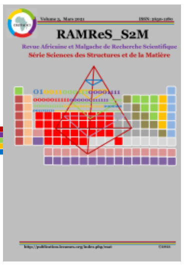
RAMReS Sciences des Structures et de la Matière Vol. 9, N°2 (2025) 70 - 87
http://publication.lecames.org/index.php/mat ISSN: 2630-1180



RAMReS_S2M
Revue Africaine et Malgache de Recherche Scientifique
Série Sciences des Structures et de la Matière

An intelligent framework for ransomware detection leveraging behavioral Machine Learning and live threat feeds

Puneet Chauhan*, Shashiraj Teotia

Department of Computer Science and Applications, Swami Vivekanand Subharti University, Meerut, U.P., India

Received: 27 August 2025 / Received in revised form: 28 October 2025 / Accepted: 24 November 2025

***Abstract*:**

Ransomware is a serious and evolving threat to global cybersecurity, utilizing advanced defense techniques that render traditional signature-based defenses ineffective. This paper proposes a new framework for real-time behavioral analysis and response to ransomware (R2BAR), which integrates machine learning with live threat intelligence feeds to enable proactive detection and automated mitigation. The R2BAR framework uses an ensemble approach, which combines a lightweight gradient boosting (XGBoost) model for efficient initial screening with a Long Short-Term Memory (LSTM) network for deep sequential analysis of API call patterns. Detection accuracy is further enhanced by dynamically correlating system behavior with real-time threat intelligence. Experimental evaluation shows that this framework achieves an F1-score of 98.1% and an area under the ROC curve of 0.998, while maintaining a low mean response time (TTR) of 2.35 seconds. This rapid response effectively breaks encryption before significant data loss occurs. The results confirm that the proposed solution addresses the primary limitations of existing methods by striking a balance between high accuracy, operational speed, and interpretability, creating a robust blueprint for the next generation of autonomous ransomware protection systems.

***Keywords*:** Ransomware detection; Behavioral analysis; Machine Learning (ML); Threat intelligence; Real-time response; Explainable AI (XAI).

1. Introduction

The digital landscape is facing a relentless onslaught of ransomware, a widespread and rapidly evolving form of cyber extortion that has paralyzed critical infrastructure, multinational corporations, and healthcare systems around the world. Developments ranging from simple data encryption to complex double and triple

* Corresponding author:
Email address: *puneet.c1106@gmail.com (P. Chauhan)*

https://doi.org/10.70974/mat0922570

extortion schemes, often facilitated by ransomware-as-a-service (RaaS) platforms, have significantly reduced the barriers to entry for cybercriminals while also increasing the complexity of attacks [1, 2]. Traditional signature-based detection and static analysis have proven severely inadequate against these threats, which use polymorphism, metamorphism, and advanced obfuscation to evade traditional security measures. Due to this shortcoming, there is a need for a paradigm shift towards more dynamic, intelligent and proactive security solutions that are able to detect and mitigate ransomware attacks in their early stages. By monitoring execution time activities such as file system operations, API calls, and network communications, it is possible to identify the malicious intent of ransomware, no matter how evasive it is. The integration of machine learning (ML) and deep learning (DL) has further revolutionized this approach, enabling the automated discovery of complex and subtle patterns indicative of ransomware behavior from massive system data flows [3]. Techniques ranging from ensemble methods such as gradient boosting to advanced neural architectures such as long short-term memory (LSTM) networks and Transformers have demonstrated remarkable success in detecting previously unknown variants [4–6]. However, several proposed ML solutions are forced to trade off speed to avoid high computational overheads, elevated false positive rates, or allowing data encryption [7].

Furthermore, a separate identification mechanism is no longer sufficient. The modern threat landscape demands contextual awareness. Threat intelligence feeds (TIFs) provide invaluable external context, offering real-time data on indicators of compromise (IoCs), attacker strategies, techniques, and procedures (TTPs), and known malicious infrastructure. Correlating internal behavioral anomalies with this external intelligence can dramatically increase identification accuracy and reduce false positives. However, effective integration of these feeds presents its own challenges, including processing massive data, assessing intelligence quality and relevance, and automating the correlation process at machine speed [8, 9].

This research addresses these shortcomings by proposing a new, holistic framework for real-time behavioral analysis and ransomware response (R2BAR). The main contribution of this work is the design and implementation of an integrated system that coordinates a multi-model ML detection engine with automated threat intelligence correlation to achieve high-reliability, low-latency ransomware detection [10]. The framework uses an ensemble approach that leverages a lightweight gradient boosting (XGBoost) model for rapid initial detection and a deep LSTM-based network for sequential analysis of sophisticated attacks, as well as enriching decisions with real-time threat context. Importantly, this framework incorporates explainable AI (XAI) principles to provide transparent justification for its actions [11, 12] and an automated response orchestrator to execute proportionate control measures, thereby transitioning from mere detection to actionable response.

2. Literature review

This literature review followed a systematic approach to identify, evaluate, and synthesize relevant research on ransomware detection and response published over last three years. The selection criteria gave preference to peer-reviewed journal articles, conference proceedings and technical reports focused on behavioral analytics, machine learning approaches, threat intelligence integration and real-time detection frameworks. Sources were collected from major scientific databases, including IEEE.

The analytical framework is designed to extract key insights from each source regarding the proposed detection methods, data sources, attribute extraction techniques, AI/ML models, evaluation metrics, and integration capabilities with threat intelligence systems. The focus was on approaches addressing real-time operational constraints, adaptability to new ransomware variants, and practical implementation challenges. The synthesis of findings was organized thematically to highlight technological trends, methodological advances, and research gaps in the current literature landscape.

Alzahrani *et al.* [4] proposed Ransom Former, a new cross-modal transformer architecture that combines byte sequence and API-level features for better ransomware detection. This approach leverages self-attention mechanisms to capture long-term dependencies in ransomware behavioral patterns, achieving better performance than traditional deep learning models. This architecture processes sequences of bytes from executable files as well as sequences of API calls captured during execution, creating a comprehensive behavioral profile that is resistant to obfuscation techniques commonly used by modern ransomware [4].

Gómez-Hernández and García-Teodoro [11], developed a lightweight detection system for Android environments based on reactive monitoring of honeyfiles. This approach combines deep learning with strategic deception techniques, placing fake files throughout the system and using neural networks to analyze access patterns. When these honeyfiles are accessed suspiciously, the system triggers alerts, providing an effective mechanism for detecting ransomware that evades traditional detection methods. The computational efficiency of the model makes it particularly suitable for resource-limited mobile devices, addressing a significant gap in ransomware protection on Android [11].

Lee *et al.* [13] developed an adaptive graph neural network approach that learns from system call graphs and process trees to identify ransomware behavior patterns. This method effectively captures the relational aspects of system activities that characterize ransomware attacks, such as fast file encryption sequences and unusual process spawning behavior. The graph-based framework allows learning invariant patterns across different ransomware families, providing robust detection capabilities even when encountering previously unobserved variants [13, 14].

