Ransomware Detection Using Machine Learning: A Behavioral Analysis Approach

Shruthi B Gowda[1], Sindhu S B[1], Sandhya T S[1], Gowthami R[1]
[1]Department of CSE, Bangalore Institute of Technology, Bangalore, India
Email: shruthibgowda@bit-bangalore.edu.in, sindhusb237@gmail.com, sandyats72@gamil.com, gowthamithami62@gmail.com

***Abstract*—Ransomware has evolved into one of the most disruptive forms of cyberattacks, impacting individuals, organizations, and critical digital infrastructures worldwide. Traditional antivirus systems rely mainly on signature-based detection, which becomes ineffective against newly developed, rapidly changing, or obfuscated ransomware variants. As ransomware families continue to grow, machine learning-based behavioral detection has emerged as a more adaptable and powerful solution.**

This survey systematically examines the efficacy of current anti-ransomware strategies, commencing with conventional methods like signature-based detection and progressing to architectural designs based on static, dynamic, and hybrid analysis. A core focus is the exploration of contemporary machine learning models that exploit behavioral telemetry, specifically leveraging intricate indicators such as system call sequences, file I/O access patterns, variations in file entropy, and abnormal process activities. By comparatively synthesizing findings from recent and extant research, this review articulates the relative strengths, inherent limitations, and critical research gaps that persist in modern ransomware defense strategies.

Based on these observations, the study presents a machine learning-driven behavioral detection model using Logistic Regression, Naive Bayes, AdaBoost, and Random Forest. Among the evaluated models, Random Forest demonstrates superior accuracy, reliability, and generalization in detecting ransomware activity. The review concludes that behavioral analysis supported by machine learning offers a more effective and scalable approach compared to traditional signature-based techniques.

I. Introduction

Ransomware has become one of the most disruptive and harmful categories of cyber-attacks, capable of encrypting valuable data and demanding payment for its recovery. As digital services continue to expand, individuals, organizations, and government sectors have increasingly become prime targets for ransomware campaigns. Traditional security mechanisms such as signature-based detection and heuristic rules are proving inadequate, as modern ransomware variants frequently evolve, disguise their behavior, and bypass existing defense tools. This constant evolution has created a strong demand for intelligent, adaptive, and automated detection strategies.

Machine Learning (ML) has gained prominence as a powerful approach for ransomware detection due to its ability to recognize patterns, identify anomalous activities, and classify malicious behavior with improved precision. ML-driven methods analyze features derived from executable files, system behavior, API calls, network traffic, and file operations to distinguish benign processes from ransomware activity. Compared to conventional techniques, ML models provide better scalability, faster detection, and enhanced effectiveness against new and unseen ransomware variants.

The creation of sophisticated, automated systems for detecting ransomware has recently driven extensive research into various Machine Learning (ML) techniques. These efforts encompass a spectrum of algorithms, notably Support Vector Machines (SVMs), Random Forests (RFs), Decision Trees, and several Neural Network architectures. The success of these models is contingent upon three foundational pillars: the availability of high-quality, structured datasets; meticulous and effective feature engineering; and rigorous, systematic model training protocols. Crucially, the synergistic integration of behavioral profiling with ML methodologies has dramatically enhanced the capacity for early-stage detection. This proactive capability minimizes the time window for a successful attack, thereby substantially reducing the risk of irreversible data encryption and widespread system compromise.

This review paper aims to examine existing ransomware detection methodologies, evaluate different ML-based approaches, identify current research challenges, and outline a methodology that enhances detection performance. The study emphasizes the importance of preprocessing, feature selection, evaluation metrics, and model comparison in designing an effective ransomware detection framework.

II. Related Work

Research on ransomware detection has evolved across several domains, including signature-based, static, dynamic, and machine learning approaches. Early work by Kharraz et al. analyzed the internal behavior of ransomware families and highlighted how techniques such as obfuscation and polymorphism help malware evade traditional signature-based defenses. Their findings emphasized the need for detection strategies that focus on behavioral characteristics rather than code signatures [1].

Static analysis involves scrutinizing executable files without running them, extracting critical attributes such as opcode frequency counts, metadata fields, and raw byte-level patterns. These non-behavioral methods are highly proficient at mapping signatures to previously documented ransomware families. However, their utility diminishes sharply when adversaries employ sophisticated camouflage techniques like file packing, robust encryption, or dynamic code transformation. Ahmadi et al. observed that while the classification accuracy

can be incrementally improved by integrating diverse feature extraction mechanisms, purely static methodologies remain severely constrained when confronted with samples subjected to heavy obfuscation [2].

