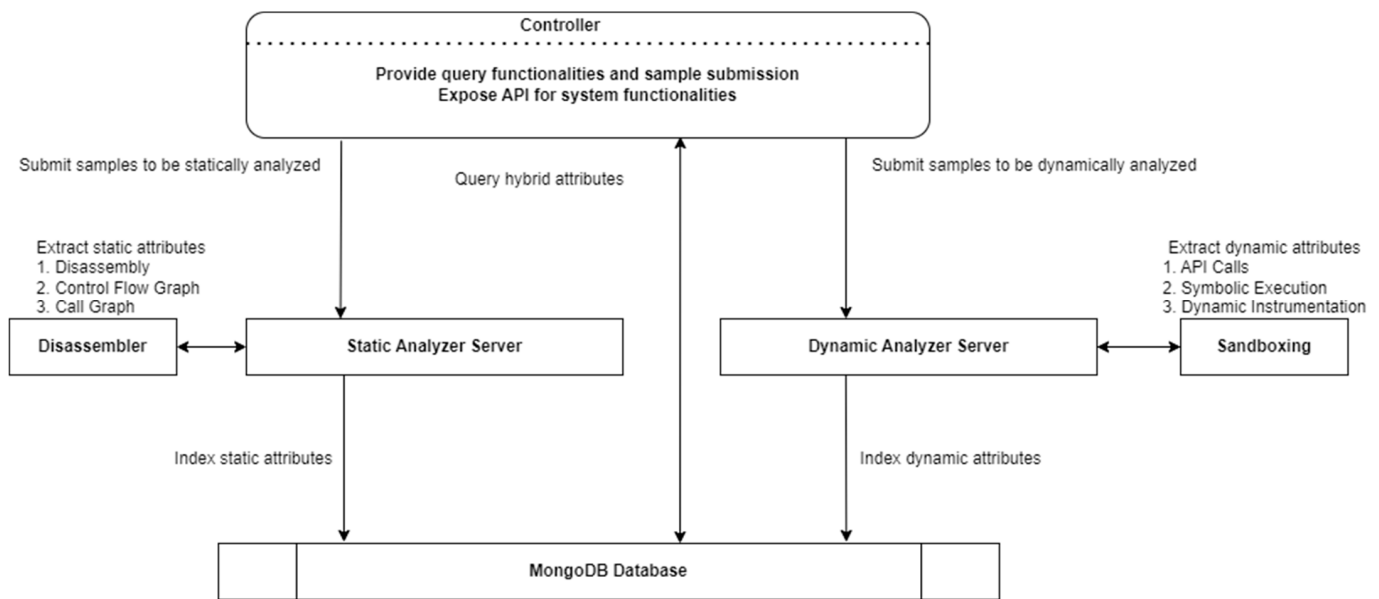


Figure 13. Full-matched ransomware classification system block diagram.

4.5. Time Complexity

In the realm of malware detection and classification, the consideration of time complexity is paramount, as the potential damages inflicted by threats may occur before a detection or classification system has the chance to identify them. Understanding the time efficiency of our system is crucial for ensuring timely responses to potential threats [50–52].

4.5.1. Static vs. Dynamic Analysis

Our system incorporates both static and dynamic analysis approaches to strike a balance between speed and accuracy. Static analysis, while fast, may exhibit reduced accuracy on certain samples. On the other hand, dynamic analysis, although accurate, tends to be slower in terms of analysis time.

4.5.2. Hybrid Approach for Optimal Time Efficiency

To address this trade-off, we have implemented a hybrid approach that combines the strengths of both static and dynamic analyses. This hybridization aims to achieve an optimal average time complexity for our system.

Static analysis time: approximately 5–10 s per sample.

Dynamic analysis time: approximately 30–60 s per sample.

By integrating static and dynamic analyses, we have achieved an average time complexity of 2–5 s per sample. Additionally, samples that have been previously analyzed and added to our database incur zero seconds of analysis time during subsequent evaluations. This strategic combination enables us to deliver efficient and accurate results within a reasonable time frame.

4.5.3. Continuous Database Augmentation

To further enhance the time efficiency of our system, we encourage users to continuously contribute samples to our database. By doing so, the likelihood of encountering previously analyzed samples increases. Consequently, the analysis time for these samples becomes virtually instantaneous, offering an additional layer of efficiency.

4.5.4. Proactive Sample Analysis

Users are also encouraged to submit suspicious samples to our system for analysis before execution. This proactive approach enables preemptive identification of potential threats, contributing to an overall improvement in system responsiveness.

