International Journal of Safety and Security Engineering
Vol. 16, No. 1, January, 2026, pp. 95-104
Journal homepage: http://iieta.org/journals/ijsse

Random Forest and One-vs-Rest Classification for Malware and Ransomware Detection

Nilesh U. Sambhe[1,2*], Prakash Prasad[3]

[1] Department of Electronics and Computer Science, Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur 440033, India
[2] Department of Computer Technology, Yeshwantrao Chavan College of Engineering, Nagpur 441110, India
[3] Department of Information Technology, Priyadarshini College of Engineering, Nagpur 440019, India

Corresponding Author Email: nilesh.sambhe@gmail.com

https://doi.org/10.18280/ijsse.160108

Received: 17 November 2025
Revised: 7 January 2026
Accepted: 23 January 2026
Available online: 31 January 2026

Keywords:
malware detection, ransomware detection, Random Forest Classifier, One-vs-Rest classification, cyberattack detection

ABSTRACT

The rapid evolution of malware and ransomware has reduced the effectiveness of conventional signature-based and heuristic detection systems. This study proposes a malware detection approach based on Random Forest Classifier (RFC) and One-vs-Rest (OvR) multiclass classification to improve the identification of diverse cyber threats. The proposed approach integrates Synthetic Minority Oversampling Technique (SMOTE)-based data balancing, feature selection using Random Forest (RF) feature importance, Principal Component Analysis (PCA), and preprocessing optimization to enhance model generalization and reduce false detections. The method was evaluated on a dataset containing nine attack categories, including Distributed Denial of Service (DDoS), Denial of Service (DoS), Structured Query Language (SQL) injection, phishing, and ransomware. Experimental results show that RFC achieved an accuracy of 95%, while the OvR classifier reached 97% in multiclass threat identification. Precision, recall, F1-score, Receiver Operating Characteristic (ROC) curves, and confusion matrices further demonstrate the reliability and discriminative ability of the proposed approach. The findings indicate that the model provides an efficient, scalable, and interpretable solution for malware and ransomware detection in endpoint security environments.

1. INTRODUCTION

Cybersecurity has been at a critical stage where the quantity, sophistication, and occurrence of malware and ransom attacks have grown to proportions that cannot be dealt with using conventional defense tools. Modern threat actors are increasingly using polymorphic code, file-less attacks, and artificial intelligence (AI)-based avoidance techniques, which take advantage of weak points in traditional signature-based scanners and heuristic rules, increasing the risk facing the organization. These dangers are reflected in mass data breaches, financial damages, compromised services, and cloud and Internet of Things (IoT) systems [1, 2]. The fast growth in the number of interconnected devices, fast networks, and automated systems increases the attack surface, making manual threat detection ineffective and requiring intelligent and automated defense models. Traditional signature-based antivirus solutions are only efficient in countering the established threat types but essentially useless in identifying zero-day malware versions and fast-evolving ransomware families [3]. Some of these limitations are overcome by heuristic detection methods by analysing the behaviour patterns, although they often have high false-positive rates, are computationally intensive, and are not adaptable to changing threat scenarios. Machine learning (ML) is also a promising technology in malware detection due to its ability to learn discriminative features, detect anomalies, and generalise over unknown types of attacks. However, the challenge of critical class imbalances, adversarial images, redundant features, and low multi-class classification rates with different cyber threats continues to face malware classification with ML, even today [4, 5].

These shortcomings need to be overcome with effective malware detection systems that combine adaptive learning, balanced data representation, optimised features, and scalable classification architecture. The current paper advances a multi-class detection system based on a Random Forest Classifier (RFC) and a One-vs-Rest (OvR) multi-class testing approach to improve the detection accuracy, interpretability, and resiliency. The combination of Synthetic Minority Oversampling Technique (SMOTE)-based data balancing, features extraction using the Random Forest (RF) importance measures and Principal Component Analysis (PCA), and multi-class discrimination using OvR will make the system more reliable in detecting the heterogeneous threats such as cross-site scripting, Distributed Denial of Service (DDoS) and Denial of Service (DoS), password attacks, phishing, Structured Query Language (SQL) injections, zero-day exploits, and ransomware. As opposed to the previous ones that utilize either a single-model classification or unbalanced datasets, the given approach is based on an end-to-end

pipeline that includes the preprocessing of data, adaptive learning, and a framework that can be extended to incorporate real-time threat intelligence updates, along with comprehensive performance assessment. The system has a much higher accuracy and strength through precision, recall, F1-score, Area Under Curve of the Receiver Operating Characteristic (AUC-ROC) analysis, and confusion matrix analysis. This framework achieves the combination of ensemble learning with multi-class classification and balanced data representation to mitigate a major set of constraints within current techniques and provide an efficient, lightweight, and scalable endpoint security.

1.1 Objectives

The main purpose of this investigation involves developing an AI-controlled malware detection framework that enhances its ability to detect malicious signals while eliminating incorrect false alarms. The main purpose of the given research is to create an ML-based malware and ransomware detection system that enhances the detection accuracy without increasing false alarms. The following are the specific objectives of research:

- To develop a working malware detection model by using the RF model to classify the benign and malicious activities with the help of network, protocol, and behavioral features.
- To improve the multi-class attack detection through the OvR classification approach, allowing it to discriminate malware and ransomware of various categories.
- To enhance the performance and strength of classification through SMOTE-based data balancing and feature selection through RF feature importance and PCA.
- To minimize false positives and false negatives by making use of ensemble learning and optimal preprocessing methods that are applicable in endpoint security settings.
- To enhance the interpretability of the models by assessing the performance based on the confusion matrices, ROC curves, and conventional measures, i.e., accuracy, precision, recall, and F1-score.

