IJARSCT

International Journal of Advanced Research in Science, Communication and Technology

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

ISSN: 2581-9429

Volume 5, Issue 2, December 2025

Impact Factor: 7.67

XRGuard: A Model-Agnostic Approach to Ransomware Detection Using Dynamic Analysis and Explainable AI

Ruday Kailas Gadekar, Om Gajanan Wakchaure, Akash Ravindra Dongare
Prof. J K Shimpi, Prof C V Patekar
Adsul Technical Campus, Chas, Ahilyanagar, Maharashtra, India

Abstract: *Ransomware remains a persistent and evolving cybersecurity threat, demanding advanced and adaptable detection strategies. Traditional methods often fall short as signature-based systems are easily circumvented by emerging ransomware variants, while techniques like obfuscation and polymorphism add complexity to the detection process. Although machine learning and deep learning techniques present viable solutions, the opacity of complex black-box models can hinder their application in critical security environments. This paper introduces XRGuard, a novel ransomware detection framework that utilizes machine learning techniques to analyze Event Tracing for Windows (ETW) logs, identifying critical file I/O patterns indicative of ransomware attacks. By incorporating XAI techniques such as SHapley Additive exPlanations (SHAP) and Local Interpretable Model-Agnostic Explanations (LIME), XRGuard bridges the trust gap associated with complex machine learning models by providing transparent and interpretable explanations for its decisions. Experimental results demonstrate that XRGuard achieves a 99.69% accuracy rate with an exceptionally low false positive rate of 0.5%. By enhancing detection accuracy and offering clear explanations of its operations, XRGuard not only improves security but also fosters trust and a deeper understanding of ransomware behaviors.*

Keywords: Event tracing for windows (ETW), FileIO, machine learning, ransomware, XAI, explainable AI, SHAP, LIME

I. INTRODUCTION

Ransomware has evolved into a pervasive and sophisticated cyber threat, inflicting substantial damage on both individuals and organizations. Cybercriminals employ this tactic to extort money, leveraging fear and urgency to maximize financial gain. The impact of ransomware is severe, resulting in significant economic losses, operational disruptions, and reputational damage. Recent reports from BleepingComputer [1] highlight the alarming increase in ransomware attacks, both in frequency and complexity. According to their report, ransomware victims paid $460 million in the first half of 2024 which is approximately 2% higher than 2023's recordbreaking trajectory from the same period despite significant law enforcement operations that disrupted large ransomwareas-a-service operations, such as LockBit.

Mitigating ransomware requires a multi-layered defense strategy that combines robust detection and proactive prevention measures. While traditional signature-based approaches are effective against known threats, they fall short when dealing with zero-day attacks and advanced variants that utilize polymorphism and obfuscation. To address these challenges, researchers have turned to machine learning (ML) and deep learning techniques, which are more efficient in detecting zero-day attacks. However, these algorithms often function as black boxes, offering little insight into their internal workings and raising concerns about trust in the overall decision-making process.

Recent advancements in Explainable AI (XAI) techniques have significantly enhanced the ability to understand and interpret AI system decisions. XAI algorithms now provide clearer insights into model operations, bridging the gap between human understanding and artificial intelligence [2], [3] [4], [5] [6], [7] [8]. In light of these developments, this

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 2, December 2025
Impact Factor: 7.67

research proposes XRGuard, an innovative hybrid defense mechanism that leverages the Windows Event Tracing for Windows (ETW) framework to capture OS kernel events.
To elucidate the internal workings of the ML models, XRGuard employed XAI techniques such as SHAP and LIME, which offer insights and rationales behind model decisions, thereby improving trust in the overall detection system.
To our knowledge, only two studies in literature used Windows inbuilt ETW logs for detecting Ransomware including [9] and [10]. Ahmed et al. [9] used an NLP-based deep learning model to fingerprint the contextual behavior of applications while Rana et al. [10] focused on agent development for capturing system logs. XRGuard employs ETW (Event Tracing for Windows) File I/O patterns to train machine learning models, enabling precise detection of ransomware behavior. Additionally, it leverages Explainable AI (XAI) models to provide clear, interpretable insights into the internal decision-making processes. To the best of our knowledge, this unique combination of ETW-based feature extraction and XAI-driven interpretability has not been explored in prior research, positioning XRGuard as a novel approach in the field of ransomware detection. Here the key contributions of this research are highlighted:
1) XRGuard, a hybrid defense mechanism is proposed that leverages the Windows inbuilt ETW logs and ML algorithms for identification of malicious event patterns.
2) The prototype has been developed using 20 distinct ransomware families, demonstrating its resilience by effectively classifying previously unseen malware samples. Equipped with XAI techniques such as SHAP and LIME, the system provides insights into the predictions and rationales behind model decisions, thereby enhancing trust in the overall detection system.
3) By leveraging advanced machine learning techniques and Explainable AI (XAI) models, XRGuard significantly enhances both the accuracy and interpretability of ransomware detection. This innovative approach enables the system to identify potential threats within 5 to 10 seconds of their launch, while maintaining an ultra-low False Positive Rate (FPR) of approximately 0.5%. This combination of speed, precision, and transparency makes XRGuard a highly effective and trustworthy solution for real-time ransomware detection.
While ransomware attacks target Windows, Linux, Mobile, and Internet of Things (IoT) ecosystems, this research primarily focuses on cryptographic Windows ransomware.
To evaluate the efficiency of the proposed approach XRGuard trained with a Random Forest classifier was compared with five commonly used machine learning models.
Experimental results indicate that the Random Forest model is the most effective for ransomware detection, achieving a detection accuracy of 99.69% with negligible false positives.
The following sections of this paper are structured as follows: The literature review and related work are discussed in Section II. Section III introduces the architecture of XRGuard, explaining the essential characteristics and parameters used for detecting various Ransomware families. Following this, Section IV provides a detailed overview of the System Design and Implementation. Section V outlines the experimental setup used in this study, while Section VI analyzes the results obtained. Finally, in Section VII, the findings are summarized and future research directions are discussed.

II. RELATED WORK

A detailed comparison of different research efforts performed between 2020-2024 is provided in this Section. Malware detection and classification commonly rely on static, dynamic, or hybrid analysis approaches. Signature matching remains one of the most commonly used static analysis techniques for ransomware detection. This technique utilizes predefined characteristics of known malware samples for detection. However, this technique faces significant limitations, particularly in detecting zero- day attacks where the malware's signature is unknown. To evade static detection methods, modern ransomware variants often incorporate novel encryption techniques or alter their signatures. Furthermore, sophisticated malware exhibits adaptive behaviors, using obfuscation techniques to conceal their malicious actions. This adds a layer of complexity to the analysis process. Advanced ransomware strains also employ evasion strategies like process and functional splitting, as well as mimicry, to bypass detection by leading commercial behavioral detection systems.

This section offers a summary of various techniques and methodologies employed by researchers for ransomware detection. Various log sources used by researchers include registry events, Windows API Calls, Disk IO Patterns, Sysmon Logs, Network telemetry, and Process Logs. Since most crypto Ransomware encrypts user files to make them accessible, File I/O monitoring is the most widely used technique for ransomware detection. Begovic et al. [11] focused on monitoring file system activities, particularly those targeting user files and the Master Boot Record (MBR), to neutralize potential threats. Malik et al. [12] discusses the critical role of features in identifying ransomware and underscores the significance of features such as Process ID and Time in the ransomware detection process. Kharraz et al. [13] proposed UNVEIL and its successor Redemption [14] that detect suspicious activity by computing a score using a heuristic function over various behavioral features. Arabo et al. [15] investigated the relationship between process behavior and ransomware to determine if this method can help evade malicious software. Ayub et al. [16] classified IRP logs with Artificial Neural Network (ANN) for ransomware

