

Anomalix: intelligent static portable executable metadata-based ransomware detection using machine learning

Yuvankumar M S¹ Rajammal K², and Rohith S³

¹Dept of CSBS, Rajalakshmi Engineering College Chennai, India 221401123@rajalakshmi.edu.in

²Dept of CSBS, Rajalakshmi Engineering College Chennai, India rajammal.k@rajalakshmi.edu.in

³Dept of CSBS, Rajalakshmi Engineering College Chennai, India 221401076@rajalakshmi.edu.in

Abstract. Ransomware is one of the worst forms of Cyberattack. Often it escapes the detection of traditional signature-based antivirus (AV) solutions or so-called legacy AV solutions through the use of obfuscation, polymorphism and exploit kits that exploit zero-day vulnerabilities. In response, we propose ANOMALIX, the static analysis-based ransomware detection framework. The framework uses both Portable Executable (PE) metadata as well as machine learning techniques to determine whether the intent is malicious or not before the execution takes place. Structural attributes - like import tables, sections, resources and memory configuration - were transmuted into discriminative features. A set of classifiers including Random Forest (RF), Decision Tree, Logistic Regression and XGBoost classifiers were trained on the resulting feature set extracted from a Kaggle PE malware dataset and the Random Forest classifier was found to be the best of the classifiers evaluated. The system is implemented as a web based service with functionality provided include confidence scoring, interpretability of AI, Automated reporting and email notification. Empirical evaluations have shown that static analysis of PE metadata is a safe, efficient and scalable solution to provide early detection of ransomware threats.

1 Introduction

The rapid growth of digital data exchange has significantly increased exposure to sophisticated cyber threats, of which ransomware is one of the most financially and operationally devastating attack vectors. Ransomware campaigns have a specific goal; they attack vital data resources and encrypt user files and demand ransom payments in exchange for decryption keys, often in digital currency. The impact of such attacks is not just in terms of financial loss, business continuity, data integrity, and institutional reputation are also severely affected.

Traditional antivirus systems are largely based on signature-based detection mechanisms, which compare the executables with known malware fingerprints. While they continually work well to combat ransomware threats that have been identified, these systems are ill-equipped to find ransomware threat variants that use evasion strategies that employ code obfuscation, polymorphic mutation, and exploit kits that aim to exploit zero-day vulnerabilities. As a consequence, new strains of ransomware have often been able to bypass the legacy security rules and release encryption payloads before they can be detected.

To address these limitations, in recent years, researchers have focused more on new malware detection techniques, which are based on machine learning processes and have the potential to detect previously

unseen malicious patterns beyond the static signatures. Static analysis or more specifically Portable Executable (PE) metadata inspection is gaining attention as a secure and non-intrusive detection approach as well as a safer alternative to dynamic behavioral monitoring. By analyzing structural attributes - those providing insights into import tables, memory allocation parameters, section distribution, as well as resources configurations - potential ransomware indicators may be found, without even having to execute the provided binary which rules out the risk of runtime attacks.

This research is to propose ANOMALIX, an intelligent ransomware detection research framework that integrates static PE metadata analysis using supervised machine learning classification. Multiple algorithms such as Random Forest, Decision tree, Logistic Regression, XGBoost, etc. Tested on a Kaggle PE malware dataset to find the best performance to predict the system is implemented through a web-based analytical system with confidence scoring, AI assisted interpretability, automation reporting & alerted notifications via mails.

The major contribution of this work is the development of a ransomware detection environment which is secure and execution free, and which consists in the combination of structural bi-intelligence & applied implementation capabilities. By dealing with the gap between machine learning classification and threat analysis that is available to users, the proposed system aims at strengthening ransomware defence mechanisms in

the early stages of cybersecurity ecological systems in the modern world.

2 Related works

Ransomware detection has come a long way in the last decade, as the cyber extortion movement has become increasingly sophisticated. Relying very much on signature based anti viral frameworks, early detection mechanisms compared the fingerprints (of executables) to repos of known malware-programs. However, advanced ransomware families are equipped with advanced evasion technologies such as polymorphic mutation, encryption packing, anti-analysis techniques but also zero day exploit distribution, making static signature databases less and less effective [13], [19]. This paradigm shift has caused moving cybersecurity research towards smart, data experiences to make it possible to detect strategies which are capable of detecting the malicious intent through structural and behavioral indicators (as opposed to known code signatures).