1.2 Structure of the paper

In this paper, it has been noted in the introduction that the challenges presented by modern malware and ransomware attacks were increasing, thus explaining the necessity of a flexible detection framework. The literature review is a survey of extant signature-based, behavioral, and machine-learning strategies, revealing salient knowledge gaps and the need to thoroughly evaluate them. One section describes the methodology of the proposed system that includes the data preprocessing, the process of class balancing using SMOTE, the feature selection algorithms, and the process of working with RF and OvR classifiers. The results and discussion section provides empirical results, which are used to measure the results of a model using accuracy, precision, recall, F1-score, ROC curves, and confusion matrices. Lastly, it concludes the work by presenting the input of the given research and giving potential future research directions, such as the incorporation of deep learning methods and the development of adversarial robustness.

2. LITERATURE REVIEW

Research efforts have been dedicated for years to developing detection methods against malware and ransomware threats, which strongly endanger cybersecurity systems. Traditional signature-based, along with heuristic-based malware detection methods, show clear limitations when confronting zero-day attacks and polymorphic malware, as well as complex evasion techniques. The difficulty of addressing malware detection and behavioral analysis persists despite which researchers now apply ML algorithms and AI methods. This part provides a review of present malware detection techniques, which examines their advantages and limitations, and locates the proposed study in the field of cybersecurity research. Devices used to detect malware made their first attempts through signature-based detection by analyzing known malicious code patterns with stored signature databases. Sambhe et al. [6] reviewed various insider threat detection approaches, highlighting the effectiveness of AI/ML-based and behavioral analysis techniques over traditional methods. It identifies key challenges like false positives, data limitations, and scalability, and suggests the need for adaptive, multi-source, and real-time detection frameworks, which Shabtai et al. [7] introduced through the monitoring of system calls and user behavior to identify anomalies. Behavioral analysis methods increased detection success, but they both produced multiple inaccurately identified threats, together with increased system processing requirements. Network security functions as a vital part in detecting cyber threats that attempt to breach endpoint systems. Zilberman et al. [8] evaluated network inspection traffic patterns by focusing on the necessity of real-time monitoring systems for blocking widespread attacks.

The article by Arabo et al. [9] uses process behavior analysis to distinguish ransomware from benign applications and also demonstrates low false-positive and false-negative rates using behavioral features and ML-inspired techniques. Muhammad et al. [10] focusing on smartphone security threats, including APTs, sensor-based attacks, side-channel attacks, and malicious apps on platforms like Google Play. The research of Sarker et al. [11] brought IntruDTree into the world as an ML intrusion detection framework with superior threat identification capabilities. The authors Roseline and Geetha [12] created an oblique RF-based malware detection system, which proved that ensemble learning methods provide superior results. Zhang et al. [13] conducted research on architectural design patterns for volumetric DDoS defense to examine the advanced level of cyber-attacks that target large-scale infrastructures. The need emerges for adaptive learning models because existing research demonstrates their necessity to adapt to emerging attack techniques.

The detection of mobile malware in network traffic through N-gram semantic-based neural network systems was proposed by Bai et al. [14]. Accurate network threat identification became possible through deep learning system analysis of packet activities and flow pattern irregularities, according to their research findings. These detection methods need advanced hardware systems to process data effectively, so they do not work well for real-time endpoint security. Crystal Ball is a machine-learning-based attack effectiveness classifier suggested by Berger et al. [15], which predicts the possible effect of new cyberattacks, emphasizing the need to perform threat evaluation proactively instead of as a

signature. A systematic bibliometric review of malware evolution was carried out by Mat et al. [16], where the trends in the research, approaches to detection, and additional issues of malware analysis emerged throughout the years. Jabbarnejad et al. [17] presented a parameter-free analytical approach to cyberattack detection of grid-connected converters, which showed strong detection rates without the need to perform large-scale model tuning or prior knowledge of attacks. Girirajan et al. [18] built a deep learning model for Android malware detection based on mixed bytecode images and an attention-based ResNet architecture, and obtained a higher classification rate through the learning of the spatial features. Akshay Kumar et al. [19] explored one-dimensional visual Malware classification using lightweight deep learning networks with reduced computational complexity but competitive detection with limited resources. Mohi ud din et al. [20] conducted a comparative analysis of various ML algorithms to separate malware and benign software, and showed that collectives tend to work better than individual classifiers, and there is a necessity for better flexibility and resistance to the new malware patterns. The research gaps identified from the reviewed studies are summarized in Table 1.

Table 1. Gap analysis

Reference	Focus Area	Findings and Limitations	Research Gap and Proposed Solution
Sambhe et al. [6]	Use of AI and ML techniques	ML-based approaches outperform traditional methods	Absence of real-time detection systems and a lack of adaptive and context-aware models. Multi-source ensemble learning framework
Shabtai et al. [7]	Behavioral malware analysis	Behavior-based detection is effective, but results in high false-positive rates	Improves precision and reduces false positives using RF feature importance and PCA
Zilberman et al. [8]	Network-based malware detection	Traffic filtering enables early detection but requires continuous monitoring and lacks endpoint focus	Proposes endpoint-level malware detection using behavioral and protocol-based attributes
Arabo et al. [9]	Detection of ransomware using process behavior analysis	Achieves good detection accuracy with low false rates. Limited to specific datasets / controlled environments	Lack of adaptive and real-time detection models. Develop AI/ML-based adaptive models
Sarker et al. [11]	ML-based intrusion detection	ML models outperform traditional IDS but are vulnerable to adversarial attacks	Uses ensemble RF with OvR classification to support adaptive learning
Zhang et al. [13]	DDoS defense patterns	Effective for network-layer DDoS attacks, but ignores malware detection	Extends protection to endpoint malware detection using multi-class RF and OvR
Girirajan et al. [18]	Android malware detection	High detection accuracy, but limited to Android platforms	Proposes platform-independent malware detection using engineered features and RF
Mohi ud din et al. [20]	ML model comparison for malware	RF outperforms SVM and DT but lacks real-time adaptability	Enhances multi-class malware detection using RF-based OvR learning

Note: SMOTE = Synthetic Minority Oversampling Technique; RF = Random Forest; PCA = Principal Component Analysis; AI = artificial intelligence; ML = machine learning; IDS = intrusion detection system; OvR = One-vs-Rest; Distributed Denial of Service (DDoS); SVM = support vector machine; DT = decision tree.