In conclusion, our approach to time complexity involves a thoughtful integration of static and dynamic analyses, coupled with continuous database augmentation and proactive sample analysis. This multifaceted strategy ensures a swift and accurate response to emerging threats in the ever-evolving landscape of malware detection.

4.6. Results

The results of our analysis show that the use of minhash and Jaccard index for feature comparison is an effective method for accurately estimating the degree of code sharing between different ransomware samples. By applying minhash to the strings, Import Address Table, and API call features extracted from our ransomware samples, we were able to identify highly similar samples with a high degree of accuracy. This approach allowed us to cluster the samples into distinct groups, enabling us to identify relationships between different ransomware families and variants more easily. In addition to the minhash and Jaccard index, we also employed other visualization techniques, such as the use of graph networks and dendrograms, to further aid in the analysis and interpretation of the data. These techniques allowed us to visually explore the relationships between different malware samples and identify patterns and trends that would have been difficult to discern using other methods. In our proposed approach, the first step is to store the ransomware samples in a database or repository. This can be done by manually collecting the samples or using an automated tool to gather them from various sources such as online scanners or honeypots. Next, the samples are indexed using a variety of features such as strings, Import Address Table, or API calls. These features are extracted from the samples using static or dynamic analysis techniques and stored in the database for later use. Once the samples are indexed, analysts can search for specific samples or groups of samples using various search criteria such as ransomware family, encryption algorithm, or date of discovery. Finally, the similarity between the samples can be visualized using various techniques such as clustering or similarity matrices. These visualizations can help analysts quickly understand the relationships between different ransomware samples and identify patterns or trends in the data.

i. Strings-Based Similarity

We propose a method for identifying the similarity between different ransomware samples using strings as a feature. By extracting all contiguous printable sequences of characters from the samples and generating the Jaccard index between all pairs of ransomware samples based on their common string relationships, we can compute the strings-based ransomware similarity. Strings taken from a binary tend to be format strings established by the programmer, which compilers in general do not transform, regardless of which compilers the ransomware authors use or what parameters they provide the compilers. This strategy allows us to bypass the compiler difficulty and accurately identify similarities between different ransomware samples. The similarity matrix generated using extracted static strings as a feature is illustrated in Figure 14.

In our static analysis, the absolute time required per sample is consistently 5 s, indicative of the efficiency of our static analyzer. Additionally, the absolute Jaccard index for similarity among the samples is 0.3. This Jaccard index value highlights the fast-processing nature of our static analysis; however, it is important to note that a Jaccard index of 0.3 signifies a lower level of accuracy in capturing similarities between the samples. This trade-off between speed and accuracy is a key consideration in our approach, aiming to strike a balance that aligns with the requirements of timely detection.

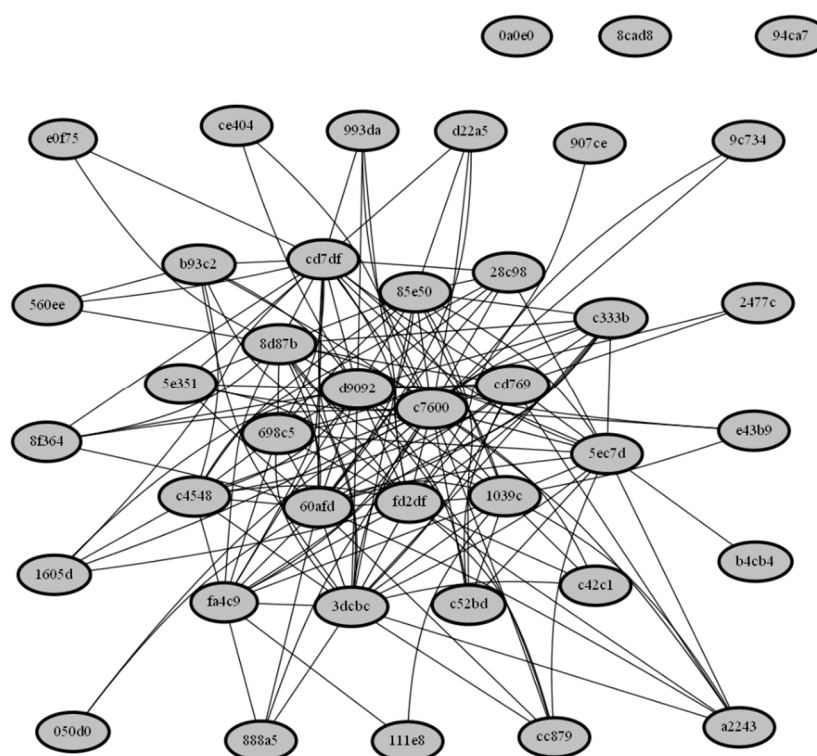


Figure 14. The similarity matrix generated using string features.