Static malware analysis has become an application of security and free-running ransomware analysis. By analyzing Portable Executable (PE) Structures - such as header configurations, import table, sections entropy and memory allocation parameters - researchers have shown that it is feasible to detect ransomware without triggering its payload.

Manavi and Hamzeh [10] proposed a ransomware detection model based on a PE-header which transforms structural metadata into grayscale representation, which is fed to convolutional neural networks model, with reasonable classification accuracy. Similarly, Mohamed et al. [11] proposed an Enhanced Joint Mutual Information feature-selection technique that focuses on API-call indicators that are highly relevant in identifying ransomware, which helps in early-stage ransomware detection, improving the performance of early-stage detection. These studies underscore the discriminative power of the PE metadata features that make them amenable for machine learning-based classification frameworks such as AnomaliX.

Machine learning has been a crucial tool in the process of detecting malware as it allows the machines to learn the complex patterns in the structure and behavior of malware from a large dataset. Ensemble classifiers - Random Forest in particular - have proven to have good predictive capabilities because of their relatively high robustness against feature noise and overfitting.

Ahmadian and Shahriari [2] proposed the 2entFOX framework that capitalizes on the power of Bayesian classification techniques over ransomware feature sets to identify highly survivable ransomware variants.

Behavioral classification using optimisation algorithms such as particle swarm optimization wrappers have further increased the efficiency of feature selection, as well as detection accuracy [16]. These works validate the effectiveness of the supervision learning models in ransomware classification and also inform about the algorithmic selection in AnomaliX detection Engine.

Dynamic analysis techniques are a means of

monitoring ransomware's behaviour during execution to find patterns of act of being malicious to the sufferers. RATAFIA, proposed in [5], uses time and frequency domain autoencoders to identify anomalous disk operations related to encryption routines. Similarly, behavioral modeling frameworks based on finite states track the transitions of the system to recognize ransomware progression stages [14].

Two-stage detection architectures by combining Markov sequence modeling with machine learning classifiers have produced high detection accuracy with a low level of false positives [7]. Despite their effectiveness, these approaches require sandbox execution environments which creates an increased computational overhead and creates operational risk for a system - which is mitigated by static analysis systems such as AnomaliX.

Several papers address the ransomware detection in the deeper layers of the system. Kernel level monitoring frameworks would intercept the system routine to detect malicious file operations and kill ransomware in real-time [8]. Memory forensic approaches include the snapshots of volatile virtual machines to extract ransomware for classification [4].

While these infrastructure-centric techniques offer high detection visibility, they frequently impose intrusive system instrumentation, virtualization overhead or unique deployment environments, limiting their scalability for lightweight systems detection platforms.

Network behavior analysis has been also explored as a vector of ransomware detection. Software-Defined Networking (SDN): SDNs are frameworks that track these communication patterns of the infected hosts to the command and control servers [20]. File-sharing traffic anomaly detection models detect abnormal patterns of access during the propagation of ransomware [12].

Although good for detecting active infections, such network-based systems have a post-execution working mode and are unable to prevent initial payload deployment, which is why pre-execution detection mechanisms are important.

Preventive detection models concentrate on stopping the ransomware activity in the stages of early encryption. Honeyfile-based systems such as R-Locker drop decoy files in the attempt to divert access attempts of ransomware [6] and CryptoDrop detects fast patterns of file modifications to mitigate early [15].

Threat hunting and cyberspace intelligence surveys also pay heightened attention to proactive monitoring techniques for detection of new ransomware campaigns [1]. However, these approaches frequently make continuous monitoring infrastructures and expert intervention compulsory.

3 Proposed System

ANOMALIX is designed as a static, execution-free ransomware detection platform which in addition to Portable Executable (PE) structural metadata analysis also uses supervised machine learning classification. The system is designed to detect malicious binaries before

they are executed thus avoiding runtime exposure threats that occur with sandbox and behavioral detection mechanisms.