3. METHODOLOGY

A malware detection framework uses a consistent methodology that combines data preprocessing steps with feature selection, along with ML classification, together with real-time threat intelligence updates to boost cybersecurity defenses [21]. The framework works at an enterprise level for malware detection and mitigation because it delivers high accuracy together with adaptability and scalability. Two parts comprise the methodology explanation, demonstrated sequentially throughout this section.

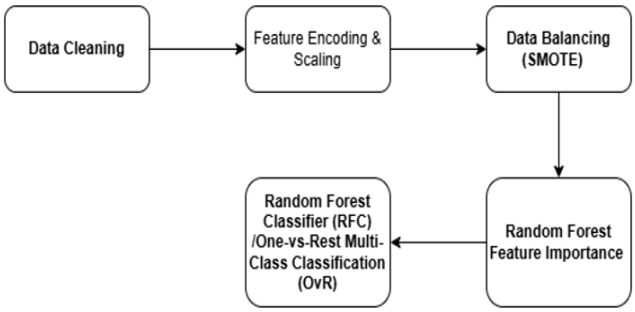

Figure 1. Block diagram

The initial part of malware detection processing uses data preprocessing to ready raw data for ML models through handling missing values, combined with encoding categorical attributes and scaling numerical dimensions and class balancing techniques [9]. The dataset D contains n samples that consist of m feature variables structured as $x_i \in \mathrm{Rm}$ with corresponding binary labels $y_i \in \{0,1\}$, indicating malware-classified instances and non-malware instances. Statistical techniques, including mean and median imputation, help replace missing data values in order to maintain complete participation in training procedures, as shown in Figure 1. The ML model requires binary vectors as input from one-hot encoding, which transforms categorical data for interpretation purposes. The z-score normalization computes standardized features through the following formula:

$$\mathrm{X_{Scaled}} = \frac{\mathrm{X} - \mu}{\sigma} \quad (1)$$

Both μ signifies the mean and σ stands for standard deviation in the formula. The standardization technique keeps all features balanced so the learning process avoids scale-based bias. The main problem in malware class identification rests in the huge disproportion between benign files and

malware files, causing predictions to become disproportionately biased [22]. The SMOTE serves as the solution to balance class distribution. SMOTE creates artificial samples of the minority class malware by computing the midpoint between real-class samples. A synthetic sample gets produced using the malware example x_{minor} along with the nearest neighbor $x_{neighbor}$.

$$x_{new} = x_{minor} + \lambda\left(x_{neighbor} - x_{minor}\right) \quad (2)$$

A value $\lambda \sim U(0,1)$ was randomly sampled from the uniform distribution. The method guarantees malware samples have optimal representation within the dataset, which leads to better model generalization.

A set of feature selection methods reduces unimportant or duplicate features that keep vital indicators for malware detection after dataset preprocessing [23]. The malware detection framework implements an RFC as the classifier while using OvR for multi-class malware classification enhancement.

3.1 Reproducibility details: Dataset, features, and evaluation protocol

A publicly available network intrusion dataset made available to academic research was experimented with, and a curated subset of about 150 labeled data in nine attack types and benign traffic was revealed to researchers. There was an imbalance in the classes of the original data, which was resolved by applying SMOTE only to the training folds, making the resulting data nearly balanced in terms of classes after over-sampling. It has features such as network identifiers (encoded sender/receiver identities and IP representations), protocol and transport features (protocol type, ports, packet size), and behavioral features such as flow statistics and flag features; categorical variables were one-hot encoded, and numerical features were z-score normalized. RF importance was used to rank feature relevance, and PCA was then applied to reduce dimensions. To assess the models, stratified k-fold cross-validation (5-fold in RF and 10-fold in OvR) with fixed random seeds was used to ensure that the results were reproducible, and the results of the evaluation were reported in terms of accuracy, precision, recall, F1-score, AUC-ROC, and confusion matrices.

3.2 Random Forest Classifier

RFC serves as an ensemble learning model for malware detection to build stronger accuracy and robustness in the classification outcomes. Researchers start the process by selecting features from the dataset, which includes properties of system behavior alongside network activity and file characteristics. Multiple decision trees (DT) receive their features through random distribution before undergoing training through distinct portions of the data, as shown in Figure 2.

Each DT learns unique patterns during its independent operation, which creates diverse proposed classifications. The aggregate voting decision of all DT determines the classification through the number of times one outcome is predicted by the majority. The model uses this procedure to eliminate biases across its trees and minimize overfitting [24]. The classification layer uses the group decision of the trees to determine whether an input sample is malware or benign. RFC uses a step-by-step procedure to improve its malware detection rates and function effectively as a dependable cybersecurity solution. The operation and deployment of the model achieve increased resistance to false positives while addressing new cyber threats through ensemble learning and majority voting techniques.