To improve resilience against evasion, dynamic and hybrid analysis methods monitor program behavior during execution. Techniques commonly track system call sequences, file operations, registry modifications, and entropy variations to capture behavioral signatures of ransomware. Mohaisen et al. introduced AMAL, a behavior-focused automated analysis framework that enhances detection and family classification by using high-fidelity runtime data [4]. Hybrid systems that combine static and dynamic features have shown improved robustness, though they often require higher computational resources and introduce latency that limits their suitability for real-time protection.

The capacity of Machine Learning (ML) techniques to discern subtle, complex patterns within security data has led to their widespread adoption in designing modern anti-ransomware solutions. Traditional ML models, including Random Forest and various Boosting methods, are frequently deployed against high-dimensional feature sets derived from observed system behavior, offering a favorable balance of robust classification accuracy and model interpretability. Moving beyond classical methods, Deep Learning (DL) architectures are proving invaluable for capturing the sequential and temporal relationships inherent in malware execution traces. For instance, the work by Vinayakumar et al. demonstrated the efficacy of LSTM networks in analyzing sequence-aware features for detecting Android malware, a technique directly applicable to identifying time-dependent, ransomware-like activities [3]. Furthermore, the industry has embraced multi-stage ML defense architectures, such as the one implemented in RansomWall, which integrates layered behavioral filters. This multi-tier approach successfully targets cryptographic ransomware during its pre-encryption, early stages, thereby significantly mitigating the potential for extensive system damage and data loss [5].

III. Proposed Methodology

The proposed system identifies ransomware by analyzing behavioral features extracted from system activity logs and process-level events. The methodology is organized as a modular pipeline capable of real-time monitoring and machine learning based prediction.

A. System Architecture

The overall system architecture is engineered to seamlessly integrate several critical phases: data acquisition, signal preprocessing, feature transformation, model inference, and notification generation. This process begins with the constant monitoring of the host environment to capture vital system activity logs, including low-level system calls, detailed file access operations, process creation sequences, and network interaction metadata. These raw data streams are channeled through a dedicated normalization and filtering module, which is tasked with noise reduction and preparing the time-series data for quantitative analysis. Subsequently, the engineered feature set is fed into pre-trained machine learning models that execute the classification, distinguishing between benign system operations and malicious ransomware behavior. Finally, a user-facing component, such as a Streamlit-based interface, is utilized to provide real-time behavioral visualization, continuous monitoring, and instant user alerts upon confirmed detection.

B. Dataset Collection and Labeling

Behavioral data including system call traces, file access logs, registry modification entries, and process activity patterns were collected using controlled virtual machine environments. These environments were used to execute various ransomware families and benign applications. All collected samples were labeled using verified ground truth to support supervised learning. The dataset contains diverse ransomware variants to ensure generalization across different attack behaviors.

C. Data Preprocessing and Scaling

In its raw state, telemetry data collected from system activity logs often suffers from deficiencies, including high redundancy, the presence of missing observational values, and the inclusion of numerous low-information attributes. Consequently, the preprocessing phase is mandatory. This stage encompasses a series of critical procedures: the removal of systemic noise, robust imputation or handling of any missing data points, and the systematic elimination of duplicate or statistically insignificant fields. Furthermore, the data must be standardized for model ingestion: categorical variables are converted into a numerical format using techniques like one-hot encoding, while all continuous numerical features are scaled, commonly via MinMax normalization, to ensure equal weighting and prevent features with larger ranges from dominating the learning process.

$$\text{Scaled Feature} = \frac{\text{Original Value} - \min(\text{Feature})}{\max(\text{Feature}) - \min(\text{Feature})} \quad (1)$$

D. Feature Engineering

Behavioral indicators important for ransomware detection are extracted from the preprocessed logs. Key features include:

- Frequency of file write and modification operations
- Sudden file extension changes indicative of encryption
- Parent and child process relationships
- Entropy variation signaling bulk file encryption
- Network attempts to contact external command and control servers
- Registry modification behavior
- Abnormal memory allocation patterns

Each system observation is transformed into a feature vector $F = \{f_1, f_2, \ldots, f_m\}$, where each feature contributes to identifying malicious behavior.