The framework combines the processes of feature extraction, data preprocessing, model training and real-time predictions in a web accessible interface. Users can enter the metadata attributes of the executables, and gain classification results with confidence scores and interpretative information. The system focuses on early-stage detection, computational efficiency, and practical deployment usability. Unlike traditional antivirus systems, which are based on signature matching, ANOMALIX targets the structural patterns that are embedded inside the PE files allowing to find known and previously unseen variants of ransomware.

3.1 System Architecture

ANOMALIX has a system architecture that consists of a combined multi-layered analytical platform offering relations between machine learning, user interaction, threat intelligence reporting, threat intelligence inference, and threat intelligence in a single no execution environment. The architecture is in three operational areas; frontend interface, back-end services to machine learning. The first and the final layer is called pipeline and the last one is the persistent data layer, which is a part of the classification of live time ransomware threats.

All the user side interactions are handled through the frontend layer. It begins with a safe user authentication procedure in which only registered users will access the analytical services. When the successful users log in, they are redirected to a dashboard where the main point of focus is navigation. Here, one can reach the prediction form in which Portable Executable metadata attributes are submitted to be analysed.

The interface also includes the addition of the AI chatbot module to offer contextual cybersecurity advice and more related assistance to enhance user ransomware conscience and understanding of security. Prediction data submitted is sent to the backend services and analytics workflow in which analytical processing takes place.

The PE has made the following first step: Data Handler to take in and to authenticate the metadata input. This is preceded by feature preprocessing which is done before features would comprise normalization and conversion to declare structured vectors as being model-relevant. The random forest classifier is then fed with processed features. The deployed predictive model was chosen because of its better stability and classification performance.

The output of the prediction is received to the prediction and report generator that is employed in generating classification results, confidences and analytical summaries. This knowledge is also enhanced by the AI Explanation engine which generates anthropocentric explanations of threats.

Another data layer that has the task of constant storage and historical tracking is also appended to the system. The purpose of this layer is to maintain the user database, prediction logs and archived analytical reports. The

history can be accessed through stored records and permit users to look at past threat analysis as report generation on the well download is on demand.

The final output is displayed through the interface of threat classification. Output results are in the form of Benign and or Malicious/Anomalous and confidence supported indicators. This makes this highly integrated architecture possible, smooth information exchange between the input and the output in the analysis, and does not sacrifice the security, scalability and execution free ransomware detection environment.

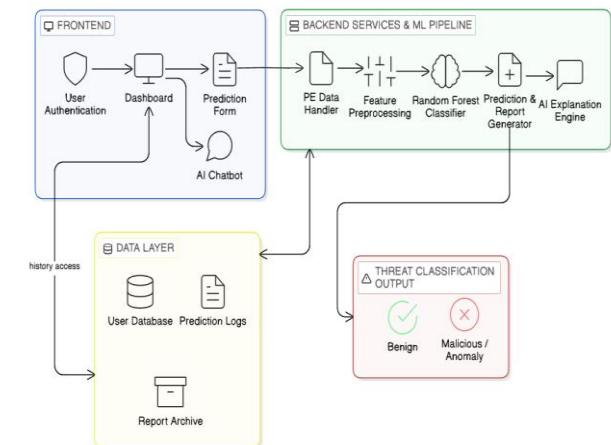


Fig. 1. Architecture Diagram

3.2 Feature Engineering and Dataset Preparation

The detection model is trained with the help of a labeled dataset on Kaggle PE malware dataset with benign and malware samples of Windows executables. Each instance is structured static data derived from Portable Executable (PE) files without the need of running the PE.

File Name	MD5 Hash	SHA256 Hash	Size (bytes)	Image Size (bytes)	Image Version	Image Date	Image Type	Image Subtype	Image Format	Image Architecture	Image Language	Image Code Page	Image Character Set	Image Subsystem	Image Origin	Image Integrity
1. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
2. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
3. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
4. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
5. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
6. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
7. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
8. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
9. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
10. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
11. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
12. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
13. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
14. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
15. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
16. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
17. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
18. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
19. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
20. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
21. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
22. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
23. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
24. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
25. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
26. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
27. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
28. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
29. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign
30. C:\Windows\System32\cmd.exe	5e995913ba6fae534a71979e2aa81801	5e995913ba6fae534a71979e2aa81801	524288	524288	10.0	20140618	PE32	Console	Windows	x86	English	UTF-8	Unicode	GUI	Microsoft Windows	Benign

Fig. 2. Sample Dataset of PE Metadata Features

Engineered features are broken down into three categories. The first are some PE header attributes such as machine type, debug size, image version fields, memory configurations, stack reserve size and characteristics of DLLs. The second of which captures import table indicators in the dependencies on external systems for libraries. The third is API call patterns which are linked to behaviors related to encryption, file manipulation, and system modification typically linked to ransomware activity.