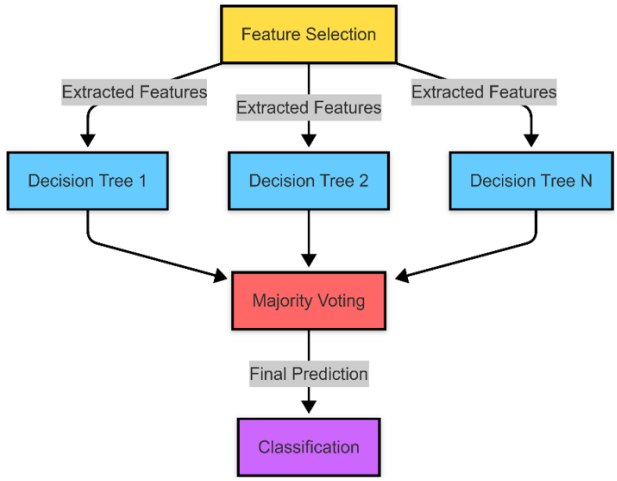

Figure 2. Random Forest Classifier (RFC) architecture for malware detection

RFC represents an ensemble learning technique that produces more accurate malware classifications through several DT aggregations. RF addresses information overfitting and high variance through multiple DT models, which use distinct subsets of data to create collective forecasting. RFC proves highly efficient for dealing with the typical cases of malware detection, which often include imbalanced data along with multivariable features as well as non-linear relationships [25]. RF's main benefit stems from its universal applicability across malware types, which happens through bootstrapping and feature randomness, while majority voting makes the classification model dependable and strong. RFC includes an organized set of procedures that perform classification work. Bootstrap sampling (bagging) generates numerous random subsets of data from the original dataset by using sampling with replacement as the first step. The independent DT receives individual subsets to train while maintaining different learning perspectives. RF differs from standard DT because it chooses random subgroupings of features during its node distributions to prevent tree correlation and boost predictive capability [26]. RF feature importance provides a ranking system that evaluates features based on their impact on model classification. The Gini impurity criterion applies to measure feature importance through the following calculation:

$$Gini = 1 - \sum_{k=1}^{K} P^2{}_k \quad (3)$$

Each node contains a probability value pk that represents the probability of class k within the node. The decreasing effect of *Gini* impurity when splitting by a specific feature X_j is calculated through:

$$\Delta Gini(x_j) = Gini_{parent} - \sum_{i=1}^{childern} \frac{N_i}{N} Gini \quad (4)$$

3.3 One-vs-Rest classifier

Multiple binary classifiers operate independently under the OvR classification framework to solve multi-class classification issues. Every classifier in OvR learns to distinguish exactly one class type while ignoring all other possible classes present in the dataset. This approach provides effective results for instances where users apply binary classification models such as logistic regression and support vector machine (SVM) or RF when handling multi-class problems [27]. The system starts with accepting feature vectors that include extracted attributes from the dataset. Each classifier receives different features from the K input classifiers to determine whether the input data belongs to its own class or not. The output of every classifier contains a probability value that shows the degree to which the analyzed data belongs in that particular class category, as shown in Figure 3.

Figure 3. One-vs-Rest (OvR) classification architecture for multi-class prediction

OvR classification represents a standard solution when handling multi-class problems through the creation of separate binary tasks for each class. The dataset contains N training samples made up of feature vectors alongside their corresponding class labels, and it takes this form:

$$D = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\} \quad (5)$$

The model contains features $x_i \in \mathrm{Rd}$ while $y_i \in \{1,2,...,K\}$ serves as the class label for determining the sample category. OvR classification trains individual binary classifiers that discriminate between each class and the entirety of the remaining classes. A binary classification task is built for each class k that uses y_k as its target variable [28]. A sample x_i with d dimensions enters as the input vector, which matches with the output label y_i among the set of categories K. Training K independent binary classifiers is the main purpose of OvR classification because each classifier needs to differentiate between one class and all remaining classes. The binary classification problem for every k uses y_k as the target variable, defined as:

$$Y_k = \begin{cases} 1, & \text{if the sample belongs to class k} \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

3.4 Evaluation metrics

Accuracy: Accuracy is a metric that quantifies the degree to which a model's predictions are right. The computation involves dividing the sum of true positive (TP) and true negative (TN) forecasts by the total number of predictions produced, which include TP, TN, false positives (FP), and false negatives (FN). Mathematically, precision is precisely defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (7)$$

Precision: Precision is a measure that calculates the ratio of correctly recognized positive predictions (TP) to all predictions classified as positive by the model. It demonstrates the precision of the model's favourable predictions [29]. Precision, in mathematical terms, is determined by dividing the number of TP by the total of TP and FP. The formula for precision is defined as:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (8)$$

Recall: Recall, often referred to as sensitivity or TP rate, is a quantitative measure that calculates the ratio of correctly predicted positive instances (TP) to the total number of actual positive cases. It denotes the model's capacity to accurately recognize all pertinent occurrences in a specific category. Mathematically, recall is computed by dividing the number of TP by the total of TP and FN. The recall formula is expressed as:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (9)$$

F1-score: The F1-score is a quantitative measure that calculates the harmonic mean of accuracy and recall, offering a balanced assessment of both. It is especially beneficial in situations when there is an imbalanced class distribution or where both FP and FN have equal significance. The F1-score may be expressed mathematically as follows:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (10)$$

4. RESULTS AND DISCUSSIONS

This section examines how an RF classifier with OvR multi-class classification detects malware through evaluation of the presented framework. To achieve endpoint security protection, the proposed detection model proves effective for identifying cyber threats and implementing countermeasures. The evaluation of classification performance relied on the implementation of metrics, which included accuracy, precision, recall, F1-score, confusion matrix, and AUC-ROC analysis. With an accuracy rate higher than 95%, the model proved its ability to separate malware from benign program files. The model demonstrates strong performance by

efficiently eliminating both wrong positive and wrong negative results, which makes it highly dependable for malware detection purposes. A confusion matrix evaluation measured the predictive performance of the constructed model. The system demonstrates effective malware detection capability through the combination of low FP and FN rates, which helps to reduce erroneous alerts to a minimum. The classifier shows excellent discrimination power between malware and non-malware samples based on its ROC curve and AUC score, which exceeds 0.98. The RF feature selection method determined network traffic patterns and API call sequences, along with system behavior logs and file metadata properties, to be the key features for malware classification. The identified features operate as essential tools for detecting malware signatures together with abnormal behavioral patterns.

Evaluation of the Malware Detection Framework was conducted through RFC and OvR multi-class classification, as shown in Table 2. The RFC model reached 95% accuracy through 200 estimators and a maximum depth of 20, along with SMOTE data balancing during 5-fold stratified cross-validation. The selected features originated from RF feature importance in combination with PCA. The OvR model optimization included 300 estimators with a maximum depth of 25 and 10-fold cross-validation, along with Recursive Feature Elimination (RFE) for feature selection and Principal Component Analysis. The approach delivered a class-wise malware identification performance at 97% accuracy, which outpaced the competition.