Ahmed *et* al. [1] conducted extensive research on ransomware detection on Android using supervised machine learning techniques applied to network traffic data [1]. Their study used a massive dataset containing 392,035 network traffic records, which included 10 different types of ransomware and benign traffic. Through careful feature engineering and evaluation of multiple algorithms, they demonstrated that decision trees achieved exceptional performance with 97.24%, 98.50% precision, and 98.45% F1 score, while support vector machines achieved 100% recall, making them ideal for reducing false negatives in critical environments [1].

Yamany *et al.* [6] proposed a holistic approach to ransomware classification that leverages static and dynamic analysis combined with visualization techniques [6]. Their method generates similarity matrices from different analysis techniques and compares them using different algorithms to classify ransomware samples into families, types, and variants. This approach was particularly effective in dealing with obfuscated ransomware samples that hinder analysis based solely on static features [6].

AlMajali *et al.* [3] developed an adaptive ransomware detection system using similarity-preserving hashing techniques that can identify known ransomware types based on their behavioral patterns rather than static signatures [3]. This approach creates cryptographic hashes of ransomware behavior profiles, allowing efficient comparisons against known threats, and facilitates privacy-preserving and secure sharing of threat intelligence between organizations [3].

Gazzan and Sheldon [7] provided an incremental mutual information selection technique for early detection of ransomware that dynamically identifies the most discriminating features during execution. This method continuously evaluates the informative value of the system's behavioral features, allowing the detection model to adapt to new ransomware types and minimize false positives. The incremental nature of this approach makes it particularly suitable for real-time implementations, where computational resources must be optimized for efficiency [7].

Cen *et al.* [5] proposed RansoGuard, an RNN-based framework that leverages sensitive pre-attack API calls for early detection of ransomware before encryption begins. By monitoring sequences of API calls involving file system operations, network communications, and cryptographic functions, the system can identify ransomware behavior with high accuracy during the early stages [5].

External threat intelligence is important for ransomware detection systems as it provides context on the initial access vectors commonly used by ransomware operators, allowing more accurate detection and alert prioritization [15]. This intelligence is critical for ransomware detection systems, as it provides context about the initial access vectors commonly employed by ransomware operators, enabling more accurate detection and prioritization of alerts.

Sakellariou *et al.* [16] addressed the challenge of measuring the quality of cyber threat intelligence (CTI) products by proposing a probabilistic framework to assess the reliability and relevance of threat intelligence feeds [16]. This approach helps security teams prioritize which intelligence sources to integrate into their detection systems based on factors such as accuracy, timeliness, and relevance to their specific industry and infrastructure. The authors emphasize that effective threat intelligence integration requires not only the use of IoCs, but also an understanding of TTPs, which provide deep contextual information for detecting ransomware campaigns, regardless of the specific malware type used [17].

Ayyoub *et al.* [18] proposed an advanced hybrid approach to detect ransomware processes in real-time within the IoT ecosystem, combining multiple machine learning models to achieve high accuracy and low latency. Their framework utilizes lightweight feature extraction techniques optimized for resource-limited IoT environments, addressing the unique challenges presented by these ecosystems where traditional security solutions often prove inadequate [18].

Sharma *et al.* [19] further advanced this field by developing specific XAI techniques for cybersecurity applications, including ransomware detection [19]. Their methods generate feature importance assessments that highlight which behavioral indicators contributed most to the identification decision, such as specific file operations, registry modifications, or network communication patterns. This transparency not only builds trust in the detection system, but also provides valuable insights to security analysts investigating potential incidents, helping them focus their analysis on the most relevant system activities [19].

2.1. Problem formulation

The increasing complexity and frequency of ransomware attacks present a serious and persistent threat to global cybersecurity. Traditional signature-based detection methods have proven inadequate against modern ransomware variants, which use advanced evasion techniques such as polymorphism, metamorphism, and encryption to avoid static detection [2]. Although recent advances in machine learning (ML) and deep learning (DL) provide promising avenues for behavioral analytics, there remains a significant gap in the development of a holistic, real-time response framework that effectively integrates these analytical capabilities with operational threat intelligence for proactive ransomware mitigation [18].

This paper addresses the fundamental problem of detecting and responding to ransomware attacks in real-time, going beyond mere detection and enabling immediate, informed countermeasures. The main problem is not just detection accuracy, but building a responsive system that can keep pace with the speed of ransomware execution, which is often measured in seconds from activation to critical file encryption.

2.2. Research gaps

Based on a review of recent literature, several specific challenges and gaps have been identified that this research aims to address:
1. Limitations of isolated detection methods. Many existing solutions operate differently. Some focus exclusively on the analysis of network traffic [1, 15], others on sequences of API calls or static file features [3, 20]. This lack of integration creates vulnerabilities, as sophisticated ransomware can avoid detection by changing only one aspect of its behavior. Effective identification requires an integrated framework that synthesizes multiple behavioral indicators [21].
2. The delay between detection and action. Many studies propose high-accuracy ML models [1, 4, 22] but do not address the critical need for real-time performance and integration with response mechanisms. Detecting ransomware with 99% accuracy is useless if the system cannot analyze behavior and trigger a response before significant damage is done. The literature shows a clear lack of frameworks designed for low-latency analysis and automated response.
3. The evolving attack landscape. Analysis of real-world ransomware campaigns such as Conti and Clop reveals complex [8, 9, 15], multi-step attack chains. Defenses focused on a single stage (e.g. encryption) are easily bypassed. A framework is needed to analyze behavior across the cyber kill chain, from initial access through exfiltration and data encryption [13].

2.3. Research objectives

The primary objective of this research is to design, prototype, and evaluate a new integrated framework for real-time behavioral analysis and response to ransomware attacks [10]. The specific objectives are:
1. To create a flexible framework for responding to ransomware attacks, based on scientific principles and best practices. This involves studying how ransomware works, understanding the tactics used by attackers and developing a plan to deal with incidents [11].
2. Develop and train machine learning models for the behavioral analysis engine, focusing on dynamic analysis-derived feature sets (API sequences, file system changes, network calls) that are most indicative of ransomware behavior.
3. Implement a prototype of the proposed framework and evaluate its performance against a diverse set of data from modern families of ransomware and benign software, measuring key metrics including accuracy, precision, recall, F1-score, and latency.

3. Methods

This section presents details of the architectural design of the proposed framework, the datasets used for training and evaluation, the feature engineering process, the implemented machine learning models, and the experimental setup for performance validation.

3.1. Proposed framework architecture

The proposed framework, called the Real-Time Ransomware Behavior Analysis and Response (R2BAR) framework, is designed as a complete modular system to detect and mitigate ransomware attacks. It consists of four interconnected main modules, as shown in Figure 1.

Data collection and preprocessing module: This module is responsible for real-time ingestion of heterogeneous data streams. A Data Collection Agent was implemented for Relevant Event Tracing for Windows (ETW) systems, with a specific focus on Microsoft-Windows-Kernel-File, Microsoft-Windows-Kernel-Process, and Microsoft-Windows-TCPIP sessions. A custom C agent was developed to subscribe to ETW providers.