Preprocessing, including missing values, etc. carrying out normalization, categorical encoding, balancing classes etc. These operations make a representative and consistent formulation of features.

3.3 Model Training and Analytical Components

All models have been trained on PE metadata samples that were labeled based on a uniform strategy of preprocessing and train-test split. Performance was evaluated from standard classification metrics such as accuracy, precision, recall, F1-score and AUC.

Based on comparative evaluation, the ensemble-based Random Forest classifier has shown the most stable learning behavior and it turned out to be the final deployed model. Its probability outputs are also used to calculate confidence scores in addition to classification results.

3.4 Workflow and Data Flow

The operational workflow is based on a structured prediction pipeline as integrated in the web-based detection. The process starts with secure authentication of the user and then the users provide Portable Executable (PE) metadata attributes via the prediction interface. The input data is validated and preprocessed at backend i.e., feature transformation is performed on raw data by converting it into vectors compatible for the model.

These processed features are then fed to the trained classification model to obtain the inference which will produce a binary output i.e. benign or malicious. In addition to the prediction, a confidence score is calculated in order to reflect the reliability of classification. The system further generates AI-assisted analytical explanations and a downloadable PDF Threat report with the options to receive email notifications to registered users.

All the analytical operations run by static inspection without triggering the ransomware payload during analysis.

3.5 Tools and Technologies Used

The proposed system is implemented with the help of a combination of machine learning, backend and web technologies used in order to ensure reliable deployment and usability. Python is the main programming language used in data preprocessing and feature engineering, and also for developing models. Classical Machine learning AI models were implemented using the Scikit-learn library and for gradient boosting based experimentation XGBoost has been used.

The backend of the web application is created by implementing the framework of Flask allowing to keep the model light and to easily accept predictions via the API. The user interface is constructed with the help of the web-based languages, such as HTML, CSS, and JavaScript, to give an interactive prediction dashboard.

This technology stack is guaranteed to have modular design, light deployment, and implementable scalability for real-world ransomware threat analysis.

4 RESULTS AND DISCUSSION

The proposed ransomware detection framework was tested using a labelled Portable Executable (PE) malware

data set, which was downloaded from Kaggle. The dataset contained benign and malicious executables samples which were characterized utilizing structured static metadata characterization attributes.

Model performance was evaluated by many well-known classification metrics such as Accuracy, Precision, Recall, F1-Score, and Area Under the ROC Curve (AUC). These are all measures of predictive reliability, error balance and generalization capability taken together.

4.1 Classification Performance Analysis

The Random Forest classifier achieved the highest detection accuracy followed by XGBoost and Decision Tree. Logistic Regression had relatively lower performance, mainly because of its linear decision boundary issue in working with complicated ransomware feature relations. ROC analysis also supported the superiority of ensemble learning, as the Random Forest ensemble model was able to obtain an AUC score of 0.996, representing the close-to-perfect class separability between benign and malicious samples.

Among the models tested, the ensemble learning Random Forest classifier turned out to have the highest predictive stability and an almost perfect classification accuracy. Its ensemble voting procedure allowed strong consideration of the feature variance while overfitting was reduced. These results enable the success of ensemble learning methods for static ransomware detection tasks to be validated.