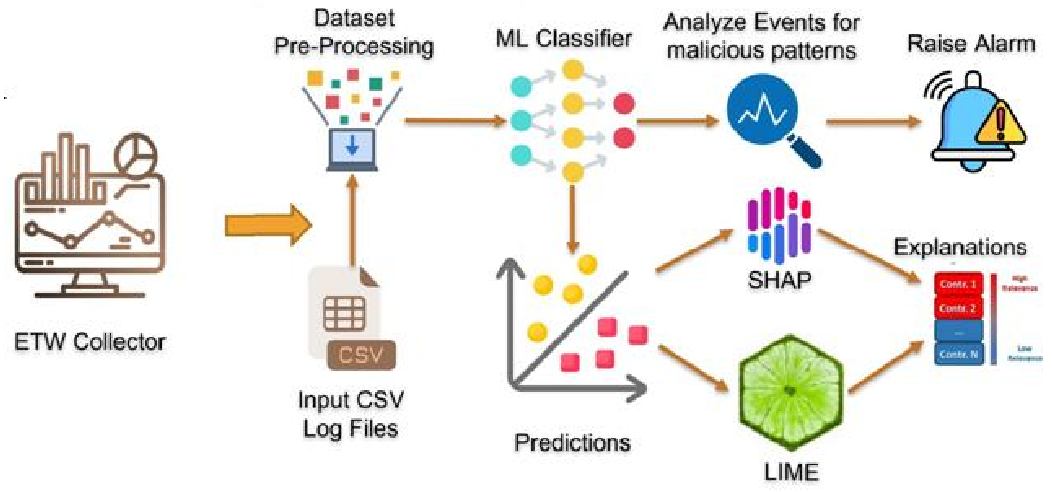

FIGURE 1. Complete flow diagram of XRGuard.

detection. Continella et al. [17] proposed ShieldFS, an addon Windows driver that makes the Windows native file system immune to ransomware attacks. Nalinipriya et al.

[18] proposed DefendR, a framework designed to prevent users from editing or overwriting personal files by using a mini-filter driver to protect sensitive data. Morris et al. [19] proposed FREEDOM a system designed to protect user files from ransomware by using Alternative Data Streams to mislead ransomware and prevent encryption. Bailluet et al. [20] used File System Filter Drivers to intercept I/O request packets and deny operations on user files.

File monitoring techniques alone are often inadequate evidence of malicious activity because similar modifications can be made by legitimate programs that handle consumer data, such as compression and lawful encryption tools. Therefore, the decoy approach cannot conclusively determine whether the alterations were executed by ransomware, leading to numerous false alarms Alqahtani and Sheldon [21].

File entropy analysis is another widely utilized technique in various research studies. It measures the randomness or unpredictability of data within a file, making it a valuable tool for detecting encryption. By analyzing file entropy, researchers can identify files that have been encrypted, as encrypted files typically exhibit higher entropy levels due to their random data patterns. Joshi et al. [22] used a mini- filter driver with Shannon's entropy and fuzzy hash for detecting ransomware using a signature-less detection method. Pont et al. [23] compared statistical tests for randomness to anti-ransomware tools and found that the thresholds used by the tools often lead to false positives, incorrectly labeling un-encrypted data as encrypted.

Ransomware often communicates with command- andcontrol (C2) servers to exchange encryption keys or transfer user data for extortion purposes. Due to this characteristic, monitoring network traffic can be an effective approach for early ransomware detection and warning. Network traffic analysis involves continuously observing and analyzing network traffic patterns to identify any anomalies or suspicious activities that could signal an ongoing ransomware attack. By detecting these unusual traffic patterns, it becomes possible to spot ransomware communication attempts, potentially allowing for quicker intervention before significant damage is done. Morato Oses et al. [24] created a model based on shared file access patterns using machine learning techniques, deep learning models, and LSTM recurrent neural networks for ransomware detection. Berrueta et al. [25] proposed a network traffic analysis technique for ransomware detection that works both on clear text and encrypted file- sharing protocols. Xia et al. [26] proposed a local detection algorithm that considers entropy, read/write frequency, and read/write patterns. Hirano et al. [27] presented an open-source dataset of low-level storage access patterns obtained by using an open-source lightweight hypervisor ‘‘Bitvisor’’ specialized in monitoring input and output devices. Tang et al. [28] presented RansomSpector, which operates on the KVM hypervisor and monitors file and network activities to detect crypto-ransomware. However, the biggest limitation in network-based detection approaches is that attackers can easily evade detection by encrypting their network traffic or utilizing covert communication channels Alraizza and Algarni [29].

Encrypting user drives is a resource-intensive task that can lead to significant CPU and memory overloads or, at the very least, cause sudden deviations from the system’s typical workload patterns. These abrupt changes in system behavior can serve as indicators of potential ransomware activity, making hardware load monitoring an effective method for anomaly detection. Modern CPUs from all major vendors are equipped with multiple Hardware Performance Counters (HPCs) designed to monitor and log the system’s status for diagnostic and analytical purposes. HPCs are specialized registers within the CPU that are continuously updated for hardware-related events, such as ‘‘cache misses’’ and ‘‘branch misses’’. The data collected from these events offers valuable insights into the system’s overall behavior, including the operations of applications, operating systems, and potentially malicious processes. By analyzing HPC data, it’s possible to identify unusual patterns or spikes in resource usage that may indicate the presence of ransomware or other malicious actors.

HPC (Hardware performance counters), Processor and disk usage data is used by Thummapudi et al. [30] for detection of ransomware attacks. ‘‘DeepWare,’’ a CNN- based ransomware detection approach is proposed by Olani et al. [31], which converts data from hardware performance counters into images and then uses these images for identifying ransomware by classifying these images. Aurangzeb et al. [32] utilized the RandomForest classifier with HPC data for classifying and detecting malicious applications as ransomware or non-ransomware. Pundir et al. [33] proposed ‘‘RanStop’’ that uses HPCs to observe micro-architectural event sets and detects known and unknown crypto-ransomware variants.

Explainable AI (XAI) provides transparency and interpretability in the decision-making processes of machine learning models, which is especially critical in malware detection systems. By integrating XAI, these systems can achieve a balance between high accuracy and actionable insights, making them more effective and reliable in real-world cybersecurity operations. AlSobeh [34], Capuano et al. [35], Noori and Albahri [36] and Mohammed and Aljanabi [37] highlighted the integration of XAI and interpretability in machine learning models, particularly in security applications where understanding model decisions is critical. ETW is a powerful logging and tracing framework integrated into the Windows operating system. Designed for performance monitoring and debugging purposes, ETW enables the collection of real- time event information from various components of the operating system and installed applications. The framework utilizes a publish-subscribe model, where providers emit events, and consumers subscribe to specific events of interest [38]. Key features of the ETW framework include low overhead, allowing it to run efficiently even in production environments without significant impact on system performance. Events generated by ETW can capture a wide range of information, including system calls, application-specific events, and hardwarerelated activities. For logging kernel-level I/O events from the Windows operating system ETW offers the ’NT Kernel Logger,’ Blake [39]. Ahmed et al. [9] proposed a technique, called ‘‘Peeler’’ (Profiling kernel Level Events to detect Ransomware). Peeler uses an NLP-based deep learning model to fingerprint the behavior of system processes. The research by Rana et al. [10] focuses on developing an ETW monitoring agent for malware detection using behavioral data.

III. PROPOSED TECHNIQUE: XRGUARD

In this research XRGuard, a dynamic analysis technique is proposed, which leverages Windows ETW logs to analyze patterns indicative of potential ransomware activity, enabling proactive detection during the initial phases of an attack. ETW logs were chosen as the primary data source due to their fine-grained detail, real-time capabilities, and deep system-level visibility, all of which are essential for identifying file I/O patterns indicative of ransomware activity. Specifically, ETW logs provide comprehensive records of file system operations—such as file creation, modification, and deletion—enabling the precise detection of encryption behaviors that are characteristic of ransomware attacks. Unlike traditional system or application logs, ETW operates at the kernel level, offering unparalleled insights into low-level system activities with minimal performance overhead. While alternative data sources like Sysmon logs and system application logs are valuable for monitoring process and network activities, they fall short in providing the focused, granular detail on file I/O operations necessary for effective ransomware detection. XRGuard is designed to detect ransomware in its early stages, thereby reducing the attack surface by providing timely warnings to users.