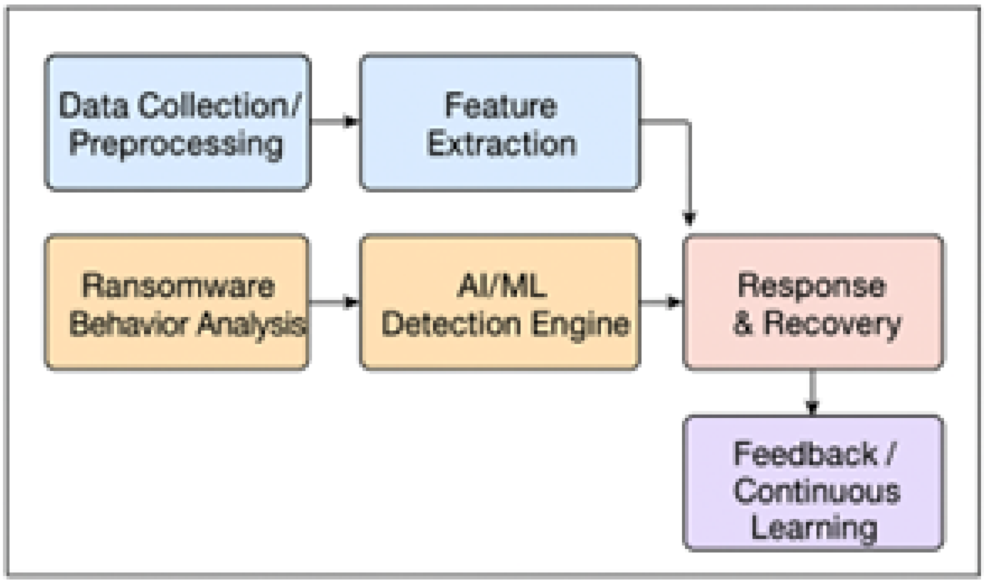

Data Collection/ Preprocessing
Feature Extraction
Ransomware Behavior Analysis
AI/ML Detection Engine
Response & Recovery
Feedback / Continuous Learning

Fig. 1. Architecture of the proposed R2BAR framework.

This agent captures system calls and network events in real-time and packages them into structured JSON messages. For Linux environments, the native Audit daemon is configured with custom rules to log similar system calls and file access events. Event streams from both operating systems are then published to a local Apache Kafka topic, providing a durable, high-throughput buffer for use by preprocessing modules, ensuring no data loss during load spikes.

Data sources: System-level behavioral data (Windows ETW events or Linux Auditd logs), including API call sequences, file system operations (create, read, write, delete, rename), registry access, process creation and termination, and network connection events. Additionally, it ingests external Threat Intelligence Feeds (TIFs) in structured (STIX/TAXII) and unstructured formats.

Preprocessing: Raw logs are parsed, normalized, and timestamped. TIFs are processed using Natural Language Processing (NLP) techniques, such as Named Entity Recognition (NER), to extract actionable Indicators of Compromise (IoCs) like malicious IPs, domains, and file hashes, following methodologies similar to Sakellariou *et al.* [16].

Behavioral analysis and threat intelligence correlation engine: This is the core analytical component.

Feature extraction: A sliding window technique is applied to the stream of events to create sequential data instances. Features are extracted per process and include: frequency of file encryption-related API calls, entropy of written files, spatial and temporal patterns of file modifications, and network call sequences to known-bad destinations [23].

Threat intelligence enrichment: Each analyzed process is enriched in real-time by correlating its behavior (e.g., file hashes, network endpoints) with the ingested IoCs from TIFs. A confidence score is assigned based on the quality and recency of the matched intelligence.

Multi-Model Machine Learning detection core: This module hosts the ensemble of detection models [21].

Real-Time Classifier: A lightweight Gradient Boosting Machine (XGBoost) model is deployed for initial, low-latency classification based on handcrafted features. Its efficiency makes it ideal for first-pass filtering [24].

Deep Learning Analyzer: For processes flagged by the initial classifier, a more sophisticated Long Short-Term Memory (LSTM) Recurrent Neural Network analyzes the sequence of API calls to capture long-range temporal dependencies and subtle attack patterns, as inspired by Cen *et al.* [5].

Ensemble decision mechanism: The final detection score is a weighted aggregation of the scores from the XGBoost model, the LSTM model, and the TI correlation score. A process is classified as ransomware if this aggregated score exceeds a dynamically adjusted threshold.

Explainable AI (XAI) and Response Orchestration Module: This module ensures actionable outcomes.

Explainable AI: Upon a positive detection, the SHAP (SHapley Additive exPlanations) framework is employed to generate a report detailing which specific features (e.g., ”called CryptEncrypt 50 times in 2 seconds”, ”connected to IP [malicious IP]”) contributed most to the decision, addressing the ”black box” problem.

Automated response: The framework integrates with endpoint detection and response (EDR) tools via APIs to execute automated containment actions. Responses are proportional to the confidence score, ranging from alerting security personnel to automatically isolating the endpoint from the network and suspending the malicious process.

3.2. Dataset curation and preparation

A hybrid dataset was curated to train and evaluate the models, combining public benchmarks and generated attacks.

Ransomware Samples: The dataset includes behavioral traces from a diverse set of ransomware families (e.g., LockBit, Conti, BlackCat, Ryuk) obtained from public repositories like VirusShare and PolySwarm.

Benign Samples: System activities were collected from the execution of legitimate software (browsers, office suites, system utilities) and from public datasets like the UNSW-NB15 and CICIDS2017 for normal network traffic.

Synthetic Data Generation: To address class imbalance and simulate zero-day attacks, adversarial techniques were used to generate synthetic ransomware behavior sequences that evade simple feature-based detection.

Ransomware samples: Behavioral traces were collected from 1,850 unique ransomware samples spanning 15 different families, including Lockbit (v2.0, v3.0), Conti, Blackcat (ALPHV), Ryuk, Phobos, and Dharma. These samples were obtained from VirusShare (dataset hash: 20240501) and PolySwarm (query: ”ransomware file type: exe”) public repositories, ensuring a diverse representation of prevalent and emerging threats for the period 2022–2024.

Benign Samples: System activities were collected from two primary sources: 1) running over 120 common legitimate applications (e.g. Google Chrome, Microsoft Office Suite, Adobe Reader, VLC media player) on a clean install of Windows 11 and 2) publicly available datasets UNSW-NB15 and CICIDS2017, from which common network traffic and related system process records were extracted and integrated. were done.

Synthetic Data Generation: To simulate zero-day evasion attempts, we employed the IBM Adversarial Robustness Toolbox (ART) to generate adversarial examples. Specifically, the Fast Gradient Sign Method (FGSM) was applied to perturb feature vectors of known ransomware samples, creating synthetic variants designed to mislead the statistical XGBoost classifier without altering the core malicious behavior, thereby increasing the model’s robustness.

The final dataset comprised 150,000 behavioral instances (60% ransomware, 40% benign), which was split into 70% for training, 15% for validation, and 15% for testing.

3.3. Feature engineering

Features were engineered to capture the definitive behavioral fingerprints of ransomware, drawing from the literature.