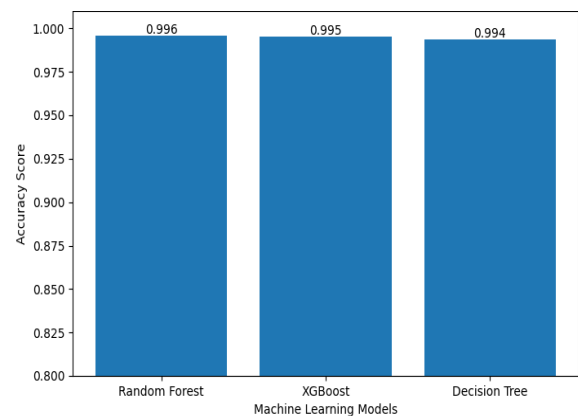


Fig .3. Model Accuracy Comparison

4.2 Confusion Matrix Interpretation

In order to further analyze the classification behavior, a confusion matrix was created for the final deployed Random Forest model. The matrix shows the distribution of the True Positives (malicious correctly detected), True Negatives (benign correctly classified), False Positives and False Negatives.

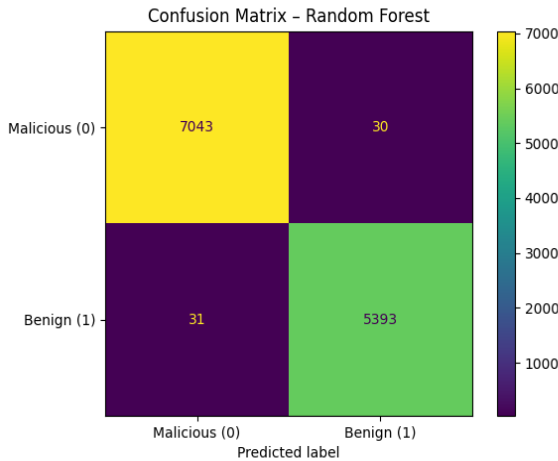


Fig.4. Confusion Matrix of the Random Forest Ransomware Detection Model

From the matrix, 7043 samples of malicious data were correctly predicted while only 30 samples of malicious data were incorrectly predicted as benign data. Similarly, 5393 benign samples were correctly classified, with 31 benign instances falsely predicted as malicious. These results provide extremely high detection precision with a minimal misclassification. The very low false negative rate is especially important in cybersecurity applications, because undetected ransomware could result in serious system compromise: At the same time, the low false positive rate means that legitimate executables will not be unnecessarily flagged which can maintain a stable operation. Overall, the confusion matrix presents a strong level of robustness and reliability of the proposed static metadata-based detection framework.

4.3 ROC Curve and AUC Analysis

The Receiver Operating Characteristic (ROC) analysis was done to test the discrimination ability of the deployed Random Forest classifier. ROC Curve shows the trade off between True Positive Rate (TPR) and False Positive Rate (FPR) at different thresholds of classification.

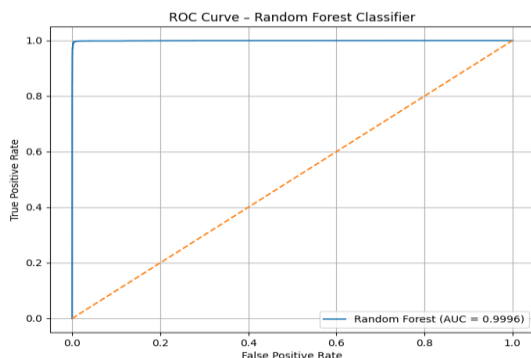


Fig. 5. ROC Curve of the Random Forest Classifier

As we can see from Figure 5 the Random Forest model was able to achieve an Area Under the Curve (AUC) value of 0.9996, which is an almost perfect separability between benign and malicious executable samples. The curve stays persistently close to the top-left corner of the plot thus providing extremely high

sensitivity for detection with minimal false positive rates.

Such a high AUC score proves the effectiveness of the proposed static metadata-based detection framework in distinguishing ransomware threats with Low false alarm probability, making it proper for reliable pre-execution malware detection.

4.4 Feature Correlation and Analytical Insights

To investigate inter-feature relationships a correlation heatmap was created for the metadata attributes for the engineered PE, representing the bounded attributes of PE metadata, which is represented in Figure 6. The visualization shows that the majority of the features have low to moderate levels of correlation, thus probably a low level of redundancy across the feature space.

Although some structural header parameters exhibit some proof of mild associations, no strong patterns of multicollinearity are evident in the data except for the diagonal self-correlations. This confirms that the feature which is selected contributes independently to the learning process as opposed to duplicating predictive information.