This research is an extension of our previous work [40]. Inthisstudy,weutilizedourETWcollectormoduletocapture Windows ETW logs, which were subsequently preprocessed, filtered, and classified using a machine learning (ML) classifier. After classification, we employed Explainable AI (XAI) techniques, specifically SHAP [41] and LIME [42], to interpret the internal decision-making processes of the black-box ML models and to identify the critical features and rationale behind the model's decisions. The complete workflow of XRGuard is illustrated in Figure 1, providing a comprehensive overview of the system's architecture and operational flow.

A. MODEL-AGNOSTIC EXPLANATIONS OF ML MODELS

Model-agnostic explainable artificial intelligence (XAI) is a technique that interprets any machine learning model, regardless of its complexity, without needing to examine its internal structure. This approach makes complex black-box models more transparent and understandable, helping to build trust and accountability in AI-driven decisions. XAI aims to enhance the interpretability of AI systems, making them more accessible and reliable in critical areas like healthcare, finance, and autonomous systems.

This research employs two widely recognized Explainable AI techniques, SHAP and LIME, to provide insights into the internal decision-making processes of our machine learning models. Figure 2 illustrates the block diagram of the XAI techniques utilized in this study. Specifically, SHAP is used to derive global explanations, offering a comprehensive understanding of the model's behavior across the entire dataset, while LIME is applied to generate local explanations, highlighting feature importance and model decisions for individual instances. Together, these techniques enhance the interpretability and transparency of the model, enabling a deeper understanding of its predictions and the underlying factors driving its outcomes.

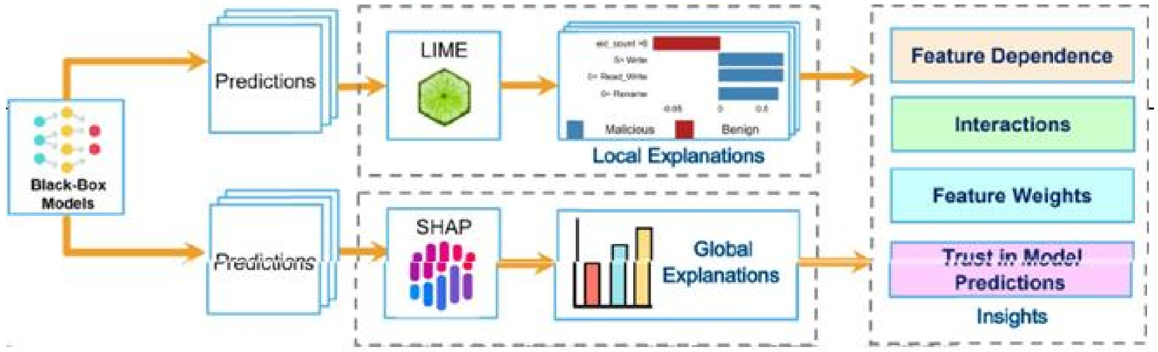

FIGURE 2. Model agnostic techniques.

1) SHAPLEY ADDITIVE EXPLANATIONS (SHAP)

SHAP is a powerful and widely used method for interpreting the predictions of complex machine learning models. Based on the concept of Shapley values [43] from cooperative game theory, SHAP assigns a value to each feature in a prediction, reflecting its contribution to the model's output. It does
this by considering all possible combinations of features and calculating the average marginal contribution of each feature across these combinations. This results in a set of additive feature attributions that sum up to the model's

prediction, providing a clear and consistent explanation of how each feature influencesthe outcome.One of thestrengths of SHAP is its solid theoretical foundation, which ensures that the feature contributions are fair and balanced, adhering to properties like consistency and accuracy. SHAP can be used with any machine learning model, making it a versatile tool for gaining insights into model behavior, improving transparency, and facilitating trust in AI systems. The Shapley value φi for feature i is
where L(f ,g,πx) is a loss function measuring how well the simpler model g approximates the complex model f , weighted by the proximity of perturbed samples to the original instance x, and (g) is a regularization term that ensures simplicity.

IV. EXPERIMENTAL SETUP

To validate the efficiency of XRGuard following experiments were designed.

A. DATASET COLLECTION

An experimental test bed was constructed to collect data for validating the efficiency of XRGuard. Unlike fields such as malware detection and image processing, ransomware detection and classification lack standardized datasets, necessitating the collection of a custom dataset. Cen et al. [44]. It was found that websites like virustotal [45] are the major repositories having ransomware samples from different families, including Cerber, TeslaCrypt, CryptoWall, Petya, and WannaCry. However, other repositories such as malwarebazaar [46], Monaco [47], and theZoo [48] also contain a substantial collection of live malware strains. Ransomware samples were collected from 20 different ransomware families from the above-mentioned repositories. To capture ETW telemetry, the ETW collector ransomware samples were executed in the sandbox and
ETW Collector was used to collect system logs during the execution. All live ransomware executions were conducted in a secure, isolated sandbox environment to ensure safety where N is the set of all features, S is a subset of N, and v(S) represents the model’s prediction using features in subset S. In simpler terms, the Shapley value is the weighted average of the marginal contributions of a feature i across all possible combinations of other features, reflecting its
overall importance to the model’s predictions.

2) LOCAL INTERPRETABLE MODEL-AGNOSTIC EXPLANATIONS (LIME)

LIMEexplainsthepredictionsofanymachinelearningmodel by approximating it locally with a simpler, interpretable model. For a given instance, LIME generates perturbed samples around it and fits an interpretable model to these samples. The resulting model helps explain the complex model’s predictions in the neighborhood of the instance. The objective function for LIME is:

$$\xi\,(g) = \mathrm{argmin}_{g \in G}\, L(f ,g, \pi\, x) + (g) \quad (2)$$

and prevent any risk of infection. The sandbox consisted of a Dell PowerEdge Server running ESXi 8.0. Two Windows guest virtual machines, one with Windows Server 2019 and the other with Windows 10, were used to run ransomware samples for data collection. A snapshot of the fresh operating system was taken on each machine and restored at the end of each data collection cycle. the complete dataset collection cycle is shown in Figure: 3
To simulate real user actions during the data collection process, daily activities such as downloading files, browsing the internet, opening Office documents, compressing large amounts of data, and navigating through local storage folders were automated using a task scheduler. A wide range of benign software packages were installed on the virtual machines, including Mozilla Firefox, Microsoft Visual Stu-

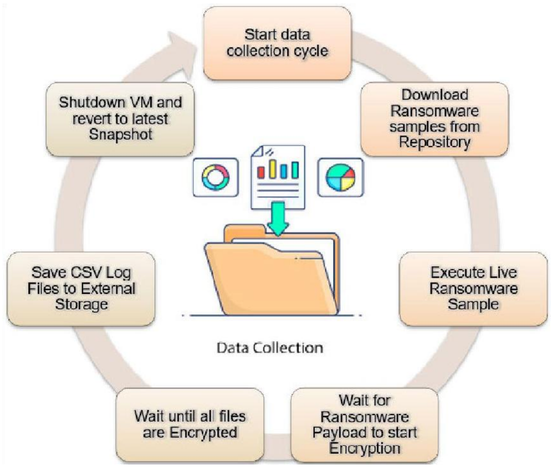
Start data collection cycle
Download Ransomware samples from Repository
Execute Live Ransomware Sample
Wait for Ransomware Payload to start Encryption
Wait until all files are Encrypted
Save CSV Log Files to External Storage
Shutdown VM and revert to latest Snapshot
Data Collection

FIGURE 3. Dataset collection cycle.

dio, Adobe Acrobat Reader, Google Chrome, 7zip, WinRAR, Microsoft Office Professional 2019, Media Player, VSCode, Microsoft Edge, and Notepad++. While the malware samples were running, these benign applications were also executed to generate benign application logs for the dataset.