Table 2. Simulation parameters for malware detection framework

Parameter	RFC	OvR
Number of estimators (trees)	200	300
Maximum depth	20	25
Minimum samples split	5	4
Minimum samples per leaf	2	2
Feature selection method	RF feature importance and PCA	PCA and RFE
Data balancing method	SMOTE	SMOTE
Cross-validation method	5-fold stratified cross-validation	10-fold cross-validation
Evaluation metrics	Accuracy, precision, recall, F1-score, confusion matrix, AUC-ROC	Accuracy, precision, recall, F1-score, confusion matrix, AUC-ROC
Accuracy achieved	95%	97%

Note: RFC = Random Forest Classifier; OvR = One-vs-Rest; RF = Random Forest; PCA = Principal Component Analysis; RFE = Recursive Feature Elimination; SMOTE = Synthetic Minority Oversampling Technique; AUC-ROC = Area Under Curve of the Receiver Operating Characteristic.

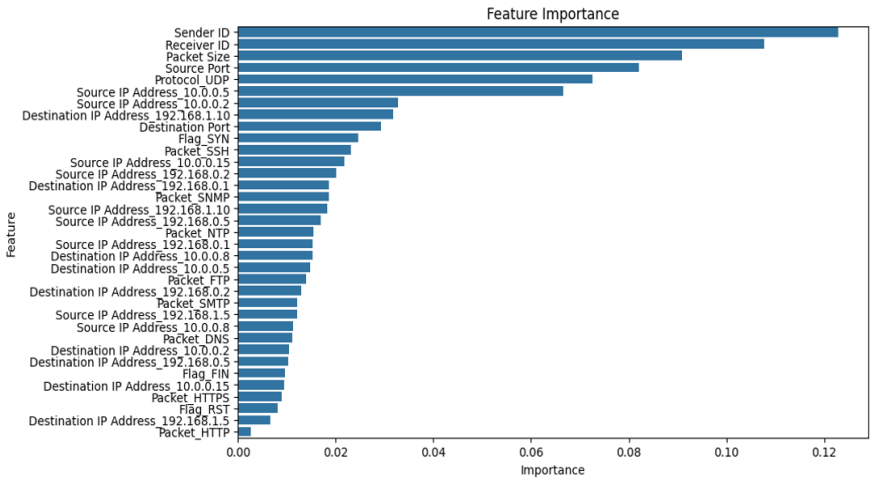

Figure 4. Feature importance analysis for malware detection

The malware detection framework enables users to determine which attributes are most instrumental for decision-making through its feature importance analysis method, as shown in Figure 4. The RFC determines the significance of features to model predictions by assigning importance scores to them. Among all important features with strong influence on malware detection stand the Sender ID and Receiver ID together with Packet Size, Source Port, and Protocol UDP. Network traffic patterns prove significant for accuracy in classification through their powerful role in identifying destructive activities between source and destination IP entities. The lower ranking features Packet HTTP, along with Flag RST and Destination IP 192.168.1.5, deliver context-relevant information, although their role in prediction remains limited. The network-based attributes dominate malware classification based on the feature importance graph because packet metadata and protocol attributes prove effective in cybersecurity threat detection systems. RFC and OvR multi-class classification served to evaluate the malware and ransomware detection framework. Different malware types, including cross-site scripting, DDoS, DoS, man-in-the-middle, and password attacks, phishing, ransomware, SQL injection, and zero-day exploits, underwent an analysis of their performance metrics through precision, recall, and F1-score measurements as shown in Table 3. RF model demonstrated 95% accuracy, but the OvR model reached 97% accuracy when evaluating the performance. Both models successfully perform network activity classification according to precision, recall, and F1-score measurements per class. Detected ransomware

demonstrated the best performance metrics of both precision and recall at 0.95 and 0.92 for RFC and 0.97 and 0.95 for OvR, which proved effective attack classification for sophisticated threats.

Table 3. Performance evaluation of a malware detection framework

Class	Precision (RFC)	Recall (RFC)	F1-Score (RFC)	Support	Precision (OvR)	Recall (OvR)	F1-Score (OvR)
Cross-site scripting	0.85	0.82	0.84	25	0.89	0.86	0.88
DDoS	0.8	0.78	0.79	15	0.87	0.85	0.86
DoS	0.88	0.85	0.86	16	0.91	0.89	0.9
Man-in-the-middle	0.9	0.88	0.89	14	0.92	0.9	0.91
Password attacks	0.87	0.85	0.86	20	0.91	0.89	0.9
Phishing	0.86	0.84	0.85	16	0.9	0.88	0.89
Ransomware	0.95	0.92	0.93	21	0.97	0.95	0.96
SQL injection	0.91	0.89	0.9	24	0.93	0.91	0.92
Zero-day exploits	0.89	0.87	0.88	14	0.91	0.89	0.9
Micro Avg	0.91	0.9	0.91	150	0.94	0.92	0.93
Macro Avg	0.9	0.88	0.89	150	0.93	0.91	0.92
Weighted Avg	0.92	0.9	0.91	150	0.95	0.93	0.94
Samples Avg	0.91	0.89	0.9	150	0.94	0.92	0.93

Note: RFC = Random Forest Classifier; OvR = One-vs-Rest; DDoS = Distributed Denial of Service; DoS = Denial of Service; SQL = Structured Query Language.

