Ransomware Detection Using Machine Learning and Deep Learning Models: A Comparative Analysis

Tanishka Mali
College of IST
The Pennsylvania State University
State College, USA
tjm7431@psu.edu

Siddhi Lad
College of IST
The Pennsylvania State University
State College, USA
spl6114@psu.edu

Yash Parmar
College of IST
The Pennsylvania State University
State College, USA
yap5049@psu.edu

Aishwarya Gandhi
College of IST
The Pennsylvania State University
State College, USA
asg6021@psu.edu

Sohan Ramalingaiah
College of IST
The Pennsylvania State University
State College, USA
spr5998@psu.edu

Siddharth Mishra
College of IST
The Pennsylvania State University
State College, USA
spm7098@psu.edu

I. Introduction

Ransomware continues to evolve as a significant threat within the landscape of digital security, contributing to extensive financial and operational damage suffered by organizations worldwide. These attacks have spiked exponentially in recent years and the number of times a company has had to pay settlement fees has also increased. In a report published by Fortinet, ransomware attacks increased by 350% since 2018. However, despite a decrease in reported attacks in 2022 and 2023, settlement fees associated with them have grown larger as malicious actors continue to target big firms [1]. These alarming figures are majorly due to cyber-criminals and ransomware syndicates using advanced and sophisticated attack tactics that help them evade detections implemented by traditional defense strategies. Moreover, such attacks are being amplified with the integration of Artificial Intelligence (AI), allowing attackers to streamline their operations and create programs that adapt in real-time to bypass security measures. In December 2024, FunkSec, a ransomware group, built its ransomware using AI such that it was able to self-modify its behavioral patterns and change tactics in real-time by analyzing the target organization's security posture. It was able to successfully bypass all forms of conventional measures including firewalls and antivirus software. The group was able to attack around 54 companies demonstrating the havoc an AI-driven ransomware can unleash [2]. With such unprecedented tactics, it becomes imperative to develop and implement rigorous detection mechanisms capable of thwarting new and evolving ransomware attacks. The recent explosion of developments within the field of artificial intelligence has engendered promising machine learning (ML) and deep learning (DL) models to identify and mitigate ransomware attacks through behavioral analysis and anomaly detection. The work presented in this report aims to evaluate and compare the performance of existing ML and DL models in predicting ransomware infections. The primary objective of this report is to assess the accuracy of these models and identify key features that contribute to their predictive effectiveness. By conducting a comparative analysis, this work seeks to answer the following questions –

Q1 – How accurate are these ML and DL models in predicting a ransomware attack over a given dataset?

Q2 – What are the most influential features that impact the performance of these models?

Q3 – How well do these features contribute towards a model's predictive capabilities?

The expected contribution of this work lies in offering insights into each model's effectiveness and features that influence their performance. By visualizing the findings produced by this work through an interactive dashboard, this work will further enhance the interpretability and usability of models for current industry professionals. This work can further help optimize feature selection, improve model interpretability, and aid future enhancements in AI-driven malware defense systems. The findings presented in this report can serve as a valuable guide for cybersecurity practitioners, researchers, and organizations seeking to deploy ML-based ransomware detection systems effectively.

II. LITERATURE REVIEW

Efficient ransomware detection can be achieved by understanding specificities associated with typical ransomware infections observed over time. These infections often begin with social engineering tactics employed to lure unsuspecting users into providing access to their systems. These tactics have

evolved over time and continue to exploit human factors to deploy their malicious programs. Once these programs enter a system, they start encrypting data or lock the owners out of their systems until they pay a specified ransom. Thus, based on the mechanism employed, ransomware can be classified into two types – ***Locker Ransomware and Crypto Ransomware***. Additionally, they can also be classified according to the type of encryption as -

Symmetric – Uses symmetric encryption algorithms such as Advanced Encryption Standard (AES) and Data Encryption Standard (DES) utilizing a single key for encryption and decryption of victim's data.

Asymmetric – Uses asymmetric encryption algorithms with the public key normally embedded within a ransomware or downloaded from a command-and-control server (C2 Server).

Hybrid – This approach typically utilizes a combination of symmetric and asymmetric encryption algorithms; often using a symmetric key to encrypt the victim's file and data while utilizing public and private keys for the encryption of the symmetric key.