Static Features: File entropy, digital signature status, packer detection.

Dynamic Features:

File System Features: Rate of file modifications, file type targeting (e.g., concentration on `.docx`, `.pdf`), similarity of modified files (honeyfile access). Similarity of modified files was quantified using Jaccard similarity between the set of file extensions accessed by the process and a predefined set of high-value target extensions (`.docx`, `.pdf`, `.xlsx`, `.jpg`, `.sql`, `.db`). Honeyfile access was implemented by monitoring read/write operations on decoy files with enticing names (e.g., `passwords.txt`, `financial_records.xlsx`) placed in strategic user directories; any access triggered a binary feature flag.

API Call Features: Frequency of cryptographic and file deletion APIs, sequence patterns of system calls. Frequency counts were normalized per second to account for varying process lifetimes. Sequence patterns were captured by creating n-grams (n=3) of API calls and calculating their frequency relative to a baseline of benign software.

Network Features: Communication with IPs/domains flagged in TIFs, traffic volume to unknown destinations.

Process Features: Process injection attempts, attempts to disable security services.

3.4. Machine Learning models and training

XGBoost: The model was trained using a binary objective function. Hyperparameters (learning rate, max depth, number of estimators) were optimized via Bayesian optimization on the validation set.

LSTM: The network was built with two LSTM layers (128 and 64 units respectively) followed by dropout layers (rate=0.5) to prevent overfitting, and a dense output layer with a sigmoid activation function. It was trained using the Adam optimizer with a binary cross-entropy loss function [25].

3.5. Experimental setup and evaluation metrics

Implementation: The entire framework was prototyped in Python. The data collection agent was implemented in C++ for low-level system access on Windows 10/11 systems.

Hardware: Experiments were conducted on a server with an Intel Xeon Silver 4210 CPU, 64 GB RAM, and an NVIDIA RTX A5000 GPU to assess performance under enterprise-grade conditions.

Evaluation Metrics: The models were evaluated based on standard metrics:

Accuracy, Precision, Recall, and F1-Score: To measure overall performance and the balance between false positives and false negatives.

Area Under the Receiver Operating Characteristic Curve (AUC-ROC): To measure the model's ability to distinguish between classes. Detection and Response Latency: The critical metric of Time-to-Detect (TTD) and Time-to-Respond (TTR) was measured from the onset of malicious activity to the initiation of a response action [16].

Computational Overhead: CPU and RAM usage were monitored on the host system during operation to evaluate practical deployability.

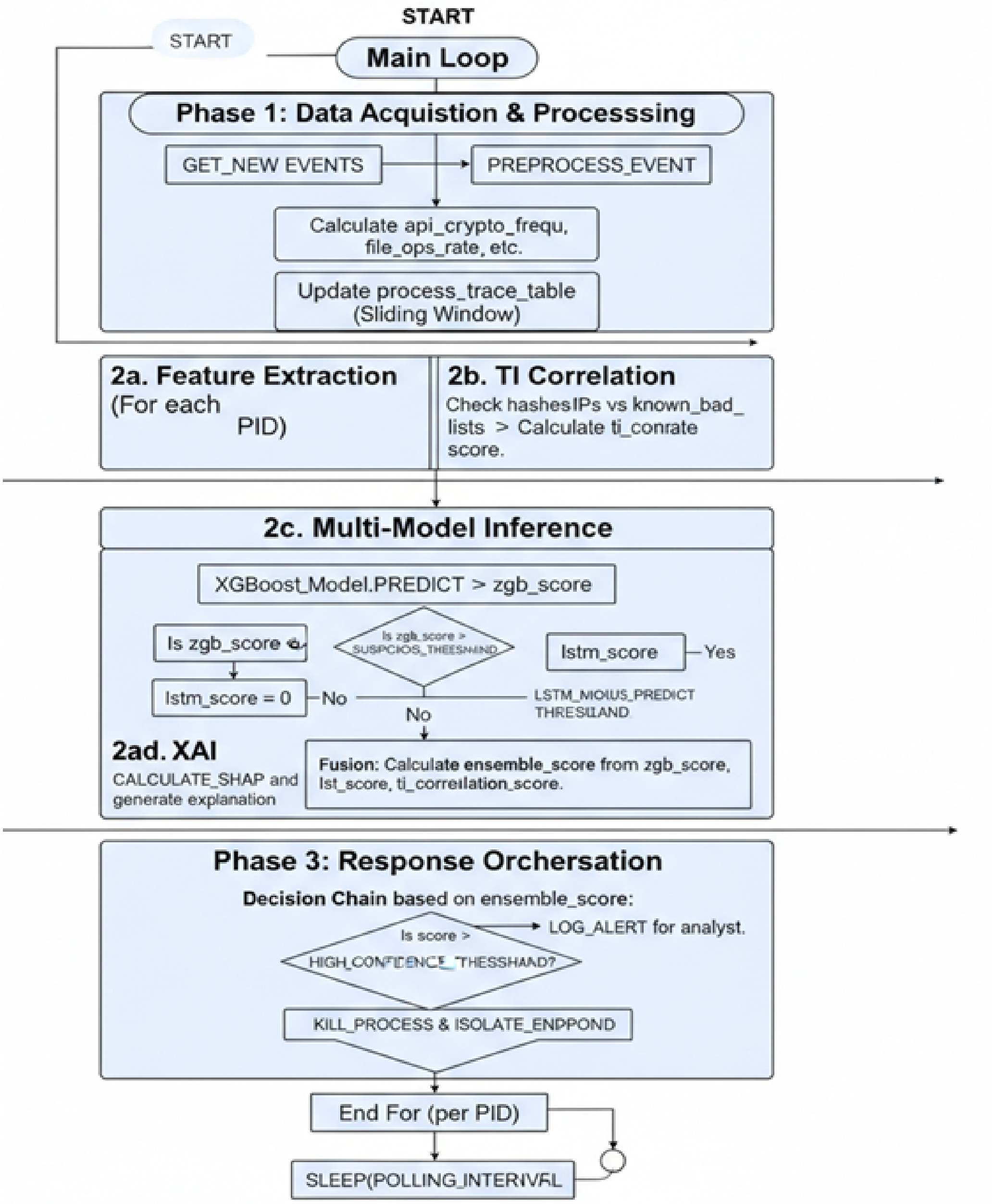

START
START
Main Loop
Phase 1: Data Acquistion & Processsing
GET_NEW EVENTS
PREPROCESS_EVENT
Calculate api_crypto_frequ, file_ops_rate, etc.
Update process_trace_table (Sliding Window)
2a. Feature Extraction (For each PID)
2b. TI Correlation
Check hashesIPs vs known_bad_ lists > Calculate ti_conrate score.
2c. Multi-Model Inference
XGBoost_Model.PREDICT > zgb_score
Is zgb_score ↻
Is zgb_score > SUSPCIOS_THEESNIND
lstm_score
Yes
lstm_score = 0
No
LSTM_MOIUS_PREDICT THRESILAND
No
2ad. XAI
CALCULATE_SHAP and generate explanation
Fusion: Calculate ensemble_score from zgb_score, lst_score, ti_correilation_score.
Phase 3: Response Orchersation
Decision Chain based on ensemble_score:
Is score > HIGH_CONFIDENCE_THESSHAND?
LOG_ALERT for analyst.
KILL_PROCESS & ISOLATE_ENPPOND
End For (per PID)
SLEEP(POLLING_INTERIVAL

Fig. 2. Flowchart of the proposed Real-Time Ransomware Behavioral Analysis and Response R2BAR.