ii. Import Address Table–Based Similarity

Ransomware analysts and reverse engineers can use the Import Address Table (IAT) feature to identify the shared code between different ransomware samples. By comparing the IAT of two samples, analysts can determine the extent to which the samples use the same imported DLLs and functions. This information can be useful in identifying the relationships between different ransomware families and in understanding the evolution of individual families over time. To generate the IAT-based similarity matrix, analysts can extract the IAT from each sample and compute the Jaccard index between all pairs of samples based on their common IAT entries. The resulting matrix can then be visualized using a variety of techniques, such as clustering or network analysis, to identify patterns and trends within the data. By using the IAT feature in combination with other static and dynamic analysis techniques, analysts can gain a more comprehensive understanding of the relationships between different ransomware samples and can more effectively classify and cluster them for further analysis. Overall, the use of the IAT feature in ransomware analysis can greatly improve the efficiency and accuracy of malware classification and clustering efforts. The similarity matrix generated using the extracted static Import address table as a feature is illustrated in Figure 15. In our import address table (IAT) analysis, the absolute time required for processing each sample falls within the range of 5 to 10 s. This indicates the efficiency of our IAT analysis, striking a balance between speed and comprehensive examination. Notably, the absolute Jaccard index for similarity among samples in the context of IAT analysis is 0.86. This high Jaccard index value attests to the accuracy of our IAT analysis, showcasing its effectiveness in capturing similarities between samples. This combination of relatively fast processing time and a high Jaccard index underlines the efficacy of our approach in achieving both speed and accuracy in import address table analysis.

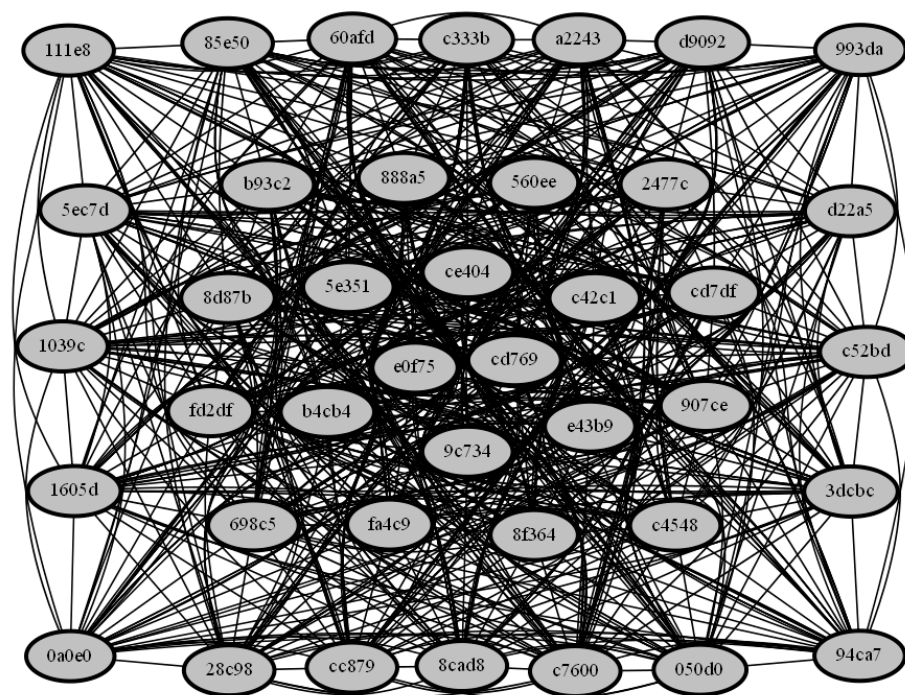


Figure 15. The similarity matrix generated using the Import Address Table (IAT) feature.