To prevent interference from built-in Windows security applications, antivirus software, and Windows Defender were disabled on the virtual machines before downloading ransomware samples from the repositories. It was observed that many binaries uploaded to these repositories were not ransomware. To verify that a binary was indeed ransomware, system activities were closely monitored after executing the binary to confirm whether a ransomware attack occurred. However, pinpointing the exact moment when ransomware begins encrypting data can be challenging, as ransomware often employs various tactics to evade detection. Asubstantialdataset,comprisingapproximately15million I/O requests, was collected during the data collection process. After pre-processing and filtering, the dataset was reduced to around 1 million rows of data. The distribution of various ransomware and benign samples in the dataset is presented in Table 1 and Table 2.

TABLE 1. Dataset distribution.

Data Labe	Number of Log Samples
Ransomware Logs	336,135
Benign samples	557,123

B. PRE-PROCESSING AND FEATURE SELECTION The collected

dataset was stored in CSV files containing parsed events from ETW logs. Each event is represented in the following structure: event = TimeStamp, TimeStampRelativeMSec, ProcessID, ParentProcessID, ProcessName, ThreadID, ParentThreadID, EventName, EventID, FilePath, FileName, FileType, IrpPtr, FileObject, FileKey, ExtraInfo, InfoClass This structure includes essential details such as timestamps, process and thread identifiers, event attributes, and file-related metadata. To correlate event sequences on

TABLE 2. Some of major Ransomware strains and their distribution in collected dataSet.

Ransomware	Number of Log Samples
dharma	12,935
wanna cry	21,700
RedBoot	13,947
uccvenum	15,193
Phobos	12,484
hive	21,515
CoronaVirus1	16,947
snatch	22,187
cerber	29,371
TeslaCrypt	33,500
Thanos	20,532

the same file with the same process ID, events sharing the same FileKey and PID are concatenated into a character string, which is then encoded into alphabetic representations as shown in Table 3 and 4.

TABLE 3. Alphabetic encoding of FileIO events.

Sr.	File IO Event	code	KernelProvide
1.	Rename	a	FileIOInit
2.	Delete	b	FileIOInit
3.	Read	c	FileIOInit
4.	Write	d	FileIOInit
5.	SetInfo	e	FileIOInit
6.	Create	f	FileIOInit
7.	Close	g	FileIOInit
8.	FileCreate	h	DiskFileIO
9.	FileDelete	i	DiskFileIO
10.	FSControl	j	FileIOInit
11.	DirEnum	k	FileIOInit
12.	Cleanup	l	FileIOInit
13.	DirNotify	m	FileIOInit
14.	QueryInfo	n	FileIOInit

TABLE 4. Alphabetic encoded event sequences after pre-processing and log cleaning.

FileName	EventID	Process	FileType	L
0067docx	cdg	System	docx	
0067docx	cdg	System	docx	
0067docx	fnclgdb	wcrt	docx	
0068docx	fdnlga	wcrt	docx	
0073docx	fnlgcb	drpbx	docx	
0075docx	fnclgdb	wcrt	docx	
0079docx	lgncda	cerber	docx	
0079docx	fdnlga	wcrt	docx	
0080docx	fdnlga	wcrt	docx	

The collected events undergo a preprocessing stage to clean the data and extract meaningful sequences. During this step, the character streams encoded in the previous phase are further filtered and normalized. Specifically, events with character sequence lengths of less than three are removed, as it has been observed that encryption operations cannot occur with fewer than three event types. Subsequently, events lacking create, read, write, delete, or rename operations are also removed, as these are essential for any encryption operation. Figure 4 shows the frequency of common event patterns found in benign and malicious FileIO logs.

Furthermore, certain features were excluded from the training dataset due to their lack of contextual information. For instance, PID (Process Identifiers) are unique runtime

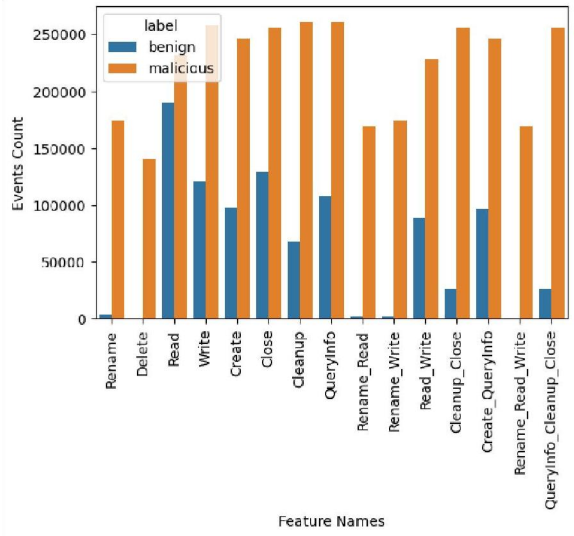

FIGURE 4. Comparison of benign as well as malicious event patterns found in the dataset.

identifiers assigned to processes, but they are inconsistent across systems or sessions and do not provide meaningful information about the nature of a process (e.g., whether it is malicious or benign). Similarly, Timestamp values alone do not convey behavioral insights unless used to derive other features, such as event frequency or time-based patterns. Features like Filenames, FileObject, and FileKey were also excluded, as they lack contextual information that could aid in distinguishing between benign and malicious activities. To prevent overfitting, only FileTypes were retained instead of using the entire FileName, ensuring the model focuses on generalizable patterns rather than specific file instances. Figure 5 shows the event correlations of finally selected features.

C. ML-BASED ANOMALY DETECTOR

After pre-processing, the events are fed into the ML models for training. Five common ML models were tested: Decision Trees, Random Forests, Support Vector Machines (SVM), Gradient Boosting, and Adaboost. These models were chosen due to their diverse strengths in handling different types of data and their proven effectiveness in anomaly detection. Decision Trees offer interpretability and are known for their ability to handle categorical data but are prone to overfitting. Random Forests address overfitting through ensemble learning, though they can be computationally expensive. SVM performs well in high- dimensional spaces but struggles with large datasets and requires careful hyperparameter tuning. Gradient Boosting is robust against overfitting and achieves high accuracy but is complex to implement and is also very resource-intensive. Adaboost effectively boosts weak classifiers but is sensitive to noisy data and outliers. These models were chosen not only for their individual strengths but also for their collective ability to address a wide range of data challenges in anomaly detection

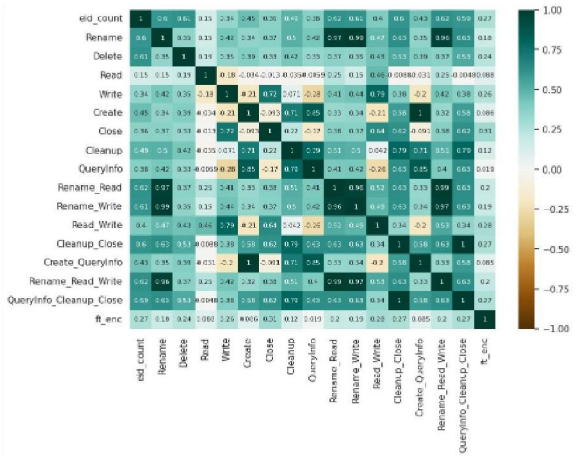

FIGURE 5. Correlation matrix of finally selected events/features.

D. EVALUATION MATRICS

To validate the accuracy and reliability of XRGuard, its results are compared with five commonly used machine learning models, and key performance metrics such as Precision, Recall, and the F1 Score were computed.
The mathematical descriptions of these metrics are provided in Equations (3)-(5).
Precision is calculated as the ratio of true positives to the sum of true positives and false positives.

TP Precision = TP + FP (3)

True positive rate: TPR, also known as Recall is the ratio of true positives to the sum of true positives and false negatives

TP Recall = TP + FN (4)

F1 score is a metric that combines precision and recall into a single value, balancing these two measures. The F1 score ranges from 0 to 1, with 1 indicating perfect precision and recall, and 0 indicating the worst performance

F-Score = 2 ×Prec isionPrecision+×RecallRecall (5)

V. MODEL EXPLANATIONS

A. LOCAL EXPLANATIONS VIA LIME

LIME constructs an interpretable, simplified model that replicates the behavior of the black-box model within the neighborhood of the specific data point. This method highlights the features that most significantly influenced a particular prediction, making it easier to trust, validate, and debug the model's decisions. A significant advantage of LIME is its model-agnostic nature, allowing it to be applied to any machine learning model, regardless of complexity. LIME offers per-prediction explanations rather than providing a global interpretation of the model. Figure 6 shows LIME explanations of 06 randomly selected model predictions. The samples were correctly classified as benign/Malicious by the ML model.