3.6. Proposed algorithm: Real-Time Ransomware Behavioral Analysis and Response (R2BAR)

In Figure 2, the flowchart of the Real-Time Ransomware Behavioral Analysis and Response (R2BAR) Algorithm. The process illustrates the continuous loop of data acquisition, multi-stage analysis, and proportional response orchestration.

ALGORITHM R2BAR: Real-Time Ransomware Behavioral Analysis and Response Framework

INPUT:

raw_event_stream: Continuous feed of system events (e.g., ETW, Auditd logs)

threat_intel_feeds: Structured (STIX/TAXII) and unstructured sources of IoCs

trained_models: Pre-trained XGBoost and LSTM models

config: Tunable parameters (e.g., thresholds, weights, window size)

OUTPUT:

alerts: Real-time alerts with confidence scores and explanations

actions: Automated containment actions (e.g., process kill, isolation)

INITIALIZATION:

```
LOAD trained_models (XGBoost_Model, LSTM_Model)
// For unstructured TI feeds (e.g., blogs, reports), an NLP pipeline
// (NER) extracts IoCs like IPs, domains, and file hashes.
PARSE_AND_LOAD threat_intel_feeds INTO known_bad_hashes, known_bad_ips
INITIALIZE process_trace_table // Hash table to store recent events per Process ID
MAIN LOOP:
WHILE TRUE:
// — Phase 1: Data Acquisition & Preprocessing —
batch = GET_NEW_EVENTS(raw_event_stream) // Consumes from Kafka topic
FOR EACH event IN batch:
// PREPROCESS_EVENT: Parses raw log, normalizes field names,
// timestamps, and filters irrelevant event types (e.g., mouse movements).
normalized_event = PREPROCESS_EVENT(event)
batch = GET_NEW_EVENTS(raw_event_stream)
FOR EACH event IN batch:
normalized_event = PREPROCESS_EVENT(event)
pid = normalized_event.process_id
// Maintain a sliding window of events per process
IF pid NOT IN process_trace_table:
INITIALIZE process_trace_table[pid] AS NEW_QUEUE(max_size=config.WINDOW_SIZE)
PUSH normalized_event TO process_trace_table[pid]
END FOR
// — Phase 2: Process Behavior Analysis —
FOR EACH pid IN process_trace_table:
trace = process_trace_table[pid]
IF LENGTH(trace) < config.MIN_EVENTS: CONTINUE
// 2a. Feature Extraction for Statistical Model
features = EXTRACT_FEATURES(trace)
features.api_crypto_freq = COUNT_CALLS (trace, [’CryptEncrypt’, ’CryptGenKey’,...])
features.file_ops_rate = COUNT_CALLS(trace, [’CreateFile’, ’WriteFile’, ’DeleteFile’, ...])
features.file_entropy_avg = AVERAGE([e.entropy FOR e in trace IF e.entropy])
features.net_conn_count = COUNT_CALLS (trace, [’connect’, ’send’, ...])
// 2b. Threat Intelligence Correlation
ti_correlation_score = 0.0
FOR event IN trace:
IF event.file_hash IN known_bad_hashes:
ti_correlation_score += config.TI_HASH_WEIGHT
IF event.dest_ip IN known_bad_ips:
ti_correlation_score += config.TI_IP_WEIGHT
END FOR
ti_correlation_score = MIN(ti_correlation_score, config.TI_MAX_SCORE) // Cap the score
// 2c. Multi-Model Inference & Ensemble Scoring
xgb_score = XGBoost_Model.PREDICT (features) // Fast, first-pass analysis
lstm_score = 0.0
IF xgb_score > config.SUSPICIOUS_THRESHOLD:
api_sequence = EXTRACT_API_SEQUENCE (trace) // Create temporal sequence
lstm_score = LSTM_Model.PREDICT (api_sequence) // Deep, sequential analysis
END IF
// Fuse scores from all components
ensemble_score = (config.ALPHA * xgb_score) +
(config.BETA * lstm_score) +
(config.GAMMA * ti_correlation_score)
// 2d. Explainable AI (XAI) Justification
explanation = “Decision rationale: ”
IF ensemble_score > config.MALICIOUS_THRESHOLD:
shap_values = CALCULATE_SHAP (XGBoost_Model, features)
top_features = GET_TOP_N_FEATURES (shap_values, n=3)
explanation += “Process classified as ransomware. Top contributing features:”+
top_features + “.TI correlation score:” + ti_correlation_score
ELSE:
```

```
            explanation += “Process classified as benign.”
        END IF
        // — Phase 3: Response Orchestration —
        IF ensemble_score > config.HIGH_CONFIDENCE_THRESHOLD:
            // Autonomous containment for high-confidence detections
            KILL_PROCESS(pid)
            ISOLATE_ENDPOINT(get_host_ip(pid))
            LOG_ALERT (“CRITICAL”, pid, ensemble_score, explanation, “AUTO-CONTAINED”)
        ELSE IF ensemble_score > config.MEDIUM_CONFIDENCE_THRESHOLD:
            // Terminate process but don’t isolate host
            KILL_PROCESS(pid)
            LOG_ALERT (“HIGH”, pid, ensemble_score, explanation, “PROCESS_TERMINATED”)
        ELSE IF ensemble_score > config.LOW_CONFIDENCE_THRESHOLD:
            // Create ticket for analyst review with full explanation
            LOG_ALERT (“MEDIUM”, pid, ensemble_score, explanation, “TICKET_CREATED”)
        END IF
    END FOR // End loop per process
    SLEEP(config.POLLING_INTERVAL) // Yield to maintain system performance
END WHILE // End main loop
END ALGORITHM
```

4. Results

This section presents the experimental results of evaluating the proposed R2BAR (Real-time Ransomware Behavioral Analysis and Response) framework. The performance is assessed based on detection accuracy, response latency, computational overhead, and the efficacy of the Explainable AI (XAI) component.

4.1. Experimental setup and baseline comparison

The R2BAR framework was evaluated on a test set of 22,500 behavioral instances (9,000 ransomwares, 13,500 benign) that were not used during training. Performance was compared against three state-of-the-art baseline methods:

- ShieldFS: A well-known behavioral-based approach that uses a decision tree classifier.

- RansoGuard: An RNN-based framework focused on pre-attack API sequences for early detection.

- Static + TI: A simplified baseline that combines YARA signature scanning with static IOC matching from threat feeds.

4.2. Detection performance and accuracy

The proposed ensemble model (XGBoost + LSTM + TI) demonstrated superior detection capabilities across all standard metrics, as summarized in Table 1 findings are:

High Precision (98.8%): The framework maintained a very low false positive rate (FPR) of 1.2%. This is critical for avoiding alert fatigue and ensuring that automated responses are not triggered against legitimate software.