Ransomware's clustering is useful for grouping a large set of samples into a known or unknown number of groups or clusters, with objects in each cluster having a high degree of similarity and objects in other clusters being dissimilar. We proposed an efficient malware indexing system that provides search functionalities, similarity checking, and sample classification and clustering. The system mainly targets native binary files. The indexing engine depends on hybrid data from static features extraction, comparing different ransomware families to find the similarity matrix between those samples. We compared different static features by checking the similarity matrix for different ransomware families. Our research has proven that the Import Address Table (IAT) is the best feature for finding similar ransomware samples. The limitations in finding similarities between ransomware samples are the classification and clustering of the packed samples. Therefore, we focused on using a dynamic analyzer integrated with sandboxing to extract dynamic features like API calls. Using dynamic analyzer and static analyzer features and comparing different features-based similarity matrices will help in clustering and classifying packed and unpacked ransomware samples.

iii. API calls-Based Similarity

To find similarities between ransomware samples, we utilized API calls as a dynamic feature. By analyzing the API calls made by a sample during runtime through sandboxing, we were able to extract valuable information about the sample's behavior and use it to compare with other samples. This method proved particularly effective in identifying packed samples, which can often be difficult to classify using static features alone. Using API calls as a dynamic feature allowed us to accurately cluster and classify a large dataset of ransomware samples, including both packed and unpacked samples. By comparing the API call similarity matrix between different ransomware families, we were able to identify shared behavior and characteristics that helped us better understand the relationships between different samples. The similarity matrix generated using extracted dynamic API calls as a feature is illustrated in Figure 16. In our dynamic analysis of method API calls, the absolute time required for processing each sample typically ranges from 30 s to 60 s, contingent upon the complexity of the sample. Despite the relatively longer processing

time, this method is designed to provide a thorough and detailed analysis of the dynamic behavior of samples.

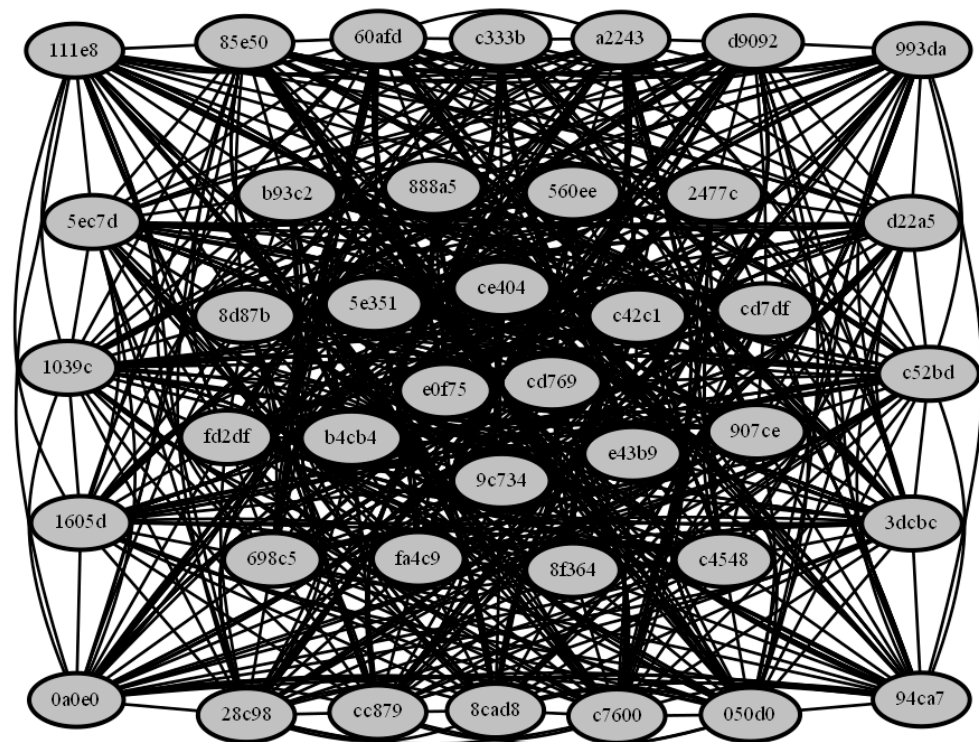


Figure 16. The similarity matrix generated using the API Call feature.

Remarkably, the absolute Jaccard index for similarity among samples in the context of dynamic analysis method API calls is 1. This perfect matching Jaccard index signifies full similarity, indicating that the dynamic analysis method precisely identifies identical patterns across samples. While the method requires more time for analysis, the perfect matching Jaccard underscores its high accuracy in capturing similarities between samples, making it a robust tool for comprehensive dynamic analysis.

To provide a concise and comprehensive overview of our ransomware classification system, we present a detailed comparison of key features, time complexities, and analysis methods in the form of a diagram and table. The diagram illustrated in Figure 17 visually encapsulates the essential characteristics of our approach, highlighting the distinct time complexities and trade-offs associated with each analyzed feature static strings, static Import Address Table (IAT), and dynamic API calls.