Detection of various kinds of ransomware is normally done via static or dynamic analysis. Static analysis offers security by examining code without execution while lacking insights about packed or obfuscated malware. This is overcome by dynamic analysis which executes the malicious code in a controlled environment, effectively identifying behavioral patterns. These tactics are used to generate signatures for implementing detection mechanisms but are rendered unreliable when dealing with new variants of malware. However, such methods can be improved in their detection capabilities by using AI techniques involving Machine Learning (ML) and Deep Learning (DL) models. These methods are powerful and can mimic and outperform human intelligence and further aid in automating decision-making processes.

Khurana [3] evaluates ML-based detection techniques and compares them with traditional signature-based mechanisms to detect ransomware. The author uses self-organizing maps (SOM), Random Forest Classifier and LSTM network to provide insight into the effectiveness of ML-based detection approaches. Further, the study looks at true positive and false positive rates to gauge efficacy of these models in discovering threats and minimizing false alarms. Similarly, Fernando et al. [4] extends the idea of using AI techniques in ransomware detection by presenting a comprehensive literature review of proposed methods. The authors provide a detailed analysis of machine learning and deep learning algorithms used for ransomware detection and explore the possible ways they may evolve over the years. They introduce the concept of 'longevity' that studies how the algorithms work over older and newer generations of ransomware. Using 'concept drift' their study tries to understand how old the data must be for the newer algorithms to be trained efficiently such that their periodical retraining can be prevented. Nevertheless, the experiment could've used more datasets to provide a comprehensive overview of how the existing research differs and how the data should be collected for them.

Hammadeh and Kavitha [5] provide an in-depth examination of ransomware detection techniques using ML and DL approaches. Their study tracks strengths and weaknesses of various architectures such as Long Short-Term Memory (LSTM), Support Vector Machine (SVM), K-Nearest Neighbor (K-NN), and Logistic Regression in ransomware classification. The results of their work indicate effectiveness of DL models like LSTM in detecting ransomware with an accuracy of 99.08%. The dataset used by the authors included application and system files that form the foundation of most digital activities, malicious ransomware samples from VirusShare and a set of YARA rules typically used for the detection of crypto signatures. While it lacks a comprehensive set of features for analysis, their study emphasizes future work in the direction of Convolutional Neural Networks (CNN) in combination with LSTM.

The work presented by Bold et al. [6] includes an extensive literature review of machine learning models used in detecting ransomware. They trained and evaluated machine learning models with the help of six algorithms including Support Vector Machine (SVM), K-Nearest Neighbours, Logistic Regression, Decision Tree, Random Forest and Artificial Neural Network. Their analysis tried to determine which ML models provided reliability and what error parameters and evaluation matrices were utilized by them. They observed a lack of emphasis on false negatives employed by most studies while evaluating a model's performance. On the other hand, they also noticed the lack of attention given to the study of Linux-targeting ransomware and malware when compared to Windows-based threats that normally include Windows API calls.

Another notable work in the direction of the type of data to be used for ransomware detection is presented by Aljabri et al. [7]. Their work highlighted the effectiveness of using memory dumps for ransomware detection. They used algorithms like Random Forest (RF), LightGBM, Adaptive Boosting (AdaBoost), Extra Tree (ET) and XGBoost to analyse recently obtained ransomware samples. They demonstrated that while memory dumps are effective indicators of detection, they are not suitable for real-time detection. While including memory dumps in the dataset reduced false positives by 2%, the sample used by them was quite small. Their study leaves room for comparative analysis of ML and DL approaches to understand improvements and trade-offs with each approach. Apart from the type of data, feature selection also plays an important role in the detection of ransomware. Malik et al. [8] study the features that are most important for ransomware detection. Their experiment identified process IDs, and the actions associated with them (found in log files) as the most important features contributing towards effective detection. In contrast, the results obtained after execution of tasks by a program did not play any role in detecting the presence of ransomware. Additionally, the authors were also able to determine time as a crucial feature which can be used to map sequence of events to classify a program as benign or malicious

Across these studies, typical themes considering type of data, features and predictive accuracy emerge. However, they lack a deeper analysis of how these features influence predictive capabilities of common machine learning and deep learning models. Our work focuses on addressing this gap while providing a comparative analysis of popular models to enhance the development of efficient ransomware detection systems.

III. METHODOLOGY

A. Dataset and Resources

In this project we employed the UGRansome Dataset [9], publicly available on Kaggle, which contains labeled network traffic data associated with ransomware activities. The dataset comprises 149,043 samples across 14 features, capturing both technical indicators and transactional metadata.