In Figure 6 (A, C, and E), it is observed that the operations eid_count, Read_Write, Cleanup_Close, Create, and Cleanup become prominent, leading to the samples being labeled as malicious. It is observed that certain ransomware strains often read and write the same file multiple times, resulting in high event counts for the same file, accompanied by Create operations for generating encrypted files, usually within the same directory. This combination is scarce in benign file operations, where eid_count is also relatively low. The absence of the Rename_Read operation also strongly indicates benign file operations. However, some ransomware strains create new encrypted files with random names and delete the original files instead of renaming them. In such cases, the Rename_Read operation is not present, as shown in Figure 6 (C). Consequently, the coexistence of Read_Write and Create events causes the ML model to classify these samples as malicious.

In contrast, Figure 6 (B, D, and F) shows Read_Write and Cleanup operations without Create events. Additionally, these samples exhibit very low eid_count, leading the ML model to classify them as benign.

B. EXPLANATIONS VIA SHAP

SHAP is a widely adopted method for interpretingpredictions from complex machine learning models, offering strong theoretical guarantees for fair and balanced feature contributions. As a model-agnostic tool, SHAP enhances transparency and builds trust in AI systems. A SHAP waterfall plot visually represents each feature's contribution to a model's prediction, with bars indicating these effects. Upward bars increase the prediction, while downward bars decrease it, and the length of each bar reflects the contribution's magnitude. A positive SHAP value means the feature increased the model's prediction, while a negative value indicates it decreased the prediction. Figure 7 displays the SHAP waterfall plots for six randomly selected model predictions. These samples were accurately classified as either benign or malicious by the ML model.

Figure 7 (A), shows that the operations delete, Write, and eid_count are on top and have large magnitudes hence having major contributions in marking the sample as malicious. eid_count is also present in Figure 7 (B, D, and F) but its values are relatively low i.e. below 06. so it is observed that if the value of eid_count is greater then our model considers it as a positive indicator for a malicious sample however if they are lower then as in the case of Figure 7 (B) it is considered as a strong indicator of the benign sample. In Figure 7 (F) Write has high magnitude but there is no delete command, hence it is considered a strong indicator for a benign sample which contradicts it as a strong indicator of a malicious sample in the presence of delete as in Figure 7 (A, C, and E). Figure 8 shows a summary of global explanations by SHAP. The SHAP summary plot illustrates the impact of various features on the model's output. Each dot represents a feature's SHAP value for a particular instance, with features sorted by their average impact on the model. The x-axis represents the SHAP value, indicating whether the feature pushes the prediction higher or lower. Features on the left have a negative impact, reducing the prediction, while those on the right increase it. The summary plot shows that Rename_Read_Write and eid_count have the most significant influence, where higher values increase the prediction. Conversely, Write and Rename_Write show that low values decrease the prediction. The color gradient reflects feature values, revealing how each contributes differently to the model's decisions.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 2, December 2025
Impact Factor: 7.67

VI. RESULTS AND DISCUSSIONS

The evaluation results, as presented in Table 5, indicate that XRGuard, utilizing the Random Forest classifier, achieved outstanding performance in distinguishing between ransomware and benign event logs, with a detection accuracy of 99.69%

TABLE 5. Results achieved for binary classifications.

Model	ACC %	Precision	Recall	F
Random Forest	99.69	0.987	0.986	0
Decision Tree	97.02	0.709	0.803	0
SVM	96.23	0.799	0.855	0
XGBoost	99.01	0.961	0.989	0
AdaBoost	98.12	0.916	0.823	0

Table 6 shows that for multiclass classification involving 20 selected ransomware strains, XRGuard achieved strong performance, achieving an average accuracy of 95.52%.

TABLE 6. Results achieved for multiclass classification using random forest classifier.

Model	Precision	Recall	F1 Score
Cerber	1.000	0.986	0.993
Clop	0.709	0.403	0.514
Dharma	0.799	0.755	0.777
Gamma	0.961	0.989	0.975
GoldenEye	0.916	0.623	0.741
Grandcrab	0.780	0.372	0.504
Hive	0.843	0.885	0.863
InfinityCrypt	0.684	0.506	0.582
Jigsaw	0.740	0.869	0.799
Katyusha	0.628	0.820	0.711
Lockbit	0.761	0.520	0.617
Phobos	0.796	0.748	0.771
Quantum_Locker	0.989	0.953	0.971
RedBoot	0.879	0.817	0.847
Satana	0.679	0.988	0.805
Snatch	0.865	0.944	0.903
TeslaCrypt	0.933	0.993	0.962

The analysis of false positives reveals that certain benign applications, such as WinZip, 7-Zip, and Windows processes like BackgroundTaskHost, svchost, and vmtoolssd, can exhibit behaviors that closely resemble those of ransomware, leading to potential misidentifications. Furthermore, it has been observed that some ransomware strains, such as Dharma, Jigsaw, and Satana, are frequently misclassified

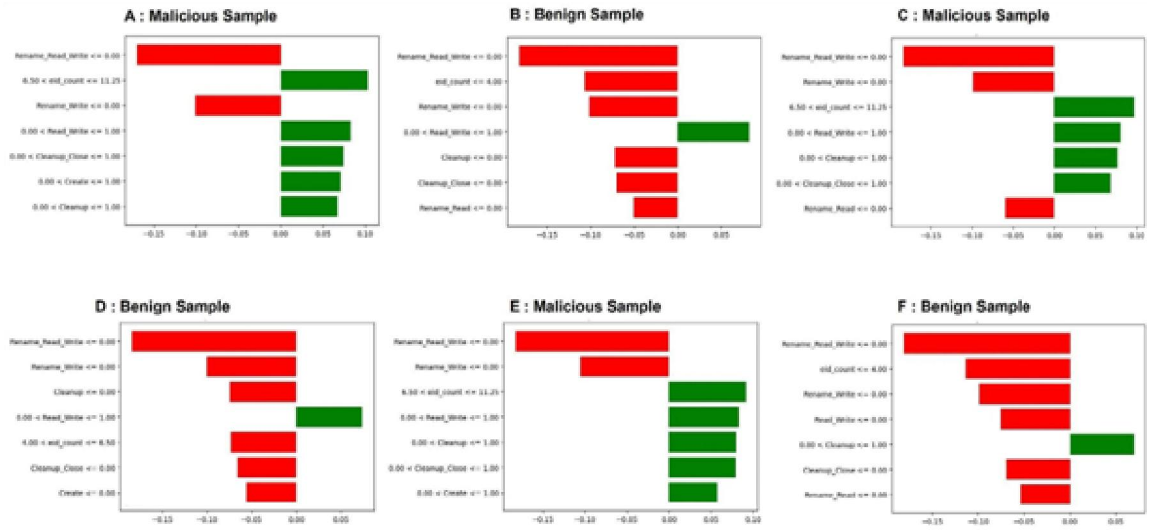

FIGURE 6. LIME explanations of 06 randomly selected samples.