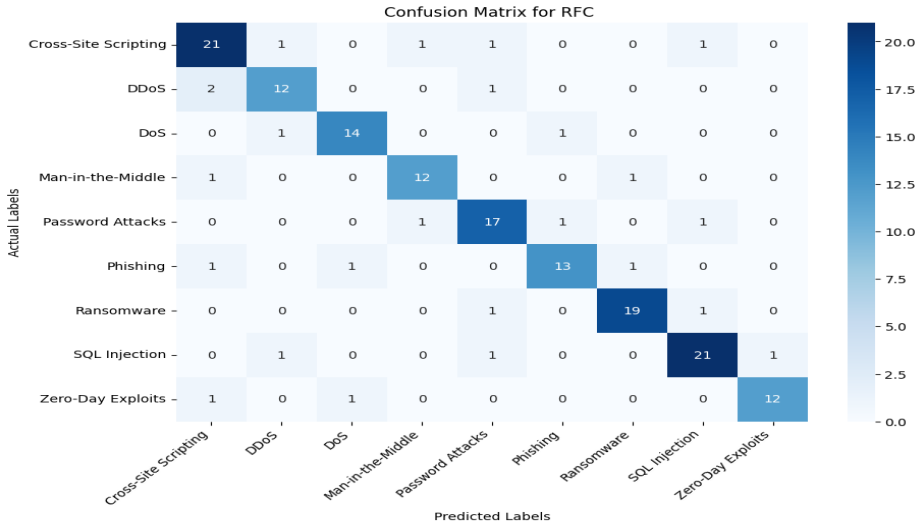

Figure 5. Confusion matrix for Random Forest Classifier (RFC)

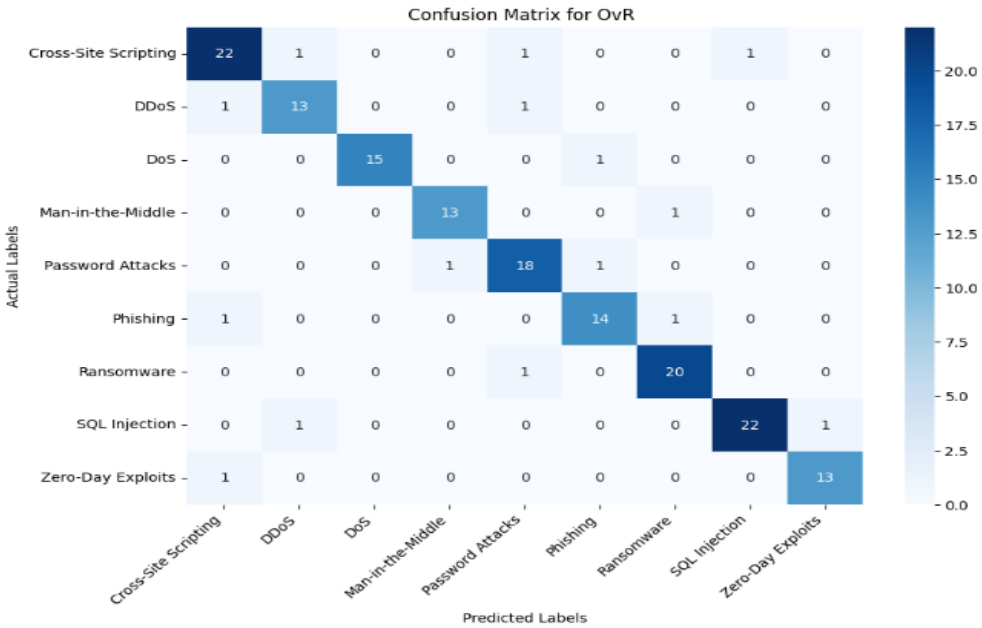

Figure 6. Confusion matrix for One-vs-Rest (OvR) multi-class classification

The RFC confusion matrix confirms how the model performs with accurate malware detection among various cyber threat categories, as shown in Figure 5. The diagonal elements demonstrate model accuracy by showing correct classifications of three particular threats: Cross-site scripting (21), SQL injection (21), and ransomware (19). Minor classification mistakes occur in the system when DDoS attacks get mistaken for cross-site scripting, while zero-day exploits wrongly match phishing instances, which indicates room for future enhancement. This model reduces the occurrence of false positives and false negatives, which enables it to establish reliable endpoint security through high precision and recall measures, delivering overall accuracy at 95% for the detection and mitigation of malware threats in cybersecurity applications. In Figure 6, the OvR multi-class classification model shows a 97% accuracy rate according to its confusion matrix, which validates its remarkable capability to detect malware.

The analysis showed that cross-site scripting (22), SQL injection (22), and ransomware (20) attack types received correct classifications while maintaining low numbers of wrong assignments. The false positive and false negative results of this model show better performance than the RFC indicators, demonstrating an enhanced classification precision level. The minor errors detected in DDoS and zero-day exploits detection show areas where the tool needs improved accuracy. The experimental results demonstrate that OvR demonstrates a strong capability to identify diverse malware specimens, thus proving it can succeed in practical cybersecurity systems. The confusion matrices (Figures 5 and 6) indicate that some of the classes of attacks have a partial overlap in feature space, resulting in minimal misclassification. Specifically, certain examples of DDoS are mixed up with cross-site scripting, which can be explained by the similarity of traffic features when under attack, like the large request rates and unusual packet patterns. Likewise, there might be regular confusion between zero-day attacks and phishing attacks, as both are poorly represented in terms of signature and are more based on a behavioral outlier instead of a protocol violation. The existence of these overlaps notwithstanding, the OvR framework enhances the ability to discriminate between classes by learning class-specific decision boundaries and hence performs better than the standalone RF model.

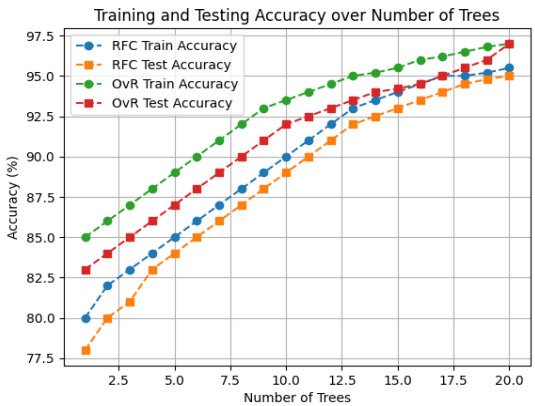

Figure 7. Cross-validation training and validation accuracy for Random Forest (RF) and One-vs-Rest (OvR) models

The results in Figure 7 are used to show the training and validation accuracy of the RF and OvR classifiers using cross-validation. The averaged values of accuracy are obtained across stratified cross-validation folds with an increase in the number of estimators, but not training epochs. RF model converges at about 95% accuracy, whereas the OvR framework is at a high validation accuracy of up to 97%, which indicates a better generalization. The blue RFC model achieves 80% accuracy during its commencement, before achieving 95% training and testing accuracy. The OvR model displayed better generalization than the other models, with its 97% accuracy results seen in red and orange. The accuracy values demonstrate the successful malware classification capabilities of both systems throughout the increasing number of epochs. The OvR classification method achieves better results in test accuracy measurements since it optimizes overfitting prevention while preserving high detection abilities.