- Network Attributes: Protocol (communication protocol), Flag (TCP control flags), Netflow_Bytes (total bytes in flow), Port.
- Ransomware Metadata: Family (ransomware family), BTC and USD (cryptocurrency payment values), SeddAddress and ExpAddress (wallet addresses).
- Target Variable: Prediction, a multi-class label categorizing samples as:
  - SS: Suspicious Activity (28,071 instances)
  - A: Attack (29,793 instances)
  - S: Severe Attack (46,466 instances)

B. Data Processing Pipeline

1) Preprocessing:

- Prior to model training, a comprehensive data preprocessing routine was applied:
- Feature Removal: The IPaddress and Time columns were removed. The former to ensure anonymization, and the latter due to its low predictive relevance.
- Categorical Encoding: Categorical features (Protocol, Flag, Family, SeddAddress, ExpAddress, and Threats) were encoded using one-hot encoding to ensure model compatibility.
- Feature Scaling: Numerical features (BTC, USD, Netflow_Bytes, Port, Clusters) were standardized using StandardScaler to normalize value ranges.
- Missing Values: Rows containing missing or null values were excluded from the dataset to prevent skewed model learning.

2.2 Class Imbalance Mitigation

The dataset exhibited significant class imbalance, with the severe attack class (S) being overrepresented. To correct this, a hybrid resampling strategy was adopted:

- Synthetic Minority Over-sampling Technique (SMOTE) was used to oversample the minority classes:
  - SS: Increased from 28,071 to 42,000 samples
  - A: Increased from 29,793 to 38,000 samples
- Random Under-sampling was subsequently applied to the majority class:
  - S: Reduced from 46,466 to 37,000 samples

This approach ensured a more balanced class distribution for effective model learning.

C. Model Development

Three supervised learning models were selected based on their interpretability, robustness to noise, and performance on imbalanced datasets:

Logistic Regression:

- Employed as a baseline linear classifier.
- Configured with class_weight='balanced' to compensate for residual class imbalance.

Random Forest Classifier:

- An ensemble of 100 decision trees trained using the Gini impurity criterion
- Provided interpretability through feature importance analysis

XGBoost Classifier:

- Implemented gradient-boosted decision trees optimized for imbalanced datasets
- Used scale_pos_weight=1.5 to adjust for skewed class distributions
- Accelerated using GPU support for improved training efficiency

D. Training and Evaluation Protocol

1) Data Partitioning: The dataset was split into training and testing subsets using a stratified 70:30 split, ensuring proportional class representation across both sets:

- Training Set: 104,330 samples
- Testing Set: 44,713 samples

2) Pipeline Integration: A modular preprocessing and resampling pipeline was constructed using scikit-learn and imblearn:

- Feature Processing: One-hot encoding and standardization were combined using a ColumnTransformer
- Resampling: SMOTE and RandomUnderSampler were integrated into an imblearn.Pipeline to streamline data balancing

3) Model Training: All models were trained using the balanced training dataset. Hyperparameters were set to default values to prioritize reproducibility and generalizability over performance tuning.

4) Evaluation Metrics: Model performance was evaluated using both aggregate and class-wise metrics:

- Accuracy, Precision, Recall, F1-Score: Reported as macro and weighted averages
- Confusion Matrix: Analyzed misclassification patterns across the three classes
- ROC-AUC Curves: Computed using a one-vs-rest strategy to assess multi-class discrimination
- Feature Importance:
  - Random Forest: Gini-based importance scores
  - XGBoost: SHAP (SHapley Additive exPlanations) values for interpretability at both global and instance levels

E. Required Resources

- The analysis was conducted using the following software and hardware infrastructure:
- Programming Language: Python 3.10
- Environment: Jupyter Notebook
- Visualization: PowerBI
- Libraries:
  - scikit-learn v1.3
  - xgboost v2.0
  - imbalanced-learn v0.12
  - pandas v2.0, matplotlib, seaborn
- Hardware Configuration:
  - CPU: Intel Core i5-12700H
  - RAM: 16 GB
- Data Collection: The ransomware detection experiments reported in this paper are based on the UGRansome Dataset, publicly released on Kaggle by Dr. Mike Wa Nkongolo and last updated in early 2023 [9]. The UGRansome dataset was created in 2021 by merging features from the original UGR '16 network telemetry corpus with a dedicated ransomware transaction dataset, using a PCA-based feature extraction followed by a fuzzy-merging algorithm to consolidate the most informative attributes. The raw data comprises 149,043 samples and 14 columns.