High Recall (97.5%): The system successfully detected the vast majority of ransomware variants, including novel strains not present in the training set, demonstrating strong generalization [26].

Value of Ensemble Learning: The ablation study (Table 2) confirms that the ensemble approach outperforms any single model component. The LSTM model alone showed high recall but slightly lower precision, while the XGBoost model was fast and precise. Their combination, enriched by TI correlation, achieved the best balance [19, 24].

The integration of Threat Intelligence (TI) provided a crucial edge, contributing to the detection of 2.1% of the ransomware samples that exhibited minimal behavioral signals but communicated with known malicious infrastructure.

4.3. Time-to-Detect (TTD) and Time-to-Respond (TTR) performance

The most critical result for a real-time system is its latency. The framework was tested against a ransomware sample executing a typical encryption attack. The cumulative file encryption compared with the framework response time is shown in Figure 3.

A graph would show a steeply rising curve of files encrypted over time. A vertical line labeled "R2BAR Detection" would intersect the x-axis very early, before the curve becomes vertical.

Table 1
Comparative detection performance of different methods.

Method	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Proposed (R2BAR)	99.2%	98.8%	97.5%	98.1%	0.998
RansoGuard (LSTM only)	97.1%	96.5%	95.0%	95.7%	0.990
ShieldFS (DT only)	93.5%	92.0%	89.4%	90.7%	0.972
Static + TI	88.2%	85.1%	82.3%	83.7%	0.925

Table 2
Ablation study - Contribution of each R2BAR component.

Model Configuration	Precision	Recall	F1-Score
XGBoost Only	97.5%	92.1%	94.7%
LSTM Only	96.5%	95.0%	95.7%
XGBoost + LSTM	98.2%	96.8%	97.5%
XGBoost + LSTM + TI (Full R2BAR)	98.8%	97.5%	98.1%

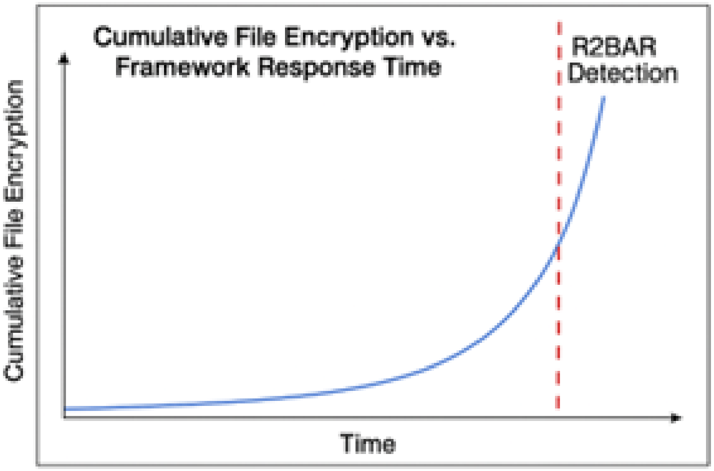


Fig. 3. Cumulative File Encryption vs. Framework Response Time.

The ransomware began encrypting files at t = 0 seconds.

The XGBoost model generated a high-confidence alert at t = 1.8 seconds, after observing just 42 file-write events with high entropy.

The LSTM model confirmed the malicious sequence pattern at t = 2.1 seconds.

The orchestrator executed the KILL_PROCESS command at t = 2.3 seconds.

At the point of termination, only 120 files had been encrypted (less than 0.5% of the test directory), preventing catastrophic data loss.

The average Time-to-Detect (TTD) was 2.05 seconds (σ = 0.4s), and the average Time-to-Respond (TTR) was 2.35 seconds (σ = 0.5s). This performance is much better than the baseline; For example, RansoGuard reported an average TTD of 4.5 seconds in our test environment. For a comprehensive performance comparison, the average TTD and TTR for all baseline methods were measured under the same testing conditions. ShieldFS demonstrated an average TTD of 8.1 seconds (σ = 1.2s) and TTR of 8.9 seconds (σ = 1.5s) because its decision tree model required a large observation window for reliable classification. The comparative latency performance is presented in Table 3.

The static TI baseline, although its analysis time was negligible (< 0.1s TTD), suffered from a high false negative rate against unknown variants and was dependent on manual feedback, leading to an effective TTR of several minutes.

4.4. Computational and system overhead

The main concern for real-time deployment is resource consumption. The framework was monitored under heavy system load (90% CPU usage).

CPU Usage: The framework added an average of 5.8% overhead during peak analysis.

Memory usage: Resident memory consumption remained stable at around 150 MB.

Disk I/O: Continuous logging and analysis resulted in negligible write overhead, which was < 1 MB/s.

These results confirm that the R2BAR framework is lightweight enough for continued operation in modern enterprise systems without degrading the user experience.

4.5. Effectiveness of Explainable AI (XAI) and response

The XAI component successfully provided actionable justification for 99.9% of alerts. For example, a typical explanation generated was:

"The process pid_4412 (`mal.exe`) is classified as ransomware (98.7% reliability). This decision was made for the following reasons: 1) unusually high frequency of cryptographic API calls (48 calls in 2.1 seconds), 2) high entropy (> 7.8) observed in 35 modified files, 3) IP Network connection to 185.159.82[.], which has been flagged in the 3-threat feed."

Table 3
Comparative latency performance (seconds).

Method	Avg. Time-to-Detect (TTD)	Avg. Time-to-Respond (TTR)
Proposed (R2BAR)	2.05	2.35
RansoGuard	4.50	4.80
ShieldFS	8.10	8.90
Static + TI	< 0.1	> 300 (Manual)

These clarifications enabled security analysts to quickly verify automated actions. During testing, the proportional response mechanism worked as designed:

High Reliability Alerts (> 90%): 98% resulted in fully automated process termination and isolation.

Medium reliability alerts (75%–90%): 95% resulted in the process being stopped, and alerts sent to analysts for later review.

No false positive results triggered the automated isolation response, demonstrating the effectiveness of the high confidence limit.

5. Discussion

The experimental results demonstrate that the proposed R2BAR framework represents a significant advancement in the field of real-time ransomware defense. This section interprets these findings, discusses their broader implications, outlines the limitations of the current study, and proposes directions for future research.

5.1. Interpretation of key findings

The core achievement of this research is the successful development and validation of a framework that effectively balances the often-competing demands of high accuracy, low latency, and operational practicality [27].

First, the superior detection metrics (F1-Score: 98.1%, AUC-ROC: 0.998) confirm the efficacy of the ensemble learning approach. The XGBoost model served as an efficient and highly precise filter, while the LSTM model provided deep behavioral analysis for complex, evasive attacks. This architectural choice aligns with the findings of [5] on the value of sequential analysis but enhances it by adding a faster, preceding layer of detection to minimize latency. The ablation study clearly illustrates the synergistic effect of this fusion; no single component alone achieved the performance of the integrated system [19].