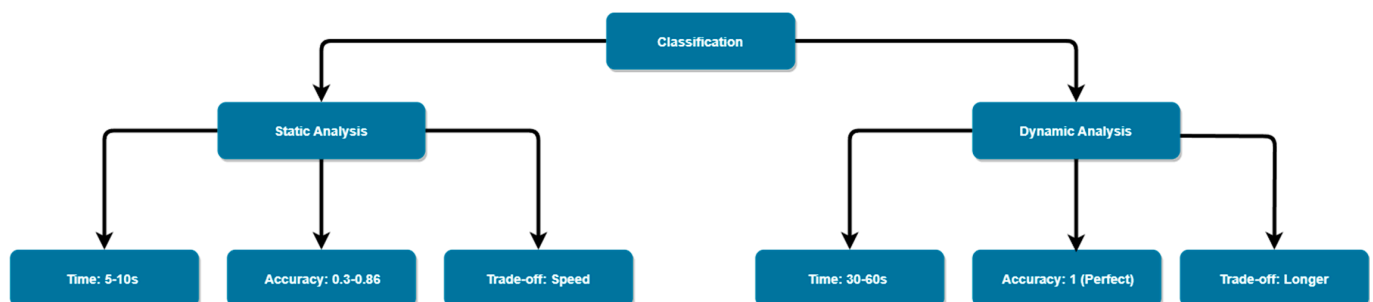


Figure 17. Ransomware classification system features.

A comparative analysis of ransomware classification features is described in Table 6, with a numerical comparison between static and dynamic analyzers.

Table 6. Comparative analysis of ransomware classification features.

Feature	Strings	Import Address Table	API Calls
Analysis Method	Static	Static	Dynamic
Time Complexity (per sample)	5 s	5–10 s	30–60 s
Similarity Accuracy	0.3	0.86	1 (Perfect Matching)
Analysis Efficiency	Fast processing	Balanced speed and comprehensive examination	Thorough analysis, longer processing time
Trade-off (Speed vs. Accuracy)	Emphasis on speed, lower accuracy	Balanced approach, high accuracy	Longer processing time for thorough dynamic analysis

5. Conclusions and Future Work

In this paper, we proposed a comprehensive approach for ransomware classification based on the comparison of similarity matrices derived from static analysis, dynamic analysis, and visualization. We extracted features from ransomware samples using multiple analysis techniques and generated similarity matrices based on these features. These matrices were then compared using various comparison algorithms to identify similarities and differences between the samples. The resulting similarity scores were used to classify the samples into different categories, such as families, variants, and versions. We evaluated our approach using a dataset of ransomware samples and demonstrated that it can accurately classify the samples with a high degree of accuracy. One advantage of our approach is the use of visualization, which allows us to classify and cluster large datasets of ransomware in a more intuitive and effective way. In addition, static analysis has the advantage of being fast and accurate, while dynamic analysis allows us to classify and cluster packed ransomware samples. Our study demonstrates the potential of using a comprehensive approach based on the comparison of multiple analysis techniques, including static analysis, dynamic analysis, and visualization, for the accurate and efficient classification of ransomware. It also highlights the importance of considering multiple analysis techniques in the development of effective ransomware classification methods, especially when dealing with large datasets and packed samples. In conclusion, our proposed comprehensive approach for ransomware classification is an effective and efficient method for accurately classifying and clustering ransomware samples. The use of visualization techniques is particularly useful for large datasets, while static analysis is fast and accurate, and dynamic analysis is useful for finding packed ransomware samples. By considering multiple analysis techniques, we can develop more effective methods for classifying and detecting ransomware, helping to protect individuals and organizations from this growing and evolving threat. In our future work, we plan to incorporate dynamic instrumentation into our ransomware classification approach to improve its accuracy and efficiency. Dynamic instrumentation involves monitoring and modifying the behavior of a program as it is being executed, which can provide valuable insights into the internal functions and communication patterns of ransomware. One approach we plan to explore is using dynamic instrumentation to track the internal functions of ransomware and how they interact with each other. By understanding the function calls and communication patterns of ransomware, we can potentially identify weaknesses and vulnerabilities that can be exploited to limit its damage or even decrypt affected files without paying the ransom. Additionally, we plan to investigate the use of machine learning techniques in combination with dynamic instrumentation to automate the process of identifying and classifying ransomware. By training a model on a large dataset of ransomware samples and their corresponding internal function calls and communication patterns, we can potentially develop a system that can accurately and efficiently classify new ransomware samples in real time.