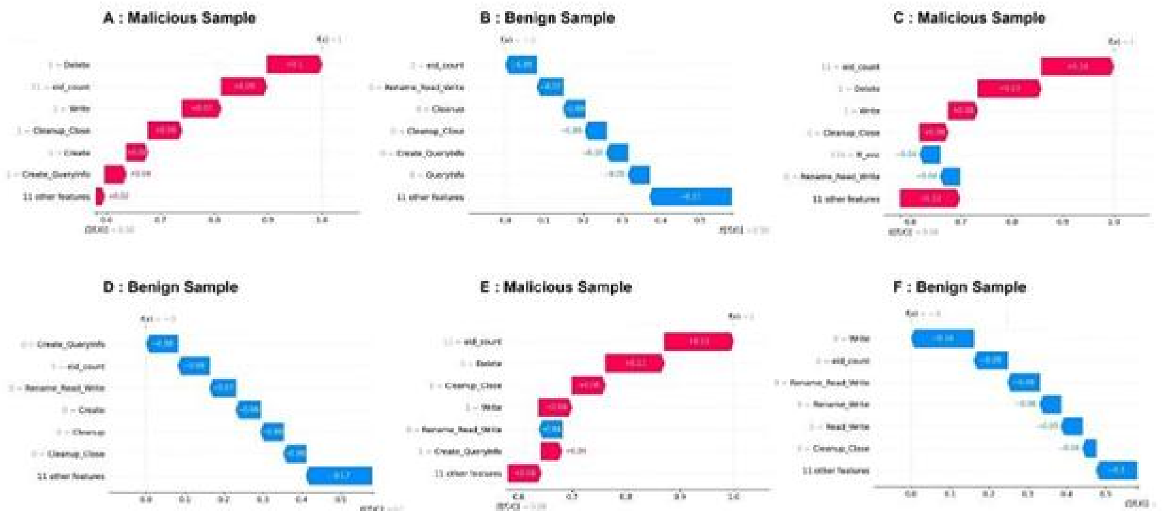

FIGURE 7. SHAP explanations of 06 randomly selected samples.

into other strains due to their similar encryption methods. This issue is compounded by the Ransomware-as-aService (RaaS) model, which produces new variants, such as Sodinokibi and Maze, with different names but nearly identical characteristics. Additionally, many ransomware generation kits, such as Hidden Tear and Stampado, are openly available on the market and are commonly used by hackers to create new ransomware variants. These variants often share similar encryption patterns but have different signatures to evade detection. As a result, ransomware created using these kits exhibits remarkably similar encryption behaviors, further complicating the accurate classification of ransomware strains.

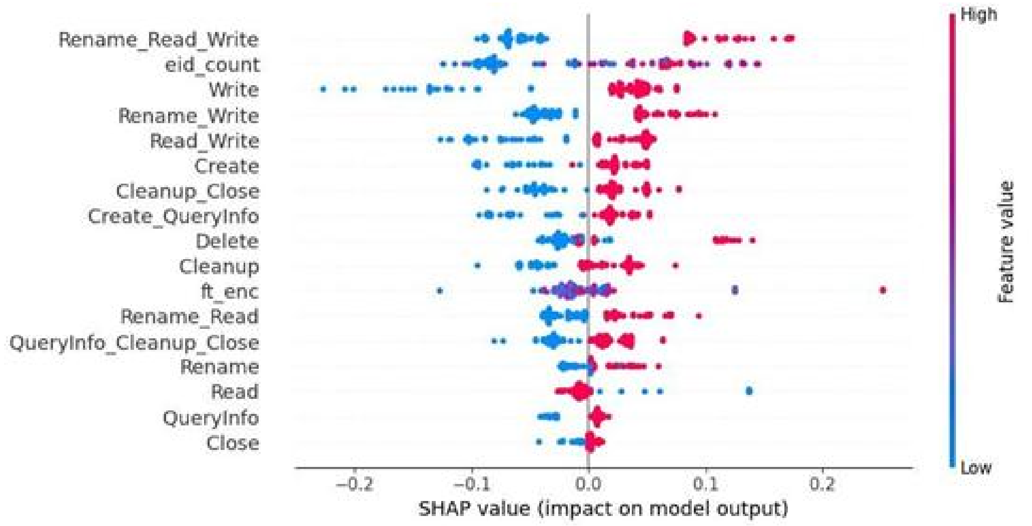

Figure 9. shows the Confusion Matrix of Random Forest Classifier.

A key aspect of XRGuard's effectiveness lies in its explainability, which aligns with Explainable AI (XAI) principles. We employed LIME and SHAP for interpretability analysis of model results. LIME provides per-prediction explanations, highlighting the most influential features that led to a particular decision. Figure 6 presents LIME explanations for six randomly selected predictions. The results confirm that LIME accurately identified key event types contributing to classification.

Malicioussamples (Figure 6 A, C, and E): The presence of eid_count, Read_Write, Cleanup_Close, and Create operations strongly contributed to the classification of ransomware. High event counts for the same file, multiple

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 2, December 2025
Impact Factor: 7.67

read-write operations, and the frequent creation of new encrypted files were observed.

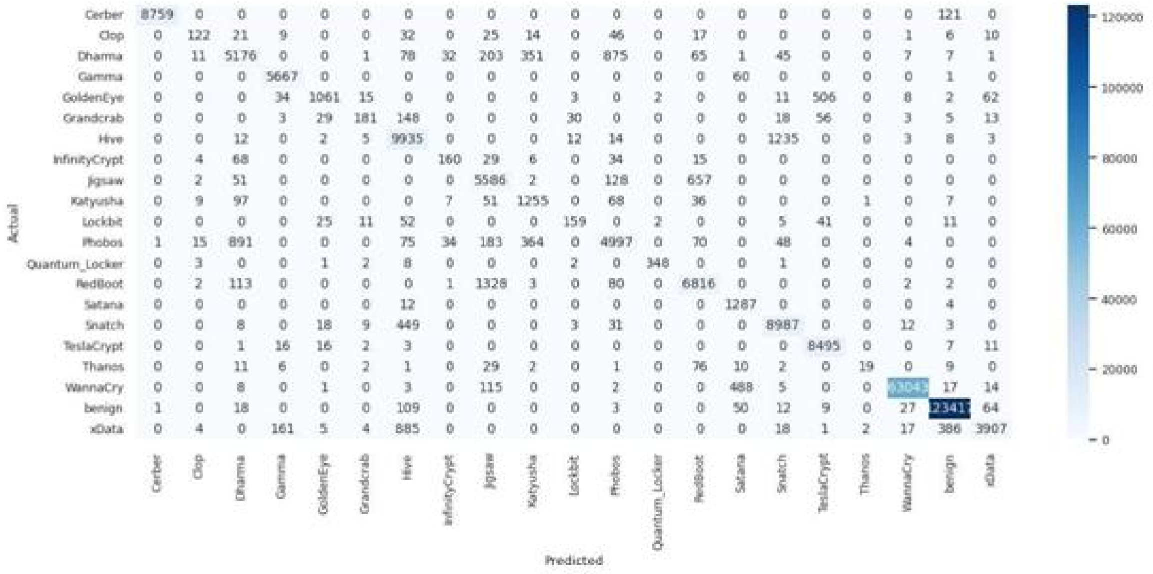

FIGURE 9. Confusion matrix of random forest classifier.

Benign samples (Figure 6 B, D, and F): Samples exhibited low eid_count values with Read_Write and Cleanup operations but lacked the Create event, leading to their classification as benign.

To further strengthen interpretability, SHAP was utilized to quantify feature importance for individual and global model predictions. SHAP's theoretical foundation ensures a fair attribution of feature contributions. The SHAP waterfall plots in Figure 7 illustrate how different features influenced classification decisions: High eid_count values combined with delete and Write operations were strong indicators of ransomware, as seen in Figure 7 (A, C, and E). In benign samples (Figure 7 B, D, and F), low eid_count values led to lower risk assessments. Interestingly, Write operations alone were not a definitive ransomware indicator unless accompanied by delete operations. Figure 8 provides a global SHAP summary plot, showing that Rename_Read_Write and eid_count had the highest influence on classification. High values of these features increased the likelihood of being classified as ransomware, while lower values of Write and Rename_Write pushed predictions toward benign classifications.

In addition to classification accuracy, XRGuard's resource efficiency was also analyzed. XRGuard's memory consumption and CPU workload are primarily influenced by the system's running tasks, particularly those involving intensive file I/O operations, such as file compression, indexing, or disk scanning. On an Intel Core i7 8th generation system with 8 GB of RAM running Windows 10, under typical usage conditions—such as internet browsing, working with Word or Office documents, or general disk browsing, XRGuard's average CPU usage ranges from 1-2 percent, with a memory footprint averaging between 30-50 MB.