IV. RESULTS

A. Overall Model Performance

Table-1 summarizes the global performance of each classifier on the held-out test set (44,713 samples).

Model/Parameter	Accuracy	Precision	Recall	F-1 Score
Logistic Regression	0.8792	0.8818	0.8792	0.8796
Random Forest	0.9882	0.9882	0.9882	0.9882
XGBoost	0.99061	0.9907	0.9906	0.9906

TABLE I
OVERALL PERFORMANCE METRICS

B. Class-Wise Breakdown

Logistic Regression

- SS (Suspicious):
  - Precision = 0.87
  - Recall = 0.93
  - F1-Score = 0.90
- A (Attack):
  - Precision = 0.82
  - Recall = 0.84
  - F1-Score = 0.83
- S (Severe):
  - Precision = 0.93
  - Recall = 0.87
  - F1-Score = 0.90

Random Forest XGBoost:

- All three classes scored 0.98 on precision, recall, and F1-score.
- The highest consistency was observed in XGBoost, with class-level F1-scores of 0.99 across the board.

The linear model struggles most with the mid-severity "A" class, whereas tree-based models resolve nearly all inter-class ambiguities.

C. Confusion Matrix Observations

Logistic Regression shows off-diagonal errors primarily between adjacent classes (e.g., mis-labelling some "A" flows as "SS" and vice versa). Random Forest and XGBoost confusion matrices are almost perfectly diagonal, indicating an extremely low rate of both false positives and false negatives for each severity level.

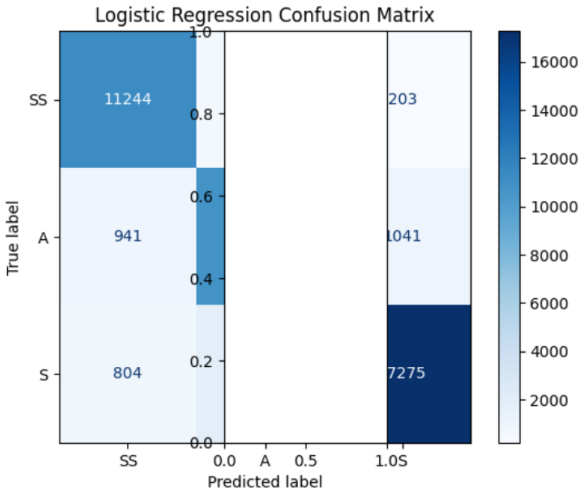

Fig. 1. Logistic Regression Confusion Matrix

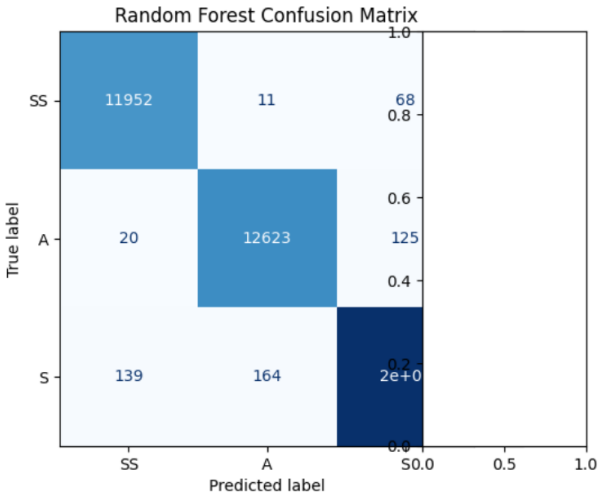

Fig. 2. Random Forest Confusion Matrix

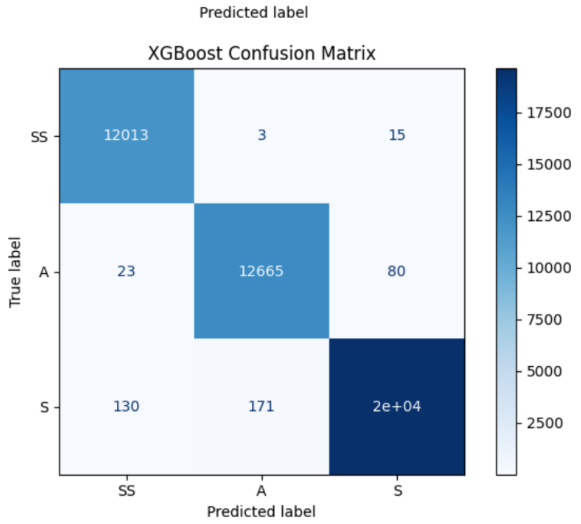

Fig. 3. XGBoost Confusion Matrix

D. ROC-AUC Analysis

Logistic Regression yields strong but slightly lower discrimination (AUC 0.95–0.97 per class). Random Forest and XGBoost achieve near-perfect ROC curves (AUC 0.99 for every class), demonstrating outstanding multi-class separability.