Author Contributions: Conceptualization, B.Y., N.A. and M.A.A.; methodology, B.Y. and M.A.A.; software, B.Y.; validation, B.Y., N.A. and M.A.A.; formal analysis, B.Y., N.A. and M.A.A.; investigation, B.Y., M.S.E. and M.A.A.; resources, B.Y.; data curation, B.Y. and M.A.A.; writing—original draft preparation, B.Y., A.D.J., N.A. and M.A.A.; writing—review and editing, B.Y., M.S.E., A.D.J., N.A. and M.A.A.; visualization, B.Y., M.S.E., A.D.J., N.A. and M.A.A.; supervision, A.D.J., M.S.E. and M.A.A.; project administration, B.Y., M.S.E., A.D.J., N.A. and M.A.A.; funding acquisition, A.D.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the University College Dublin (UCD), School of Computer Science, Dublin, Ireland, grant number 13/RC/2077.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: This article does not contain any studies with human participants or animals performed by any of the authors.

Data Availability Statement: Data in this research paper will be shared upon request made to the corresponding author.

Conflicts of Interest: All authors declare that they have no conflict of interest for the presented work.

References

- Gopinath, M.; Sethuraman, S.C. A comprehensive survey on deep learning based malware detection techniques. *Comput. Sci. Rev.* **2023**, *47*, 100529.
- Brown, A.; Gupta, M.; Abdelsalam, M. Automated machine learning for deep learning based malware detection. *Comput. Secur.* **2024**, *137*, 103582.
- Kok, S.; Abdullah, A.; Jhanjhi, N.; Supramaniam, M. Ransomware, threat and detection techniques: A review. *Int. J. Comput. Sci. Netw. Secur.* **2019**, *19*, 136.
- Yadav, C.S.; Singh, J.; Yadav, A.; Pattanayak, H.S.; Kumar, R.; Khan, A.A.; Haq, M.A.; Alhussen, A.; Alharby, S. Malware analysis in iot & android systems with defensive mechanism. *Electronics* **2022**, *11*, 2354.
- Rey, V.; Sánchez, M.S.; Celdrán, A.H.; Bovet, G. Federated learning for malware detection in IoT devices. *Comput. Netw.* **2022**, *204*, 108693. [\[CrossRef\]](#)
- Johnson, S.; Gowtham, R.; Nair, A.R. Ensemble Model Ransomware Classification: A Static Analysis-based Approach. In *Inventive Computation and Information Technologies: Proceedings of ICICIT 2021*; Springer Nature: Singapore, 2022; pp. 153–167.
- Al-rimy, B.A.S.; Maarof, M.A.; Shaid, S.Z.M. Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions. *Comput. Secur.* **2018**, *74*, 144–166. [\[CrossRef\]](#)
- Akhtar, Z. Malware detection and analysis: Challenges and research opportunities. *arXiv* **2021**, arXiv:2101.08429.
- Tahir, R. A study on malware and malware detection techniques. *Int. J. Educ. Manag. Eng.* **2018**, *8*, 20. [\[CrossRef\]](#)
- 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. [\[CrossRef\]](#)
- Yamany, B.E.M.; Azer, M.A. SALAM Ransomware Behavior Analysis Challenges and Decryption. In *Proceedings of the 2021 Tenth International Conference on Intelligent Computing and Information Systems (ICICIS)*, Cairo, Egypt, 5–7 December 2021; pp. 273–277.
- Fernando, D.W.; Komninos, N.; Chen, T. A study on the evolution of ransomware detection using machine learning and deep learning techniques. *IoT* **2020**, *1*, 551–604. [\[CrossRef\]](#)
- Khan, F.; Ncube, C.; Ramasamy, L.K.; Kadry, S.; Nam, Y. A digital DNA sequencing engine for ransomware detection using machine learning. *IEEE Access* **2020**, *8*, 119710–119719. [\[CrossRef\]](#)
- Liu, K.; Xu, S.; Xu, G.; Zhang, M.; Sun, D.; Liu, H. A review of android malware detection approaches based on machine learning. *IEEE Access* **2020**, *8*, 124579–124607. [\[CrossRef\]](#)
- Bae, S.I.; Lee, G.B.; Im, E.G. Ransomware detection using machine learning algorithms. *Concurr. Comput. Pract. Exp.* **2020**, *32*, e5422. [\[CrossRef\]](#)
- Chakkaravarthy, S.S.; Sangeetha, D.; Cruz, M.V.; Vaidehi, V.; Raman, B. Design of intrusion detection honeypot using social leopard algorithm to detect IoT ransomware attacks. *IEEE Access* **2020**, *8*, 169944–169956. [\[CrossRef\]](#)
- El-Kosairy, A.; Azer, M.A. Intrusion and ransomware detection system. In *Proceedings of the 2018 1st International Conference on Computer Applications & Information Security (ICCAIS)*, Riyadh, Saudi Arabia, 4–6 April 2018; pp. 1–7.
- Vishwakarma, R.; Jain, A.K. A honeypot with machine learning based detection framework for defending IoT based botnet DDoS attacks. In *Proceedings of the 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, 23–25 April 2019; pp. 1019–1024.
- Keong Ng, C.; Rajasegarar, S.; Pan, L.; Jiang, F.; Zhang, L.Y. VoterChoice: A ransomware detection honeypot with multiple voting framework. *Concurr. Comput. Pract. Exp.* **2020**, *32*, e5726. [\[CrossRef\]](#)