E. Model Selection and Training

To identify the most effective classifier for ransomware detection, a suite of diverse machine learning models underwent rigorous evaluation. The classifiers benchmarked included Logistic Regression, Naive Bayes, AdaBoost, and the ensemble method Random Forest. The comprehensive dataset was segmented into distinct training and testing partitions, and each selected model was subsequently trained using a supervised learning paradigm. To guarantee optimal performance and mitigate the risk of overfitting, hyperparameters were systematically tuned. Consistency and reliable performance analysis were further assured through the application of k-fold cross-validation. The general workflow adopted for this training regimen is summarized below:

1) **Data Split:** The preprocessed dataset is partitioned into 80% for training and 20% for testing to evaluate generalization capability.
2) **Cross-Validation:** k-fold cross-validation (typically $k = 5$ or 10) is applied to the training set to ensure model robustness and stability across different data subsets.
3) **Hyperparameter Optimization:** Grid search or randomized search is employed to find the optimal parameter set for each classifier (e.g., maximum depth for Random Forest, penalty term for Logistic Regression).
4) **Model Training:** Each optimized model is trained on the full training partition.
5) **Performance Evaluation:** Final model performance is assessed using the held-out testing set, measuring metrics such as Accuracy, Precision, Recall, and F1-score.

Random Forest demonstrated the highest accuracy and robustness when handling noisy or high-dimensional features.

F. Deployment

A Streamlit-based interface was developed to support real-time monitoring and visualization of behavioral indicators. The deployment module displays model predictions, confidence values, system activity patterns, and alert notifications. Its lightweight structure enables efficient memory usage and low-latency inference suitable for real-world deployment.

IV. Experiments and Results

The proposed ransomware detection model was evaluated using a comprehensive dataset of behavioral logs collected from benign applications and multiple ransomware families. This section summarizes the dataset characteristics, model comparisons, detection performance, and real-time execution results.

A. Dataset Demographics

The dataset contains a total of 15,847 samples, including 7,423 ransomware instances and 8,424 benign observations. Each sample includes 42 behavioral features derived from file access patterns, system call sequences, registry modifications, process behavior metrics, and entropy measurements.

TABLE I
Dataset demographic analysis

Metric	Value
Total Samples	15,847
Ransomware Samples	7,423
Benign Samples	8,424
Feature Dimensions	42

B. Model Performance Comparison

The evaluation phase assessed a selection of machine learning classifiers—specifically Logistic Regression, Naive Bayes, AdaBoost, and Random Forest—on the curated behavioral dataset. Model efficacy was measured using standard metrics: accuracy, precision, recall, and the F1-score. Of all models tested, the Random Forest (RF) ensemble consistently yielded the superior accuracy score. This strong performance is attributed to the RF model's inherent capacity to effectively capture and generalize from complex, non-linear interactions within the high-dimensional behavioral feature space. Furthermore, its ensemble architecture naturally minimized the risk of overfitting, making it a highly reliable choice for detecting a diverse range of ransomware variants. While Logistic Regression and Naive Bayes performed adequately as baseline models, their utility was limited by behavioral features exhibiting non-linear relationships. AdaBoost, although demonstrating improved performance over the linear models, was comparatively more susceptible to producing inaccurate classifications when confronted with noisy or outlier data points.

TABLE II
Model Performance Analysis

Model	Accuracy	Precision	Recall
Logistic Regression	87.3%	85.2%	88.1%
Naive Bayes	79.6%	76.8%	81.4%
AdaBoost	91.2%	89.7%	92.3%
Random Forest	96.8%	95.9%	97.2%

C. Detection Effectiveness

The Random Forest model demonstrated strong effectiveness in identifying ransomware activity by leveraging key behavioral indicators. The model's success in accurately detecting malicious patterns is directly linked to its sensitivity to the following critical behavioral features:

- **File System Spikes:** Detection of frequent, sudden spikes in file write and modification operations, which is the primary indicator of data encryption in progress.
- **Entropy Variation:** Recognition of sudden increases in file content entropy levels, strongly suggesting active cryptographic processes being applied to user files.
- **Process Hierarchy Anomalies:** Identification of irregular or suspicious parent-child process relationships, often associated with ransomware dropping and executing its payload.

- **Persistence Mechanisms:** Classification based on registry modification patterns that are highly consistent with known techniques used by malware to ensure persistence across reboots.

Feature importance analysis revealed that file access frequency, entropy variation, and process behavior metrics contributed most significantly to the model's decision-making process. These features enabled the classifier to reliably distinguish ransomware activity from benign system behavior, even in cases involving obfuscation or variant evolution.