VII. CONCLUSION & FUTURE WORK

In this work, XRGuard is introduced, an advanced ransomware detection framework that integrates machine learning with Explainable AI (XAI) to address the evolving cybersecurity challenges ransomware poses. XRGuard leverages Event Tracing for Windows (ETW) logs to analyze critical file I/O patterns indicative of ransomware attacks, offering a nuanced view of system behavior. XRGuard enhances detection accuracy and provides transparent and interpretable insights into its decision-making process by employing SHAP and LIME. Our experimental results demonstrate that XRGuard achieves a notable 99.69% accuracy rate with an exceptionally low false positive rate of 0.5%. Furthermore, XRGuard effectively distinguishes ransomware from benign applications, bridging the gap between high detection performance and the need for actionable explanations. In the future, the suggested anti- ransomware system could be enhanced by integrating a more diverse range of ransomware strains and benign applications in the evaluation process, thereby increasing its robustness and generalizability. Additionally, incorporating adaptive learning mechanisms to update XRGuard's models in real-time as new ransomware variants emerge would bolster its resilience against evolving threats. Implementing continuous learning approaches could further ensure XRGuard remains effective in the face of new obfuscation and evasion techniques, maintaining its relevance and accuracy in the dynamic field of cybersecurity.

DECLARATIONS

Conflict of interest: The authors declare that they have no Conflict of interest.
Data Availability Statement: Data in this research paper will be shared upon request made to the corresponding author.

REFERENCES

[1] B. Toulas. (Aug. 2024). Ransomware Rakes in Record-breaking $450 Million in First Half of 2024. [Online]. Available: https://www.bl eepingcomputer.com/news/security/ransomware-rakes-in-record-break ing-450-million-in-first-half-of-2024/

[2] S. Gulmez, A. G. Kakisim, and I. Sogukpinar, ''XRan: Explainable deep learning-based ransomware detection using dynamic analysis,'' Comput. Secur., vol. 139, Apr. 2024, Art. no. 103703. [Online]. Available: https:// www.sciencedirect.com/science/article/pii/S016740482400004X

[3] M. M. Alani, A. Mashatan, and A. Miri, ''XMal: A lightweight memorybased explainable obfuscated-malware detector,'' Comput. Secur., vol. 133, Oct. 2023, Art. no. 103409. [Online]. Available: https://www.sc iencedirect.com/science/article/pii/S016740482300319X

[4] M. Kumar, S. Darshan, S. Harshavardhan, A. Kodipalli, and T. Rao, ''Enhancing malware detection accuracy: A comparative analysis of machine learning models with explainable AI,'' in Proc. 4th Asian Conf. Innov. Technol. (ASIANCON), Aug. 2024, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10837854?casatoken=N wz qJvKCgIAAAAA:u2hyr5SihADFGakq-M1Deu8nDSL3trWTnZ80hVqY QlZfnullIbcF61JHOYhjxmE6BD6s8-97t7

[5] R. Ch, J. Manoranjini, S. Pallavi, U. Naresh, S. Telang, and S. Kiran, ''Advancing malware detection using memory analysis and explainable AI approach,'' in Proc. 2nd Int. Conf. Intell. Cyber Phys. Syst. Internet Things (ICoICI), Aug. 2024, pp. 518–523. [Online]. Available: https://ieeexp lore.ieee.org/abstract/document/10696406?casatoken=CMk-SZXmWmA AAAAA:QUQniS7vk-GH2FbdF48xWmDniH6SvE7WYHXNtc0w2Xk1 Ws5Vx0vkJijKzRlAjLc87gDzHOsm8G

[6] A. Galli, V. La Gatta, V. Moscato, M. Postiglione, and G. Sperlì, ''Explainability in AI-based behavioral malware detection systems,'' Comput. Secur., vol. 141, Jun. 2024, Art. no. 103842. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404824001433

[7] F. S. Prity, M. S. Islam, E. H. Fahim, M. M. Hossain, S. H. Bhuiyan, M. A. Islam, and M. Raquib, ''Machine learning-based cyber threat detection: An approach to malware detection and security with explainable AI insights,'' Hum.-Intell. Syst. Integr., vol. 6, no. 1, pp. 61–90, Dec. 2024, doi: 10.1007/s42454-024-00055-7.

[8] E. Baghirov, ''A comprehensive investigation into robust malware detection with explainable AI,'' Cyber Secur. Appl., vol. 3, Dec. 2025, Art. no. 100072. [Online]. Available: https://www.sciencedirect.com/scien ce/article/pii/S2772918424000389

[9] M.E.Ahmed,H.Kim,S.Camtepe,andS.Nepal,''Peeler:Profiling kernellevel events to detect ransomware,'' in Computer Security—ESORICS 2021, E. Bertino, H. Shulman, and M. Waidner, Eds., Cham, Switzerland: Springer, 2021, pp. 240–260.

[10] S. Rana, N. Kumar, A. Handa, and S. K. Shukla, ''Automated windows behavioral tracing for malware analysis,'' Secur. Privacy, vol. 5, no. 6, Nov. 2022, Art. no. e253. [Online]. Available: https://onlinel ibrary.wiley.com/doi/pdf/10.1002/spy2.253

[11] K. Begovic, A. Al-Ali, and Q. Malluhi, ''Cryptographic ransomware encryption detection: Survey,'' Comput. Secur., vol. 132, Sep. 2023, Art.

no. 103349. [Online]. Available: https://www.sciencedirect.com/scien ce/article/pii/S0167404823002596

[12] S. Malik, B. Shanmugam, K. Krishnan, and S. Azam, ‘‘Critical feature selection for machine learning approaches to detect ransomware,’’ Int. J. Comput. Digit. Syst., vol. 11, no. 1, pp. 1167–1176, Mar. 2022. [Online]. Available: https://journal.uob.edu.bh/handle/123456789/4610

[13] A. Kharraz, S. Arshad, C. Mulliner, W. Robertson, and E. Kirda, ‘‘UNVEIL: A large-scale, automated approach to detecting ransomware,’’ in Proc. 25th USENIX Secur. Symp. (USENIX Secur.), 2016, pp. 1–17.

[14] A. Kharraz and E. Kirda, ‘‘Redemption: Real-time protection against ransomware at end-hosts,’’ in Research in Attacks, Intrusions, and Defenses (Lecture Notes in Computer Science), M. Dacier, M. Bailey, M. Polychronakis, and M. Antonakakis, Eds., Cham, Switzerland: Springer, 2017, pp. 98–119.

[15] A. Arabo, R. Dijoux, T. Poulain, and G. Chevalier, ‘‘Detecting ransomware using process behavior analysis,’’ Proc. Comput. Sci., vol. 168, pp. 289– 296, Jan. 2020. [Online]. Available: https://www.sciencedi rect.com/science/article/pii/S1877050920303884

[16] M. A. Ayub, A. Continella, and A. Siraj, ‘‘An I/O request packet (IRP) driven effective ransomware detection scheme using artificial neural network,’’ in Proc. IEEE 21st Int. Conf. Inf. Reuse Integr. Data Sci. (IRI), Las Vegas, NV, USA, Aug. 2020, pp. 319–324. [Online]. Available: https://ieeexplore.ieee.org/document/9191509/

[17] A. Continella, A. Guagnelli, G. Zingaro, G. De Pasquale, A. Barenghi, S. Zanero, and F. Maggi, ‘‘ShieldFS: A self-healing, ransomwareaware filesystem,’’ in Proc. 32nd Annu. Conf. Comput. Secur. Appl., Los Angeles, CA, USA, Dec. 2016, pp. 336–347. [Online]. Available: https://dl.acm.org/doi/10.1145/2991079.2991110

[18] G. Nalinipriya, V. Govarthini, S. Kayalvizhi, S. Christika, J. Vishvaja, and K. R.
R. Amara, ‘‘DefendR—An advanced security model using mini filter in unix multi-operating system,’’ in Proc. 8th Int. Conf. Smart Struct. Syst. (ICSSS), Apr. 2022, pp. 1–6.