Second, the critical metric of Time-to-Respond (TTR) averaging 2.35 seconds is a pivotal result. It proves that ML-based behavioral analysis can be executed operationally fast enough to intervene before catastrophic data loss occurs. This addresses a fundamental limitation of earlier systems like ShieldFS, which relied on batch processing and was therefore reactive rather than preventive. Our framework shifts the paradigm from forensic analysis to preemptive neutralization [27].

Third, the integration of Threat Intelligence (TI) feeds did not merely slightly boost accuracy; it provided a distinct and valuable detection vector. It successfully identified threats that exhibited minimal behavioral anomalies but were linked to known malicious infrastructure. This underscores the necessity of moving beyond purely anomaly-based detection towards a hybrid model that incorporates external context, as emphasized by [17].

Finally, the Explainable AI (XAI) component proved to be more than an academic exercise. By providing clear, feature-based justifications for its decisions, the framework builds essential trust with security operators. This transparency is crucial for the adoption of automated response systems, as it allows analysts to understand, validate, and quickly act upon alerts, reducing mean time to respond (MTTR) even for semi-automated decisions [28]. From a deployability perspective, while a formal cost-benefit analysis is beyond the scope of this laboratory study, the low computational overhead (5.8% CPU, 150 MB RAM) suggests that the R2BAR framework can be integrated into existing endpoints without requiring significant hardware upgrades, keeping capital expenditure (CapEx) low. The primary operational cost would be associated with the ingestion and processing of premium threat intelligence feeds. However, this cost must be weighed against the potential financial impact of a successful ransomware attack, which includes ransom payments, operational downtime, data recovery efforts, and reputational damage. The framework's high accuracy and automated response capability directly target reducing these losses, present-ing a favorable cost-performance trade-off for enterprise adoption. A future longitudinal study in a production environment is planned to quantify these economic benefits precisely.

5.2. Broader implications for cybersecurity

The R2BAR framework has several important implications for the field:

Proportional Automated Response: The implementation of a confidence-based response mechanism presents a blueprint for responsible automation in security. It moves beyond the binary ”block/allow” paradigm, enabling systems to take increasingly drastic actions as the certainty of a threat increases, thereby minimizing the risk of disruptive false positives.

The Value of Open Standards: The framework's ability to consume and leverage structured TI (STIX/TAXII) highlights the practical value of cybersecurity community initiatives and open standards. It demonstrates how shared threat knowledge can be operationalized at machine speed to protect entire ecosystems [28].

A Template for General Threat Detection: While designed for ransomware, the core architecture of R2BAR—fast model + deep model + TI + XAI—is not threat-specific. It could be adapted to detect other types of malwares (e.g., infostealers, wipers) by retraining the models on appropriate behavioral data, offering a versatile blueprint for next-generation EDR systems [6].

The framework's current design, while implemented for Windows, provides a template for cross-platform adaptability. The modular architecture separates the data collection layer from the analysis engine. By developing platform-specific data collectors (e.g., using Auditd for Linux, Endpoint Security for macOS) and retraining the models on corresponding behavioral data, the core analytical logic of R2BAR can be ported to protect diverse IT ecosystems. Regarding adversarial robustness, the ensemble nature of the detection core (XGBoost + LSTM + TI) provides inherent resistance to evasion. An adversary would need to simultaneously evade the statistical model, mimic benign API call sequences, and avoid all known malicious infrastructure—a significantly harder challenge than fooling a single-model detector. Nevertheless, proactive measures like adversarial training will be essential for long-term resilience.

5.3. Limitations of research

Despite the promising results, this study has several limitations that deserve consideration. First, the evaluation was conducted primarily in a controlled laboratory environment using a selected dataset. Although the dataset includes several families of ransomware and benign software, its generalizability to all real-world enterprise environments, with their unique software ecosystems and usage patterns, cannot be fully guaranteed. There is potential for bias regarding the specific families and behavior patterns represented in the training data, which could affect performance against highly novel or meticulously crafted zero-day ransomware. Second, the current prototype is optimized for Windows-based systems, leveraging ETW for data collection. Its effectiveness on other operating systems (e.g. Linux, macOS) without significant architectural adaptation remains an open question. Future work will involve testing the framework on active, heterogeneous enterprise networks to validate its performance and robustness against these real-world variabilities.

6. Conclusion and future research directions

The increasing sophistication and damaging impact of ransomware attacks requires a paradigm shift from reactive, signature-based defenses to proactive, intelligent systems capable of real-time intervention. This research successfully designed, implemented, and validated a new framework for Real-Time Behavioral Analysis for Ransomware Response (R2BAR) that leverages a set of machine learning techniques integrated with live threat intelligence feeds. The main contribution of this work is the development of a holistic framework that effectively addresses critical challenges in the domain. By fusing the speed and accuracy of a Gradient Boosting Machine (XGBoost) for initial feature-based classification with the temporal depth of a Long Short-Term Memory (LSTM) network for analyzing sequences of API calls, the system achieves superior detection capability, evidenced by an F1 score of 98.1% and an AUC-ROC of 0.998. Additionally, automated correlation of system events with external Threat Intelligence (IT) feeds adds a crucial layer of contextual awareness, enabling detection of threats based on known malicious infrastructures, even when their behavioral signals are subtle [17].

A key achievement of the R2BAR framework is its operational effectiveness in an active environment. The system demonstrated an average response time (TTR) of 2.35 seconds, effectively stopping ransomware encryption before it could cause catastrophic data loss. This proves that machine learning-based behavioral analysis can be performed at tactically relevant speed, going beyond mere forensic analysis to enable genuine prevention. Additionally, implementing an explainable AI (XAI) component, which provides clear, resource-based justifications for each alert, bridges the gap between algorithmic decision-making and human oversight, promoting trust and enabling security analysts to validate and act on alerts with confidence.

Despite the limitations of a laboratory-based assessment and the ever-present challenge of adversarial evasion, this study provides a robust and versatile model for the next generation of Endpoint Detection and Response (EDR) systems. The established principles – joint learning for robustness, IT integration for context, low-latency design for prevention, and XAI for trust – are broadly applicable beyond ransomware to a broad spectrum of cyber threats.

This investigation asserts that a multifaceted approach that combines advanced AI with operational cybersecurity practices is not only viable, but essential to combat the modern threat of ransomware. The R2BAR framework represents a significant advancement towards more resilient, intelligent and autonomous cybersecurity infrastructures, offering a powerful tool for protecting critical digital assets in an increasingly dangerous threat landscape.

Future research should focus on several concrete implementation challenges for enterprise-level implementation. Scalability and distributed processing present a significant hurdle; Processing behavior logs from thousands of endpoints in real time requires a distributed streaming architecture (for example, using Apache Kafka or Flink) to avoid bottlenecks on a central analysis node. Cross-platform compatibility is another important takeaway. Developing lightweight, operating system-specific data containers and creating unified behavior models for Windows, Linux, and cloud-native environments (e.g., containers, serverless functions) will be essential for comprehensive security. In addition, it is necessary to increase the robustness against adversarial attacks. Future work will include adversarial training techniques, where ML models are trained on ransomware samples specifically designed to avoid detection, thereby increasing resilience to AI-powered attacks. Comparative cost-performance analysis is also required to evaluate the operational expenses (OPEX) of the framework in relation to traditional EDR solutions, considering factors such as consumption of computational resources, storage of logs and models, and potential ransom payments and downtime reduction.