Table 4 provides a comparative study of the suggested malware detection models with reference to some of the popular ML algorithms. Classical classifiers such as DT, naïve bayes, k-nearest neighbours (KNN), SVM, and XGBoost have moderate levels of accuracy with a range of between 82% and 92%, with the corresponding precision, recall, and F1-scores highlighting their low ability to differentiate between more complex and dynamic malware patterns. Conversely, the accuracy of the proposed RF model is much higher, 95%, which proves to be more accurate in generalisation and discrimination of the features. The OvR classifier further improves the performance, with an accuracy of 97% and the best precision, recall, and F1-score of all the models used in the study. Such high performance highlights the effectiveness of an ensemble learning coupled with a multi-class classification, which validates the fact that the proposed framework provides a more trustworthy and stronger malware detection solution than the current machine-learning methods. Although the performance is high, the framework has weaknesses: it operates on a curated publicly available dataset, which might not be representative of real-world variability; there might be a risk of feature leakage due to network and protocol characteristics, and adversarial robustness and online adaptation are not addressed. Future research must be able to resolve these problems with larger and constantly updated datasets, adversarial testing, and incremental learning.

Table 4. Comparison of the proposed framework with existing ML techniques

Model	Accuracy	Precision	Recall	F1-Score
Decision tree	86%	0.84	0.82	0.83
Naïve bayes	82%	0.80	0.78	0.79
KNN	88%	0.85	0.84	0.84
SVM (RBF Kernel)	90%	0.88	0.87	0.87
XGBoost	92%	0.90	0.89	0.89
RF (proposed)	95%	0.91	0.90	0.91
OvR (Proposed)	97%	0.94	0.92	0.93

Note: ML = machine learning; KNN = k-nearest neighbours; SVM = support vector machine; RBF = radial basis function; RF = Random Forest; OvR = One-vs-Rest.

5. CONCLUSIONS

This paper introduces an ML-based system of detection of malware and ransomware, attacking a model that is based on an RF as well as an OvR multi-class approach. The proposed

pipeline includes features of SMOTE-based balancing of the classes, analysis of feature importance, and dimensionality reduction through PCA, which also helps to solve such typical issues as the imbalance of the classes, the redundancy of the features, or inter-class confusion. As illustrated by experimental findings that were made with the help of stratified cross-validation, it is stated that the RF model reaches an accuracy of 95%, and the OvR framework makes the model even more accurate (97%), with very high values of precision, recall, F1-score, and AUC-ROC. Confusion matrix analysis shows that the OvR strategy increases the separability of classes and reduces the misclassification of the highly related types of attacks in comparison to the standalone RF model. These findings validate the claim that, in endpoint security, ensemble learning and the use of explicit multi-class discrimination give a robust and understandable method of malware and ransomware classification. Although the given implementation is targeted at offline assessment with a set of curated public data, the framework allows extending to larger datasets, incremental learning, and analysis of robustness in adversarial settings. This study did not assess real-time deployment, dynamically updated threat intelligence, and adversarial resilience, but these directions are still significant to work on in the future. On the whole, the results support the efficiency of the RF + OvR pipeline as a feasible point of departure to multi-class malware recognition and encourage further research on adaptive and resilient cybersecurity systems.

REFERENCES

[1] Patel, N., Ukani, V., Mistry, N. (2014). A novel methodology for analyzing malicious behavior of the android applications. International Journal of Advanced Research in Engineering and Technology (IJARET), 5(4): 213-221. https://iaeme.com/Home/article_id/IJARET_05_04_023.

[2] Ahmed, M.E., Kim, H., Camtepe, S., Nepal, S. (2021). Peeler: Profiling kernel-level events to detect ransomware. In European Symposium on Research in Computer Security, pp. 240-260. https://doi.org/10.1007/978-3-030-88418-5_12

[3] Hirano, M., Kobayashi, R. (2022). Machine learning-based ransomware detection using low-level memory access patterns obtained from live-forensic hypervisor. In 2022 IEEE International Conference on Cyber Security and Resilience (CSR), Rhodes, Greece, pp. 323-330. https://doi.org/10.1109/CSR54599.2022.9850340

[4] Kunku, K., Zaman, A.N.K., Roy, K. (2023). Ransomware detection and classification using machine learning. In 2023 IEEE Symposium Series on Computational Intelligence (SSCI), Mexico City, Mexico, pp. 862-866. https://doi.org/10.1109/SSCI52147.2023.10371924

[5] Rhode, M., Burnap, P., Jones, K. (2019). Dual-task agent for run-time classification and killing of malicious processes. https://doi.org/10.48550/arXiv.1902.02598

[6] Sambhe, N.U., Prasad, P. (2025). Comprehensive review on insider threat detection approaches. In Proceedings of 5th International Conference on Artificial Intelligence and Smart Energy, pp. 302-310. https://doi.org/10.1007/978-3-031-90482-0_24

[7] Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C., Weiss, Y. (2012). "Andromaly": A behavioral malware detection framework for android devices. Journal of Intelligent Information Systems, 38(1): 161-190. https://doi.org/10.1007/s10844-010-0148-x