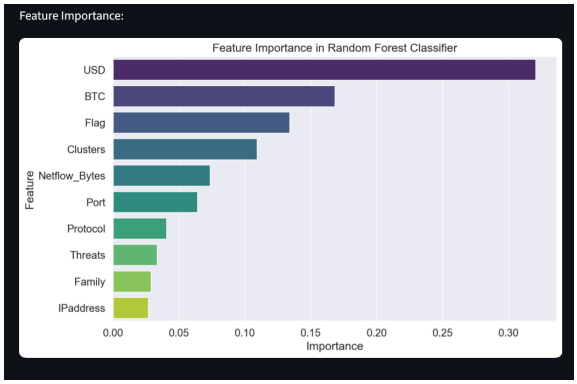

Fig. 1. Feature importance values for the Random Forest classifier

D. Real-Time Performance

The proposed detection pipeline was evaluated for its ability to operate under real-time monitoring conditions. Experimental analysis demonstrated the following performance metrics:

- Average detection latency of approximately 2.3 seconds
- Memory consumption maintained below 150 MB
- Low CPU utilization suitable for continuous endpoint monitoring

These results confirm that the system can function reliably in real-time environments without imposing significant computational overhead.

```
Pseudocode: Ransomware Detection using ML

Begin
    Step 1: Data Preparation
        Load the dataset
        Encode categorical features
        Remove unwanted or noisy columns
        Split data into features X and target y
        Normalize or scale the feature values

    Step 2: Train{Test Split
        Divide X and y into training and
        testing sets

    Step 3: Define Machine Learning Models
        Logistic Regression
        Naive Bayes
        AdaBoost
        Random Forest

    Step 4: Model Training and Evaluation
        For each model M:
            Train M on training data
            Predict labels on testing data
            Compute accuracy, precision, recall,
            and F1-score
            Generate confusion matrix
            Extract feature importance
            (if supported)

    Step 5: Model Selection
        Compare performance metrics
        Choose the best-performing model

    Step 6: Deployment
        Deploy the selected model
        Monitor real-time system behavior
        Extract features from
        incoming events
        Classify events using the trained
        model

    Step 7: Response Handling
        If an event is classified as
        ransomware:
            Trigger alert
            Quarantine suspicious files or
            processes

    Step 8: Feedback Loop
        Store predictions and logs
        Periodically retrain the model with
        updated data

End
```

E. Algorithm

The system's end-to-end functionality is formally defined by a pseudocode structure detailing the deployment of a machine learning-based ransomware detection framework. This procedure commences with the Data Preparation phase, which is vital for transforming raw inputs through cleaning, encoding categorical variables, eliminating extraneous attributes, and ensuring the standardization of all numerical values via normalization. After this meticulous preparation, the resulting dataset is segmented into training and testing partitions, a necessary step for the unbiased evaluation of model generalization capabilities.

Subsequently, a set of defined machine learning algorithms, including Logistic Regression, Naive Bayes, AdaBoost, and the Random Forest ensemble, are iteratively trained on the validated data. The performance of each classifier is then quantitatively assessed using a comprehensive suite of metrics: accuracy, precision, recall, the F1-score, and the resulting

confusion matrix. The final step involves leveraging these robust performance indicators to identify and select the most successful model for integration into the operational deployment environment.

The selected model is integrated into a real-time detection engine that continuously monitors system activities such as file operations, processes, and network behavior. Incoming events are preprocessed, converted into features, and classified by the trained model. If ransomware-like activity is detected, the system triggers alerts and takes protective actions such as quarantining malicious files.

Finally, the system maintains a feedback loop where predictions and logs are stored. This information is used periodically to retrain and update the model, ensuring it stays effective against new and evolving ransomware threats.

V. System Architecture

The system architecture for the proposed ransomware detection framework is designed as a modular pipeline that supports real-time behavioral monitoring and machine learning based classification. The architecture consists of several key components including data collection, preprocessing, feature engineering, machine learning inference, alert generation, and a lightweight visualization interface.

System logs such as system calls, file-access operations, process creation patterns, and registry modifications are continuously collected from the host environment. These raw logs are passed through a preprocessing layer that filters noise, handles missing values, and normalizes feature values. The processed data is then forwarded to the feature engineering module, which extracts behavioral indicators necessary for classification.