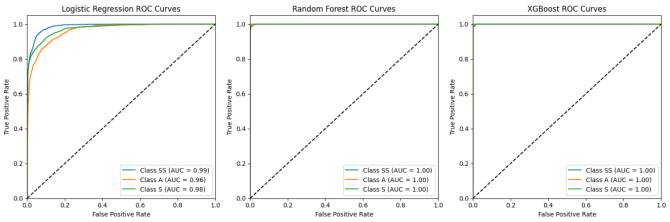

Fig. 4. ROC Curves

E. Feature Importance

The top predictors identified by both Random Forest and XGBoost are:

- Netflow_Bytes (total flow volume)
- BTC / USD (cryptocurrency transaction amounts)
- Protocol indicators (e.g., TCP SYN, ACK flags)
- Ransomware Family
- Threat flags

These features account for the majority of model discriminative power, underscoring the central role of data volume and payment metadata in detecting and classifying ransomware traffic.

V. Discussion

In this study, we evaluated and compared three machine-learning approaches—Logistic Regression, Random Forest, and XGBoost—to detect and classify ransomware activity using the UGRansome network-traffic dataset. We designed a robust data-processing pipeline that included feature engineering, normalization, and a hybrid SMOTE/under-sampling strategy to address pronounced class imbalance. Our findings demonstrate that, while a simple linear model can serve as a baseline, ensemble tree-based methods yield substantially higher detection accuracy and class-specific performance.

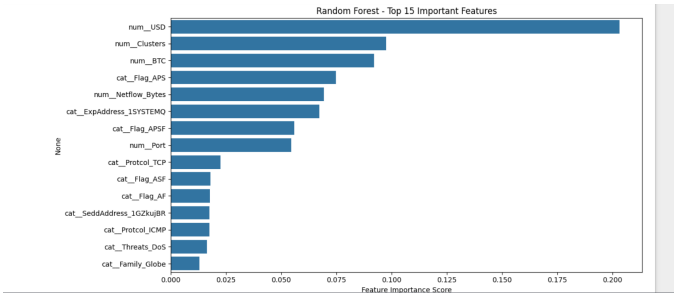

Fig. 5. Random Forest - Top 15 Important Features

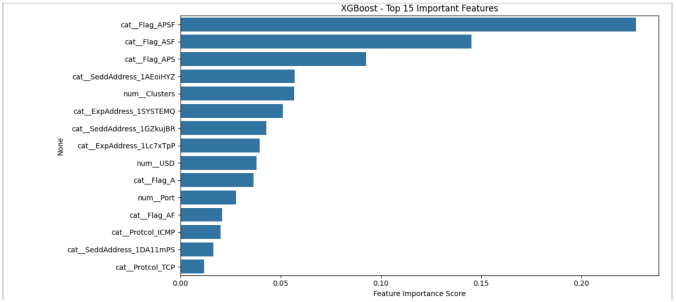

Fig. 6. XGBoost - Top 15 Important Features

Despite these strong results, several limitations must be acknowledged. The UGRansome dataset focuses exclusively on network-level telemetry and omits host-based artifacts (e.g., memory dumps, process behavior), which may be critical for detecting sophisticated ransomware variants. Additionally, our hyperparameter configurations were left at defaults to ensure

Feature-importance analyses revealed that network flow volume (Netflow_Bytes), cryptocurrency transaction values (BTC/USD), and specific protocol/flag indicators were the primary drivers of model decisions. This insight not only validates the relevance of transaction metadata in ransomware detection but also provides practical guidance for network monitoring systems: prioritizing these high-impact features can streamline real-time detection and reduce computational overhead.

Specifically, Random Forest achieved 98.82 % accuracy and balanced precision/recall across all ransomware severity levels, and XGBoost further improved on this with 99.06 % accuracy and near-perfect F1-scores for each class. Confusion matrices for the ensemble models were almost fully diagonal, indicating minimal false alarms or missed detections. ROC-AUC analysis reinforced these results, with both Random Forest and XGBoost obtaining class-wise AUCs of approximately 0.99, highlighting their superior discriminative power in a multi-class setting.

reproducibility; further tuning could yield marginal gains. Finally, real-world deployment would require continuous model retraining and validation to adapt to evolving ransomware tactics.