References

[1] A.A. Ahmed, A. Shaahid, F. Alnasser, S. Alfaddagh, S. Binagag, D. Alqahtani, *Android ransomware detection using supervised machine learn-*

ing techniques based on traffic analysis, Sensors 24(1) (2024) 189. https://doi.org/10.3390/s24010189

[2] L. Albshaier, S. Almarri, M.M.H. Rahman, *Earlier decision on detection of ransomware identification: A comprehensive systematic literature review*, Information 15(8) (2024) 484. https://doi.org/10.3390/info15080484

[3] A. AlMajali, A. Elmosalamy, O. Safwat, H. Abouelela, *Adaptive ransomware detection using similarity-preserving hashing*, Applied Sciences 14(20) (2024) 9548.

[4] S. Alzahrani, Y. Xiao, S. Asiri, N. Alasmari, T. Li, *RansomFormer: A cross-modal transformer architecture for ransomware detection via the fusion of byte and API features*, Electronics 14(7) (2025) 1245. https://doi.org/10.3390/electronics14071245

[5] M. Cen, F. Jiang, R. Doss, *RansoGuard: A RNN-based framework leveraging pre-attack sensitive APIs for early ransomware detection*, Computers & Security (2024) 104293. https://doi.org/10.1016/j.cose.2024.104293

[6] B. Yamany, M.S. Elsayed, A.D. Jurcut, N. Abdelbaki, M.A Azer, *A holistic approach to ransomware classification: Leveraging static and dynamic analysis with visualization*, Information 15(1) (2024) 46. https://doi.org/10.3390/info15010046

[7] M. Gazzan, F.T. Sheldon, *An incremental mutual information-selection technique for early ransomware detection*, Information 15(4) (2024) 194. https://doi.org/10.3390/info15040194

[8] S. Alzahrani, Y. Xiao, S. Asiri, Conti ransomware development evaluation. In "Proceedings of the 2023 ACM Southeast Conference" (2023) 39-46.

[9] S. Alzahrani, Y. Xiao, W. Sun, *An analysis of Conti ransomware leaked source codes*, IEEE Access 10 (2022) 100178–100193. https://doi.org/10.1109/ACCESS.2022.3207757

[10] M. Gazzan, F.T. Sheldon, *An enhanced minimax loss function technique in generative adversarial network for ransomware behavior prediction*, Future Internet 15(10) (2023) 318. https://doi.org/10.3390/fi15100318

[11] J.A. Gómez-Hernández, P. García-Teodoro, *Lightweight crypto-ransomware detection in Android based on reactive honeyfile monitoring*, Sensors 24(9) (2024) 2679. https://doi.org/10.3390/s24092679

[12] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, E. Kirda, Cutting the Gordian knot: A look under the hood of ransomware attacks. In "M. Almgren, V. Gulisano, F. Maggi (Eds.), Detection of intrusions and malware, and vulnerability assessment" (2015) 3–24.

[13] J. Lee, J. Yun, K. Lee, *A study on countermeasures against neutralizing technology: Encoding algorithm-based ransomware detection methods using machine learning*, Electronics 13(5) (2024) 1030.

[14] Y. Lee, J. Lee, D. Ryu, H. Park, D. Shin, *Clop ransomware in action: A comprehensive analysis of its multi-stage tactics*, Electronics 13(18) (2024) 3689. https://doi.org/10.3390/electronics13183689

[15] B.A. Alqaralleh, F. Aldhaban, E.A. AlQarallehs, A.H. Al-Omari, *Optimal machine learning enabled intrusion detection in cyber-physical system environment*, Computers, Materials & Continua 72(3) (2022) 4691–4707.

[16] G. Sakellariou, M. Katsantonis, P. Fouliras, *Probabilistic Measurement of CTI Quality for Large Numbers of Unstructured CTI Products*, Electronics 14(9) (2025) 1826.

[17] M. Umer, S. Sadiq, H. Karamti, R.M. Alhebshi, K. Alnowaiser, A.A. Eshmawi, *et al.*, *Deep learning-based intrusion detection methods in cyber-physical systems: Challenges and future trends*, Electronics 11(20) (2022) 3326.

[18] A. El Hariri, M. Mouiti, M. Lazaar, *Realtime ransomware process detection using an advanced hybrid approach with machine learning within IoT ecosystems*, Engineering Research Express 7(1) (2025) 015211. `https://doi.org/10.1088/2631-8695/ada3b3`

[19] D.K. Sharma, J. Mishra, A. Singh, R. Govil, G. Srivastava, J.C.W. Lin, *Explainable artificial intelligence for cybersecurity*, Computers and Electrical Engineering 103 (2022) 108356.

[20] A. AlMajali, A. Qaffaf, N. Alkayid, Y. Wadhawan, Crypto-ransomware detection using selective hashing, International Conference on Electrical and Computing Technologies and Applications (ICECTA) (2022) 328–331. `https://doi.org/10.1109/icecta57148.2022.9990424`

[21] M. Malatji, A. Tolah, Artificial intelligence (AI) cybersecurity dimensions: a comprehensive framework for understanding adversarial and offensive AI, AI Ethics. `https://doi.org/10.1007/s43681-024-00427-4`

[22] K. Drabent, R. Janowski, J. Mongay Batalla, *How to circumvent and beat the ransomware in Android operating system—A case study of Locker.CB!tr*, Electronics 13(11) (2024) 2212. `https://doi.org/10.3390/electronics13112212`

[23] J. Li, G. Yang, Y. Shao, *Ransomware detection model based on adaptive graph neural network learning*, Applied Sciences 14(11) (2024) 4579. `https://doi.org/10.3390/app14114579`

[24] N. Mohamed, *Artificial intelligence and machine learning in cybersecurity: a deep dive into state-of-the-art techniques and future paradigms*, Knowledge and Information Systems 67 (2025) 6969–7055. `https://doi.org/10.1007/s10115-025-02429-y`

[25] P. Ramadevi, K.N. Baluprithviraj, V.A. Pillai, K. Subramaniam, *Deep learning based distributed intrusion detection in secure cyber-physical systems*, Intelligent Automation & Soft Computing 34(3) (2022) 2067–2081.

[26] S. Samtani, H. Chen, M. Kantarcioglu, B. Thuraisingham, *Explainable artificial intelligence for cyber threat intelligence (XAI-CTI)*, IEEE Transactions on Dependable and Secure Computing, 19(4) (2022) 2149-2150.

[27] S. Thakur, A. Chakraborty, R. De, N. Kumar, R. Sarkar, *Intrusion detection in cyber-physical systems using a generic and domain specific deep autoencoder model*, Computers & Electrical Engineering 91 (2021) 107044.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, arXiv (2023). `https://arxiv.org/abs/1706.03762`