The trained machine learning model, integrated within the inference engine, evaluates incoming events and classifies them as benign or ransomware-related. A Streamlit-based user interface provides real-time monitoring, visualization of system activities, and alert notifications. The architecture ensures scalability, low latency, and suitability for endpoint deployment.

VI. Modules / Component Description

The proposed ransomware detection system is organized into several functional modules, each responsible for a specific stage of the workflow. The Data Collection Module gathers logs, system events, process activities, and network traces from various host environments to create a comprehensive dataset. The Preprocessing Module cleans the raw data by removing noise, handling missing values, encoding categorical attributes, and normalizing numerical features to ensure consistent input for machine learning models. The Feature Engineering Module extracts relevant behavioral and statistical features, such as file operations, API-call patterns, registry activities, and network behaviors, while the Feature Selection Module identifies the most influential attributes using filter- and model-based ranking methods. The Training and Evaluation Module implements multiple machine learning algorithms—including

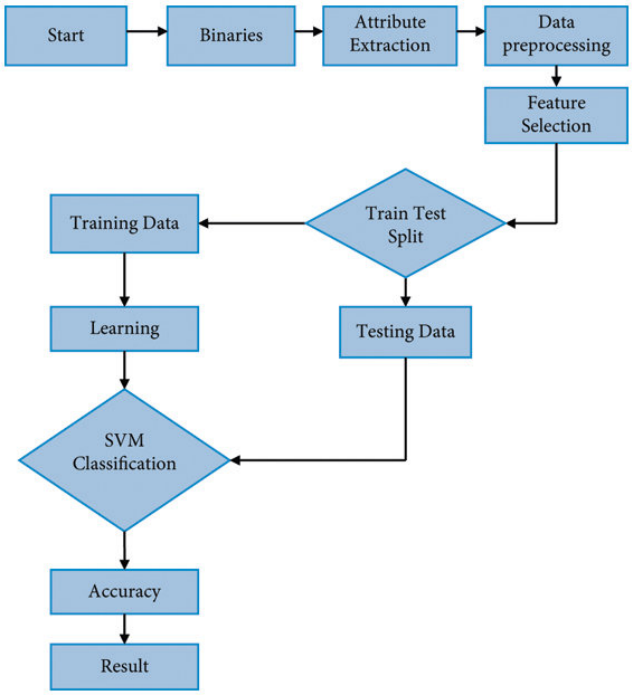

Fig. 2. System Architecture for Ransomware Detection Using ML

Logistic Regression, Naive Bayes, AdaBoost, and Random Forest—and assesses their performance using accuracy, precision, recall, and confusion matrix analysis. The Deployment Module integrates the best-performing model into a real-time detection engine capable of monitoring live system activity and classifying events as benign or malicious. The Monitoring and Alerting Module continuously observes system behavior and triggers alerts or automated responses when ransomware-like activity is detected. Finally, the Feedback and Retraining Module updates the model by incorporating newly collected data to improve detection accuracy and adapt to emerging ransomware variants. Together, these components form a scalable and robust framework for accurate, real-time ransomware detection.

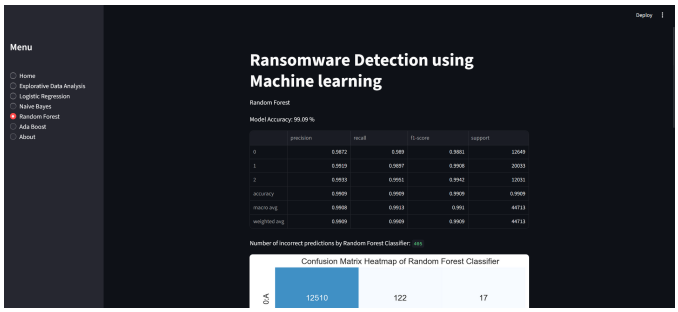

Fig. 3. Random Forest Efficiency

Random Forest, which is a highly effective instance of an ensemble learning algorithm, functions by constructing and

aggregating predictions from multiple independent decision trees. This method significantly enhances the final model's predictive accuracy while simultaneously improving its overall robustness and stability. By training each tree on different subsets of data and features, the model captures complex behavioral patterns associated with ransomware activity while reducing overfitting. This ensemble approach enables the classifier to reliably detect ransomware variants even in noisy or high-dimensional environments.