20. Pont, J.; Arief, B.; Hernandez-Castro, J. Why current statistical approaches to ransomware detection fail. In Proceedings of the International Conference on Information Security, Bali, Indonesia, 16–18 December 2020; Springer International Publishing: Cham, Switzerland, 2020; pp. 199–216.
21. Yewale, A.; Singh, M. Malware detection based on opcode frequency. In Proceedings of the 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), Ramanathapuram, India, 25–27 May 2016; pp. 646–649.
22. Rezaei, S.; Afraz, A.; Rezaei, F.; Shamani, M.R. Malware detection using opcodes statistical features. In Proceedings of the 2016 8th International Symposium On Telecommunications (IST), Tehran, Iran, 27–28 September 2016; pp. 151–155.
23. Verma, V.; Muttou, S.K.; Singh, V.B. Multiclass malware classification via first-and second-order texture statistics. *Comput. Secur.* **2020**, *97*, 101895. [\[CrossRef\]](#)
24. Du, P.; Sun, Z.; Chen, H.; Cho, J.H.; Xu, S. Statistical estimation of malware detection metrics in the absence of ground truth. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2965–2980. [\[CrossRef\]](#)
25. Bijitha, C.V.; Sukumaran, R.; Nath, H.V. A survey on ransomware detection techniques. In *Secure Knowledge Management in Artificial Intelligence Era: 8th International Conference, SKM 2019, Goa, India, 21–22 December 2019*; Proceedings 8; Springer: Singapore, 2020; pp. 55–68.
26. Bello, A.; Maurushat, A. Synthesis of Evidence on Existing and Emerging Social Engineering Ransomware Attack Vectors. In *Cybersecurity Issues, Challenges, and Solutions in the Business World*; IGI Global: Hershey, PA, USA, 2023; pp. 234–254.
27. Cai, C.X.; Zhao, R. Salience theory and cryptocurrency returns. *J. Bank. Financ.* **2024**, *159*, 107052. [\[CrossRef\]](#)
28. Oz, H.; Aris, A.; Levi, A.; Uluagac, A.S. A survey on ransomware: Evolution, taxonomy, and defense solutions. *ACM Comput. Surv. (CSUR)* **2022**, *54*, 1–37. [\[CrossRef\]](#)
29. Alzahrani, S.; Xiao, Y.; Sun, W. An analysis of conti ransomware leaked source codes. *IEEE Access* **2022**, *10*, 100178–100193. [\[CrossRef\]](#)
30. Shu, R.; Xia, T.; Williams, L.; Menzies, T. Omni: Automated ensemble with unexpected models against adversarial evasion attack. *Empir. Softw. Eng.* **2022**, *27*, 26. [\[CrossRef\]](#)
31. Alagappan, A.; Venkatachary, S.K.; Andrews, L.J.B. Augmenting Zero Trust Network Architecture to enhance security in virtual power plants. *Energy Rep.* **2022**, *8*, 1309–1320. [\[CrossRef\]](#)
32. Whyte, C.; Mazanec, B. *Understanding Cyber-Warfare: Politics, Policy and Strategy*; Routledge: Oxford, UK, 2023.
33. Berrueta, E.; Morato, D.; Magaña, E.; Izal, M. A survey on detection techniques for cryptographic ransomware. *IEEE Access* **2019**, *7*, 144925–144944. [\[CrossRef\]](#)
34. Kara, I.; Aydos, M. The rise of ransomware: Forensic analysis for windows based ransomware attacks. *Expert Syst. Appl.* **2022**, *190*, 116198. [\[CrossRef\]](#)
35. Gómez-Hernández, J.A.; Sánchez-Fernández, R.; García-Teodoro, P. Inhibiting crypto-ransomware on windows platforms through a honeyfile-based approach with R-Locker. *IET Inf. Secur.* **2022**, *16*, 64–74. [\[CrossRef\]](#)
36. Almomani, I.; Alkhayer, A.; El-Shafai, W. A crypto-steganography approach for hiding ransomware within HEVC streams in android IoT devices. *Sensors* **2022**, *22*, 2281. [\[CrossRef\]](#)
37. Ahmed, M.; Afreen, N.; Ahmed, M.; Sameer, M.; Ahamed, J. An inception V3 approach for malware classification using machine learning and transfer learning. *Int. J. Intell. Netw.* **2023**, *4*, 11–18. [\[CrossRef\]](#)