[8] Zilberman, P., Puzis, R., Elovici, Y. (2015). On network footprint of traffic inspection and filtering at global scrubbing centers. IEEE Transactions on Dependable and Secure Computing, 14(5): 521-534. https://doi.org/10.1109/TDSC.2015.2494039

[9] Arabo, A., Dijoux, R., Poulain, T., Chevalier, G. (2020). Detecting ransomware using process behavior analysis. Procedia Computer Science, 168: 289-296. https://doi.org/10.1016/j.procs.2020.02.249

[10] Muhammad, Z., Anwar, Z., Javed, A.R., Saleem, B., Abbas, S., Gadekallu, T.R. (2023). Smartphone security and privacy: A survey on APTs, sensor-based attacks, side-channel attacks, Google play attacks, and defenses. Technologies, 11(3): 76. https://doi.org/10.3390/technologies11030076

[11] Sarker, I.H., Abushark, Y.B., Alsolami, F., Khan, A.I. (2020). IntruDTree: A machine learning based cyber security intrusion detection model. Symmetry, 12(5): 754. https://doi.org/10.3390/sym12050754

[12] Roseline, S.A., Geetha, S. (2018). Intelligent malware detection using oblique random forest paradigm. In 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, pp. 330-336. https://doi.org/10.1109/ICACCI.2018.8554903

[13] Zhang, Z.Y., Xiao, G.R., Song, S.C., Aygun, R.C., Stavrou, A., Zhang, L.X., Osterweil, E. (2024). Revealing the architectural design patterns in the volumetric DDoS defense design space. TechRxiv. https://doi.org/10.36227/techrxiv.23245082.v2

[14] Bai, H., Liu, G., Liu, W., Quan, Y., Huang, S. (2021). N-gram, semantic-based neural network for mobile malware network traffic detection. Security and Communication Networks, 2021(1): 5599556. https://doi.org/10.1155/2021/5599556

[15] Berger, H., Hajaj, C., Mariconti, E., Dvir, A. (2021). Crystal ball: From innovative attacks to attack effectiveness classifier. IEEE Access, 10: 1317-1333. https://doi.org/10.1109/ACCESS.2021.3138628

[16] Mat, S.R.T., Ab Razak, M.F., Kahar, M.N.M., Arif, J.M., Mohamad, S., Firdaus, A. (2021). Towards a systematic description of the field using bibliometric analysis: Malware evolution. Scientometrics, 126(3): 2013-2055. https://doi.org/10.1007/s11192-020-03834-6

[17] Jabbarnejad, A., Vaez-Zadeh, S., Eslahi, M.S. (2024). An analytical parameter-free cyberattack detection method for grid-connected converters. IEEE Transactions on Power Electronics, 39(12): 15770-15784. https://doi.org/10.1109/TPEL.2024.3430542

[18] Girirajan, S., Ramesh, C., Saraswat, M., Kochher, V., Kodge, B.G., Brajesh, K., Raj, S. (2025). A deep learning approach for android malware detection using mixed bytecode images and attention-based ResNet. Journal of Information Systems Engineering & Management. 10(2s): 298-316. https://doi.org/10.52783/jisem.v10i2s.319

[19] Akshay Kumar, E., Rangasamy, J. (2023). Light-weight

deep learning models for visual malware classification. In Advances in Signal Processing, Embedded Systems and IoT: Proceedings of Seventh ICMEET-2022, pp. 485-495. https://doi.org/10.1007/978-981-19-8865-3_44

[20] Mohi ud din, S., Rizvi, F., Sharma, N., Sharma, D.K. (2022). Comparative analysis of various machine learning algorithms for detection of malware and benign. In International Advanced Computing Conference, pp. 212-218. https://doi.org/10.1007/978-3-031-35641-4_17

[21] Ma, S., Liu, W. (2023). Classification model of packed malware with attention mechanism. In International Conference on Computer Network Security and Software Engineering (CNSSE 2023), pp. 238-246. https://doi.org/10.1117/12.2683184

[22] Brown, A., Gupta, M., Abdelsalam, M. (2024). Automated machine learning for deep learning based malware detection. Computers & Security, 137: 103582. https://doi.org/10.1016/j.cose.2023.103582

[23] Alraizza, A., Algarni, A. (2023). Ransomware detection using machine learning: A survey. Big Data and Cognitive Computing, 7(3): 143. https://doi.org/10.3390/bdcc7030143

[24] Singh, J., Singh, J. (2021). A survey on machine learning-based malware detection in executable files. Journal of Systems Architecture, 112: 101861. https://doi.org/10.1016/j.sysarc.2020.101861

[25] Raghavendra, R., Dutta, M.V. (2023). Machine learning in malware detection: A survey of analysis techniques. International Journal of Advanced Research in Computer and Communication Engineering, 12(4): 204-210. https://doi.org/10.17148/IJARCCE.2023.12435

[26] Mohammed, Z.Q., Abooddy, C.H., Zaidan, O.H. (2025). Integrated framework for real-time cyber threat detection and mitigation in IoT and healthcare systems using deep learning and blockchain. International Journal of Safety & Security Engineering, 15(8): 1611-1625. https://doi.org/10.18280/ijsse.150807

[27] Gorment, N.Z., Selamat, A., Cheng, L.K., Krejcar, O. (2023). Machine learning algorithm for malware detection: Taxonomy, current challenges, and future directions. IEEE Access, 11: 141045-141089. https://doi.org/10.1109/ACCESS.2023.3256979

[28] Atmojo, Y.P., Susila, I., Hariyanti, E., Hostiadi, D.P., Pradipta, G.A., Ayu, P.D.W. (2025). Network anomaly activity detection model based on feature correlation analysis. International Journal of Safety & Security Engineering, 15(9): 1809-1817. https://doi.org/10.18280/ijsse.150905

[29] Loco, P., Alonso, S., Hartmann, G., Whitmore, J., McLaughlin, E. (2024). Adaptive behavior-based ransomware detection via dynamic flow signatures. https://doi.org/10.21203/rs.3.rs-5317374/v1