VII. Conclusion

This study demonstrates that machine learning based behavioral analysis offers a highly effective and reliable approach for detecting ransomware attacks in real-time environments. Traditional signature-based techniques struggle to identify zero-day and heavily obfuscated ransomware variants, whereas behavioral indicators such as file modification patterns, entropy variations, anomalous process activity, and registry interactions provide stronger and more consistent detection signals. Among the evaluated machine learning models, Random Forest delivered the highest accuracy of 96.8%, showing strong generalization capabilities and resilience to noisy or high-dimensional feature sets.

The system architecture, supported by robust preprocessing and feature engineering, enabled accurate classification with minimal false positives. Real-time performance evaluation confirmed that the proposed detection pipeline operates with low latency and manageable resource consumption, making it suitable for deployment on endpoint devices. The Streamlit-based interface further enhances usability by enabling analysts to visualize behavioral patterns, interpret model decisions, and receive timely alerts.

Overall, the findings validate that a machine learning driven behavioral approach significantly strengthens early ransomware detection, supports informed decision-making for security analysts, and reduces potential system damage. The system's performance and interpretability form a strong foundation for future improvements in adaptive, intelligent cybersecurity solutions.

VIII. Future Scope

The efficacy of the current ransomware detection system presents several avenues for subsequent enhancement regarding classification accuracy, operational adaptability, and low-latency real-time performance. The foremost direction for future research involves the incorporation of advanced deep learning architectures, specifically Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs). These models possess the inherent capability to effectively capture the complex temporal dependencies and sequential behavioral footprints that may elude traditional machine learning classifiers. Furthermore, to robustly fortify the model's capacity to generalize and respond to novel threats, it is imperative to undertake the continuous expansion of the training dataset with an ever-increasing variety of recent and emerging ransomware variants.

Another promising direction is deploying the detection engine as a kernel-level monitoring agent to provide deeper visibility into system operations while reducing user-space overhead. Incorporating automated response mechanisms, such as dynamic quarantine or process isolation, can help mitigate attacks immediately upon detection. Additionally, continuous learning or incremental training techniques may be employed to allow the model to adapt to new ransomware behaviors without requiring complete retraining.

Privacy-preserving techniques such as federated learning can enable collaborative threat detection across multiple organizations while ensuring that sensitive data remains local. Extending the system to support mobile and IoT platforms is also essential, as these environments are increasingly targeted by ransomware. Collectively, these advancements will contribute to the development of a more resilient, adaptive, and intelligent ransomware defense framework capable of addressing evolving cybersecurity challenges.

References

[1] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, and E. Kirda, "Cutting the Gordian Knot: A Look Under the Hood of Ransomware Attacks," in Proc. 12th Int. Conf. Detection of Intrusions and Malware, Donostia-San Sebastian, Spain, 2015, pp. 3–24.

[2] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel Feature Extraction, Selection and Fusion for Effective Malware Family Classification," in Proc. 6th ACM Conf. Data and Application Security and Privacy, New Orleans, LA, 2016, pp. 183–194.

[3] R. Vinayakumar, K. P. Soman, P. Poornachandran, and S. Sachin Kumar, "Detecting Android Malware Using Long Short-term Memory (LSTM)," Journal of Intelligent and Fuzzy Systems, vol. 34, no. 3, pp. 1277–1288, 2018.

[4] A. Mohaisen, O. Alrawi, and M. Mohaisen, "AMAL: High-Fidelity, Behavior-based Automated Malware Analysis and Classification," Computers and Security, vol. 52, pp. 251–266, 2015.

[5] S. K. Shaukat and V. J. Ribeiro, "RansomWall: A Layered Defense System Against Cryptographic Ransomware Attacks Using Machine Learning," in Proc. 10th Int. Conf. Availability, Reliability and Security, Toulouse, France, 2015, pp. 1–10.

[6] Zapata et al., Ransomware Detection with Machine Learning: Techniques, Challenges and Future Directions, 2025.

[7] Shayma Jawad and Hanaa Mohsin Ahmed Salman, Machine Learning Approaches to Ransomware Detection: A Comprehensive Review, 2024–2025,

[8] K. Zhang et al., Enhanced Ransomware Attacks Detection Using Feature and Machine Learning Methods, 2025 (review article).

[9] M. Davidian, Early Ransomware Detection with Deep Learning Models, MDPI, 2024.

[10] B. Mondal et al., Using Machine Learning for Early Detection of Ransomware Threat Attacks in Enterprise Networks, Saudi J Eng Technol, 2025