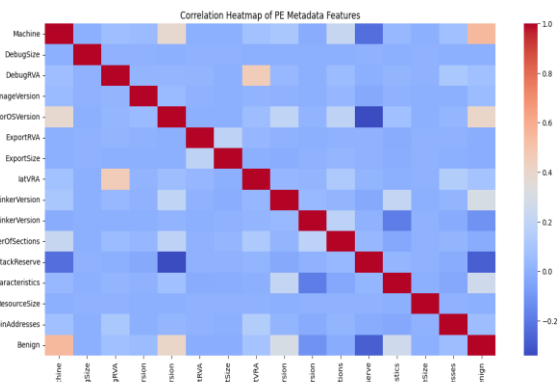


Fig. 6. Correlation Heatmap of Engineered PE Metadata Features

As it can be seen, the target variable ("Benign") is moderately correlated with specific structural attributes, supporting the relevance of the attributes in distinguishing malicious from legitimate executables. Overall the analysis of correlations helps in validation of the stability and suitability of the engineered feature set in the supervised classification tasks.

4.5 Confidence Scoring and Prediction Reliability

Other than the binary classification, the deployed model provides an output of probability-based confidence that corresponds to prediction certainty. These scores are quantitative reliability indicators of each analyzed executable sample. The higher the confidence values, the better the certainty of a model, which helps in making a decision in the real world of the cybersecurity monitoring environment.

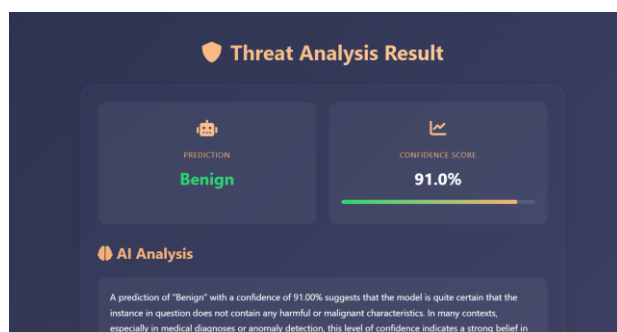


Fig. 7. Prediction and Confidence Scoring result

The combination of confidence scoring helps improve interpretability and allows risk prioritized threat response.

4.6 System Effectiveness Discussion

Experimental results also prove that static PE metadata analysis can be used for highly reliable detection of ransomware without binary execution. This is an execution-free approach, thus eliminating the risks of a sandbox, it prevents the activation of payloads, and it allows for a scalable pre-deployment screening. Ensemble machine learning helps to increase detection robustness by detecting complex structural threat signatures embedded in executable files.

Overall the proposed framework demonstrates a high level of operational feasibility in terms of early-stage ransomware identification enterprise.

5 Conclusion

This work introduced ANOMALIX, which is a static analysis-driven ransomware detection framework aimed to detect the malicious executables before their execution. By making use of structured Portable Executable (pe) metadata, the system makes secure execution-free threat assessment possible. Several supervised learning models were tried to determine the effectiveness of detection. Among them, results for the ensemble-based Random Forest classifier showed the best predictive reliability and provided better accuracy and class separation than other evaluated algorithms. The combination of scoring confidence, correlation validation further led to increased analytical robustness.

Beyond model development, the framework was made operational using a Flask-based Web platform based on end capabilities of real-time prediction, AI-assistant explanation, automated report generation, and emails alerting. This end-to-end deployment signifies the applicability of this system for real-world cybersecurity environments.

Overall, the results ensure that static PE metadata analysis in combination with machine learning constitutes a scalable and safe solution for early ransomware detection without the execution risk to the systems.

Reference

Journal Articles

[1] F. Aldauji, O. Batarfi, and M. Bayousef, "Utilizing Cyber Threat Hunting Techniques to Find Ransomware Attacks: A Survey of the State of the Art," *IEEE Access*, vol. **10**, pp. 61695–61706, Jun. 2022.

[2] M. M. Ahmadian and H. R. Shahriari, "2entFOX: A Framework for High-Survivable Ransomware Detection," in *Proc. Int. Symp. Computer Science and Intelligent Control (ISCISC)*, Tehran, Iran, 2016, pp. 1–6.