38. Chaganti, R.; Ravi, V.; Pham, T.D. A multi-view feature fusion approach for effective malware classification using Deep Learning. *J. Inf. Secur. Appl.* **2023**, *72*, 103402. [\[CrossRef\]](#)
39. Eren, M.E.; Bhattarai, M.; Rasmussen, K.; Alexandrov, B.S.; Nicholas, C. MalwareDNA: Simultaneous Classification of Malware, Malware Families, and Novel Malware. In Proceedings of the 2023 IEEE International Conference on Intelligence and Security Informatics (ISI), Charlotte, NC, USA, 2–3 October 2023; pp. 1–3.
40. Marques, A.B.; Branco, V.; Costa, R.; Costa, N. Data Visualization in Hybrid Space—Constraints and Opportunities for Design. In Proceedings of the International Conference on Design and Digital Communication, Barcelos, Portugal, 3–5 October 2022; Springer Nature: Cham, Switzerland, 2022; pp. 3–15.
41. Rimón, S.I.; Haque, M.M. Malware Detection and Classification Using Hybrid Machine Learning Algorithm. In Proceedings of the International Conference on Intelligent Computing & Optimization, Hua Hin, Thailand, 27–28 October 2022; Springer International Publishing: Cham, Switzerland, 2022; pp. 419–428.
42. Mallik, A.; Khetarpal, A.; Kumar, S. ConRec: Malware classification using convolutional recurrence. *J. Comput. Virol. Hacking Tech.* **2022**, *18*, 297–313. [\[CrossRef\]](#)
43. Abbasi, M.S.; Al-Sahaf, H.; Mansoori, M.; Welch, I. Behavior-based ransomware classification: A particle swarm optimization wrapper-based approach for feature selection. *Appl. Soft Comput.* **2022**, *121*, 108744. [\[CrossRef\]](#)
44. Kim, J.; Lee, S. Malware Visualization and Similarity via Tracking Binary Execution Path. *Teh. Vjesn.* **2022**, *29*, 221–230.
45. Saxe, J.; Sanders, H. *Malware Data Science: Attack Detection and Attribution*; No Starch Press: San Francisco, CA, USA, 2018.
46. Kong, K.; Zhang, Z.; Guo, C.; Han, J.; Long, G. PMMSA: Security analysis system for android wearable applications based on permission matching and malware similarity analysis. *Future Gener. Comput. Syst.* **2022**, *137*, 349–362. [\[CrossRef\]](#)
47. Mudgil, P.; Gupta, P.; Mathur, I.; Joshi, N. A novel similarity measure for context-based search engine. In *Proceedings of the International Conference on Innovative Computing and Communications: Proceedings of ICICC 2022*; Springer Nature: Singapore, 2022; Volume 2, pp. 791–808.

48. Abbas, A.R.; Mahdi, B.S.; Fadhil, O.Y. Breast and lung anticancer peptides classification using N-Grams and ensemble learning techniques. *Big Data Cogn. Comput.* **2022**, *6*, 40. [[CrossRef](#)]
49. Cucchiarelli, A.; Morbidoni, C.; Spalazzi, L.; Baldi, M. Algorithmically generated malicious domain names detection based on n-grams features. *Expert Syst. Appl.* **2021**, *170*, 114551. [[CrossRef](#)]
50. Di Mauro, M.; Galatro, G.; Liotta, A. Experimental review of neural-based approaches for network intrusion management. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 2480–2495. [[CrossRef](#)]
51. Dong, S.; Xia, Y.; Peng, T. Network abnormal traffic detection model based on semi-supervised deep reinforcement learning. *IEEE Trans. Netw. Serv. Manag.* **2021**, *18*, 4197–4212. [[CrossRef](#)]
52. Pelletier, C.; Webb, G.I.; Petitjean, F. Deep learning for the classification of Sentinel-2 image time series. In Proceedings of the IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium, Yokohama, Japan, 28 July–2 August 2019; pp. 461–464.

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.