[19] J. Morris, D. Lin, and M. Smith, ‘‘Fight virus like a virus: A new defense method against file-encrypting ransomware,’’ 2021, arXiv:2103.11014.

[20] N. Bailluet, H. Bouder, and D. Lubicz, ‘‘Ransomware detection using Markov chain models over file headers,’’ in Proc. 18th Int. Conf. Secur. Cryptogr., 2021, pp. 403–411. [Online]. Available: https://www.scit epress.org/DigitalLibrary/Link.aspx?doi=10.5220/0010513100002998

[21] A. Alqahtani and F. T. Sheldon, ‘‘A survey of crypto ransomware attack detection methodologies: An evolving outlook,’’ Sensors, vol. 22, no. 5, p. 1837, Feb. 2022. [Online]. Available: https://www.mdpi.com/14248220/22/5/1837

[22] Y. S. Joshi, H. Mahajan, S. N. Joshi, K. P. Gupta, and A. A. Agarkar, ‘‘Signature-less ransomware detection and mitigation,’’ J. Comput. Virol. Hacking Techn., vol. 17, no. 4, pp. 299–306, Dec. 2021. [Online]. Available: https://link.springer.com/10.1007/s11416-021-00384-0

[23] J. Pont, B. Arief, and J. Hernandez-Castro, ‘‘Why current statistical approaches to ransomware detection fail,’’ in Information Security (Lecture Notes in Computer Science), W. Susilo, R. H. Deng, F. Guo, Y. Li, and R. Intan, Eds., Cham, Switzerland: Springer, 2020, pp. 199–216.

[24] D. M. Oses, E. Berrueta, E. Magaña, and M. Izal, ‘‘A chronological evolution model for crypto-ransomware detection based on encrypted file-sharing traffic,’’ SSRN Electron. J., 2022. [Online]. Available: https://www.ssrn.com/abstract=4074557

[25] E. Berrueta, D. Morato, E. Magaña, and M. Izal, ‘‘Crypto-ransomware detection using machine learning models in file-sharing network scenarios with encrypted traffic,’’ Expert Syst. Appl., vol. 209, Dec. 2022, Art. no. 118299. [Online]. Available: https://www.sci encedirect.com/science/article/pii/S0957417422014312

[26] T. Xia, Y. Sun, S. Zhu, Z. Rasheed, and K. Shafique, ‘‘Toward a networkassisted approach for effective ransomware detection,’’ ICST Trans. Secur. Saf., vol. 21, no. 24, Jul. 2018, Art. no. 168506. [Online]. Available: https://eudl.eu/doi/10.4108/eai.28-1-2021.168506

[27] M. Hirano, R. Hodota, and R. Kobayashi, ‘‘RanSAP: An open dataset of ransomware storage access patterns for training machine learning models,’’ Forensic Sci. Int., Digit. Invest., vol. 40, Mar. 2022, Art. no. 301314. [Online]. Available: https://www.scienc edirect.com/science/article/pii/S2666281721002390

[28] F. Tang, B. Ma, J. Li, F. Zhang, J. Su, and J. Ma, ''RansomSpector: An introspection-based approach to detect crypto ransomware,'' Comput. Secur., vol. 97, Oct. 2020, Art. no. 101997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/ S0167404820302704
[29] A. Alraizza and A. Algarni, ''Ransomware detection using machine learning: A survey,'' Big Data Cognit. Comput., vol. 7, no. 3, p. 143, Aug. 2023. [Online]. Available: https://www.mdpi.com/2504-2289/7/3/143
[30] K. Thummapudi, P. Lama, and R. V. Boppana, ''Detection of ransomware attacks using processor and disk usage data,'' IEEE Access, vol. 11, pp. 51395–51407, 2023. [Online]. Available: https://ieeexplo re.ieee.org/document/10132856
[31] G. O. Ganfure, C.-F. Wu, Y.-H. Chang, and W.-K. Shih, ''DeepWare: Imaging performance counters with deep learning to detect ransomware,'' IEEE Trans. Comput., vol. 72, no. 3, pp. 600–613, Mar. 2023. [Online]. Available: https://ieeexplore.ieee.org/document/9770351/
[32] S. Aurangzeb, R. N. B. Rais, M. Aleem, M. A. Islam, and M. A. Iqbal, ''On the classification of microsoft-windows ransomware using hardware profile,'' PeerJ Comput. Sci., vol. 7, p. e361, Feb. 2021.
[33] N. Pundir, M. Tehranipoor, and F. Rahman, ''RanStop: A hardwareassisted runtime crypto-ransomware detection technique,'' 2020, arXiv:2011.12248.
[34] A. M. R. AlSobeh, ''OSM: Leveraging model checking for observing dynamic behaviors in aspect-oriented applications,'' Online J. Commun. Media Technol., vol. 13, no. 4, Oct. 2023, Art. no. e202355. [Online]. Available: https://www.ojcmt.net/article/osm-leveraging-model- checkingfor-observing-dynamic-behaviors-in-aspect-oriented-applications-13771
[35] N. Capuano, G. Fenza, V. Loia, and C. Stanzione, ''Explainable artificial intelligence in CyberSecurity: A survey,'' IEEE Access, vol. 10, pp. 93575– 93600, 2022.
[36] W. Esam Noori and A. S. Albahri, ''Towards trustworthy myopia detection: Integration methodology of deep learning approach, XAI visualization, and user interface system,'' Appl. Data Sci. Anal., vol. 2023, pp. 1–15, Feb. 2023. [Online]. Available: https://mesopo tamian.press/journals/index.php/ADSA/article/view/100
[37] S. Y. Mohammed and M. Aljanabi, ''Advancing translation quality assessment: Integrating AI models for real-time feedback,'' EDRAAK, vol. 2024, pp. 1–7, Jan. 2024. [Online]. Available: https://peninsulapress.ae/Journals/index.php/EDRAAK/article/view/42
[38] MSDN. (Jan. 2020). ETW Framework Conceptual Tutorial—Message Analyzer. [Online]. Available: https://learn.microsoft.com/en- us/messageanalyzer/etw-framework-conceptual-tutorial
[39] Blake. (2024). Monitoring File Mods Through ETW and Velociraptor. [Online]. Available: https://bmcder.com/blog/event-tracing-for- windowsmonitoring-file-and-process-interactions
[40] M. A. Alvi and Z. Jalil, ''RansomGuard: A framework for proactive detection and mitigation of cryptographic windows ransomware,'' J. Comput. Virol. Hacking Techn., vol. 20, no. 4, pp. 867–884, Sep. 2024, doi: 10.1007/s11416-024-00539-9.
[41] M. T. Ribeiro, S. Singh, and C. Guestrin, '''Why should I trust you?': Explaining the predictions of any classifier,'' in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, New York, NY, USA, Aug. 2016, pp. 1135–1144, doi: 10.1145/2939672.2939778.
[42] S. M. Lundberg and S.-I. Lee, ''A unified approach to interpreting model predictions,'' in Proc. Adv. Neural Inf. Process. Syst., vol. 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., 2017. [Online]. Available: https://proceedings. neurips.cc/paperfiles/paper/2017/file/8a20a862197863 2d76c43dfd28b67767-Paper.pdf
[43] C. Molnar, Interpretable Machine Learning, 2nd ed., San Francisco, CA, USA: GitHub, 2022. [Online]. Available: https://christophm.git hub.io/interpretable-ml-book
[44] M. Cen, F. Jiang, X. Qin, Q. Jiang, and R. Doss, ''Ransomware early detection: A survey,'' Comput. Netw., vol. 239, Feb. 2024, Art. no. 110138. [Online]. Available: https://www.sciencedirect.com/ science/article/pii/S138912 8623005832
[45] VirusTotal. (2024). VirusTotal—Home. [Online]. Available: https://www. virustotal.com/gui/home/upload
[46] MalwareBazaar. (2024). MalwareBazaar| Malware Sample Exchange. [Online]. Available: https://bazaar.abuse.ch/