[3] A. Alzahrani, A. Alshehri, H. Alshahrani, and R. Alharthi, "RanDroid: Structural Similarity Approach for Detecting Ransomware Applications," *Int. J. Advanced Computer Science*, vol. **9**, no. 2, pp. 112–120, 2018.

[4] A. Cohen and N. Nissim, "Trusted Detection of Ransomware in a Private Cloud Using Machine Learning," *J. Cloud Security*, vol. **5**, no. 1, pp. 55–66, 2018.

[5] S. Dutta, S. Sinha, D. Mukhopadhyay, and A. Chattopadhyay, "RATAFIA: Ransomware Analysis Using Time and Frequency Informed Autoencoders," in *Proc. IEEE Security Symposium*, 2019, pp. 442–450.

[6] J. A. Gómez-Hernández, L. Álvarez-González, and P. García-Teodoro, "R-Locker: Thwarting Ransomware Action Through a Honeyfile-Based Approach," *Computers & Security*, vol. **73**, pp. 389–398, 2017.

[7] J. Hwang, J. Kim, S. Lee, and K. Kim, "Two-Stage Ransomware Detection Using Dynamic Analysis and Machine Learning Techniques," *Electronics*, vol. **9**, no. 11, pp. 1853–1864, 2020.

[8] D. Javaheri, M. Hosseinzadeh, and A. M. Rahmani, "Detection and Elimination of Spyware and Ransomware by Intercepting Kernel-Level System Routines," *J. Information Security*, vol. **10**, no. 3, pp. 221–232, 2019.

[9] S. Jung and Y. Won, "Ransomware Detection Method Based on Context-Aware Entropy Analysis," *Security and Communication Networks*, 2018.

[10] F. Manavi and A. Hamzeh, "A New Method for Ransomware Detection Based on PE Header Using Convolutional Neural Networks," *Pattern Recognition Letters*, vol. **138**, pp. 276–282, 2020.

[11] T. M. H. Mohamed, B. A. S. Al-Rimy, and S. A. Almalki, "Ransomware Detection Model Based on Joint Mutual Information Feature Selection Method," *Engineering, Technology & Applied Science Research*, vol. **14**, no. 4, pp. 15400–15407, 2024.

[12] D. Morato, E. Berrueta, E. Magaña, and M. Izal, "Ransomware Early Detection by the Analysis of File Sharing Traffic," *IEEE Trans. Network and Service Management*, vol. **15**, no. 3, pp. 1039–1053, 2018.

[13] M. N. Olaimat, M. A. Maarof, and B. A. S. Al-Rimy, "Ransomware Anti-Analysis and Evasion Techniques: A Survey and Research Directions," *ACM Computing Surveys*, **vol. 54**, no. 8, pp. 1–36, 2021.

[14] G. Ramesh and A. Menen, "Automated Dynamic Approach for Detecting Ransomware Using Finite-State Machines," *Int. J. Cybersecurity*, **vol. 5**, no. 2, pp. 90–102, 2020.

[15] N. Scaife, H. Carter, P. Traynor, and K. R. B. Butler, "CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data," in *Proc. USENIX Security Symposium*, 2016, pp. 1–18.

[16] M. S. Abbasi, H. Al-Sahaf, M. Mansoori, and I. Welch, "Behavior-Based Ransomware Classification: A Particle Swarm Optimization Wrapper Approach," *Applied Soft Computing*, **vol. 121**, 2022.

[17] Y. Zhang, M. Li, X. Zhang, Y. He, and Z. Li, "A Novel Ransomware Attack Method to Dynamically Generate Malicious Payloads," *Computers & Security*, **vol. 118**, 2022.

[18] A. Zimba and M. Mulenga, "Demystifying WannaCry Crypto Ransomware Network Attacks," *Digital Forensics Review*, **vol. 10**, no. 1, pp. 33–44, 2018.

[19] R. Gupta, S. Tanwar, G. Sharma, and I. E. Davidson, "Ransomware Detection, Avoidance, and Mitigation Scheme: A Review and Future Directions," *J. Network Security*, **vol. 14**, no. 2, pp. 85–102, 2022.

[20] K. Cabaj, M. Gregorczyk, and W. Mazurczyk, "Software-Defined Networking-Based Crypto Ransomware Detection Using HTTP Traffic," in *Proc. ICCCN*, 2017, pp. 1–6.