Future work will explore the integration of multi-modal data—combining network telemetry with endpoint logs and host-level signals—to enhance detection robustness. We also plan to investigate deep-learning architectures (e.g., graph neural networks on network flows) and automated hyperparameter optimization techniques to push detection accuracy even closer to 100 %. Ultimately, the methodologies and insights presented here form a solid foundation for developing production-grade ransomware detection frameworks capable of safeguarding critical infrastructure.

VI. Appendix

A PowerBI dashboard was created for visualization of the results obtained after training and testing the models to demonstrate the use of such tools in helping cybersecurity professionals gain additional insights about model performance.

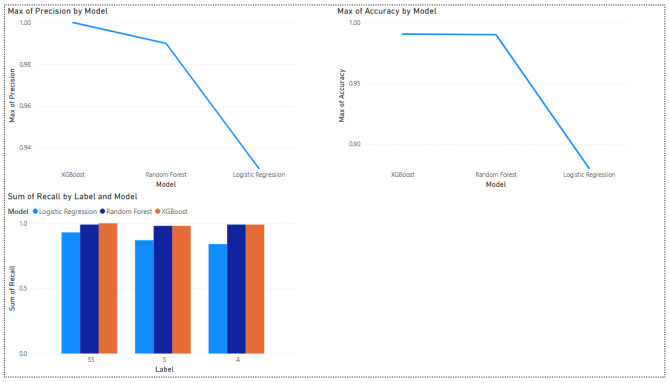

Fig. 7. PowerBI Dashboard

References

[1] "Recent ransomware payments and settlements."

[2] "Exploring the depths: Includes 2024 quarterly analyses!."

[3] S. Khurana, "Ransomware threat detection and mitigation using machine learning models," in *2023 IEEE International Conference on ICT in Business Industry & Government (ICTBIG)*, (Indore, India), pp. 1–6, 2023.

[4] D. W. Fernando *et al.*, "A study on the evolution of ransomware detection using machine learning and deep learning techniques," *IoT*, vol. 1, pp. 551–604, Dec. 2020.

[5] K. Hammadeh and M. Kavitha, "Unraveling ransomware: Detecting threats with advanced machine learning algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 14, Jan. 2023.

[6] R. Bold *et al.*, "Reducing false negatives in ransomware detection: A critical evaluation of machine learning algorithms," *Applied Sciences*, vol. 12, p. 12941, Dec. 2022.

[7] M. Aljabri *et al.*, "Ransomware detection based on machine learning using memory features," *Egyptian Informatics Journal*, vol. 25, p. 100445, Mar. 2024.

[8] S. Malik, B. Shanmugam, K. Krishnan, and S. Azam, "Critical feature selection for machine learning approaches to detect ransomware," *International Journal of Computing and Digital Systems*, vol. 11, pp. 1167–1176, Mar. 2022.

[9] M. Nkongolo, J. P. van Deventer, and S. M. Kasongo, "UGRansome1819: A Novel Dataset for Anomaly Detection and Zero-Day Threats," *Information*, vol. 12, p. 405, Oct. 2021.

[10] T. Mane, P. Nimase, P. Parihar, and P. Chandankhede, "Review of malware detection using deep learning," in *Proceedings of the International Conference on Advances in Computing and Data Sciences*, pp. 255–262, Oct. 2021.

[11] A. Razgallah, R. Khoury, S. Hallé, and K. Khanmohammadi, "A survey of malware detection in android apps: Recommendations and perspectives for future research," *Computer Science Review*, vol. 39, p. 100365, 2021.

[12] M. Hirano and R. Kobayashi, "RanSMAP: Open dataset of Ransomware Storage and Memory Access Patterns for creating deep learning-based ransomware detectors," *Computers Security*, vol. 150, p. 104202, 2025.

[13] A. Yewale and M. Singh, "Malware detection based on opcode frequency," in *2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, 2017.

[14] Z. Xu, S. Ray, P. Subramanyan, and S. Malik, "Malware detection using machine learning based analysis of virtual memory access patterns," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2017.

[15] J. Rutkowska, "Introducing stealth malware taxonomy." Online, 2006. Accessed: [Insert Date].