ISSN 2277-2685
IJESR/April. 2026/ Vol-16/Issue-2s1/273-278
Maimoona Khanam *et. al.,* / International Journal of Engineering & Science Research
Full Length Article

Dynamic Ransomware Detection Using Time-Based API Call Analysis

Maimoona Khanam[1], Alkaan Kausar[2], Syed Abdul wahab Asif[3]
[1,2]B.E.Students; Department of AI&DS(Artificial intelligence and data science) ISL Engineering College Hyderabad India.
[3]Assistant Professor; Department of AI&DS (Artificial intelligence and data science) ISL Engineering College Hyderabad India.
Mail Id;160522747007@islec.edu.in, 160522747003@islec.edu.in
Accepted 27-04-2026

Abstract
The increasing growth of cyber threats has made malware detection an important challenge in cybersecurity. This project proposes a machine learning-based multi-class malware detection system capable of identifying malware types such as ransomware, adware, keyloggers, rootkits, and botnets. The system uses preprocessed and normalized feature-based data to train models including Random Forest, XGBoost, and LSTM. Comparative analysis shows that XGBoost provides the highest classification accuracy and reliability. A Flask-based web application is developed to provide real-time malware prediction along with preventive measures. The proposed system demonstrates the effectiveness of machine learning, especially ensemble techniques, in enhancing malware detection and cybersecurity protection.

Keywords
Malware Detection, Machine Learning, XGBoost, Random Forest, LSTM, Cybersecurity, Multi-Class Classification, Flask Web Application.

Introduction

The rapid growth of digital technologies and internet usage has increased the risk of malware attacks, threatening data security and system integrity. Traditional signature-based methods are often ineffective against new and evolving malware variants. To address this challenge, this project develops a multi-class malware detection system using machine learning techniques. The system classifies various malware types based on extracted features after data preprocessing and normalization. Models such as Random Forest, XGBoost, and LSTM are implemented and compared, with XGBoost achieving the best performance due to its high accuracy and scalability. The final model is integrated into a Flask-based web application that provides real-time malware prediction along with information and preventive measures, offering a practical solution for modern cybersecurity needs.

Literature Review

Recent research shows that machine learning is highly effective for ransomware and malware detection. Mehdi Roohi (2023) and Chengyu Song (2023) demonstrated that API call sequences and behavioral features can accurately identify ransomware, even when the code is obfuscated. Abhishek Singh (2024) showed that temporal features such as API call intervals improve detection performance. Natalia Zuba (2024) found that Random Forest provides high accuracy with faster execution compared to deep learning models. Hassan Al-Mohannadi (2023) proposed a real-time API monitoring framework for early ransomware detection. These studies confirm that machine learning-based behavioral analysis is a reliable approach for modern malware detection.

Methodology

Module Names

User Authentication Module
Data Preprocessing and Feature Extraction Module
Model Training and Prediction Module
Flask-Based Web Interface Module
Ransomware Prediction Module
Result Display Module

Module Description

User Authentication Module:
Handles user registration, login, session management, and logout to ensure secure access to the system.

Data Preprocessing and Feature Extraction Module:
Cleans the dataset, handles missing values, and applies feature scaling to prepare data for model training and prediction.

Model Training and Prediction Module:
Trains machine learning models such as Random Forest, XGBoost, and LSTM, selects the best model, and performs predictions.

Flask-Based Web Interface Module:
Provides a user-friendly web interface for registration, login, input submission, and result viewing.

Ransomware Prediction Module:
Uses the trained model to classify whether the input corresponds to ransomware or other malware types.

Result Display Module:
Displays the predicted malware type along with its description, impact, and preventive measures.

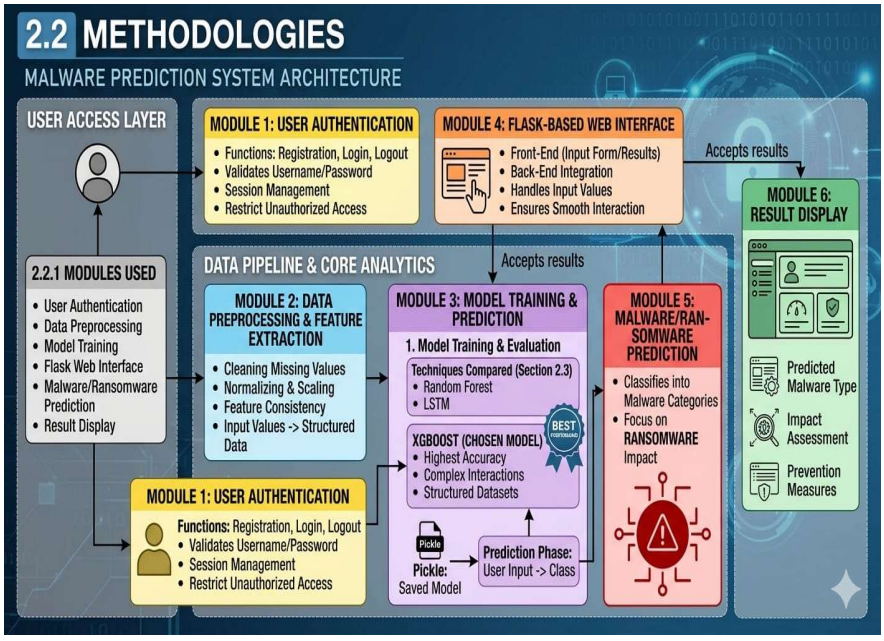
Technique Used or Algorithm Used

Existing Technique

Traditional malware detection methods include signature-based and heuristic-based techniques. Signature-based detection identifies known malware by matching files with stored signatures, while heuristic-based detection analyzes suspicious behavior patterns. These methods are effective for known threats but struggle to detect new and evolving malware variants.

Proposed Technique

This project uses machine learning algorithms such as Random Forest, XGBoost, and LSTM for malware classification. Random Forest improves accuracy by combining multiple decision trees, XGBoost enhances performance through boosting, and LSTM captures sequential patterns in data. Among these, XGBoost achieved the highest accuracy and was selected as the final model for the system.



Implementation

Algorithm: XGBoost-Based Multi-Class Malware Detection

Algorithm: Multi-Class Malware Detection Using XGBoost

Input: Malware dataset containing extracted features and corresponding malware class labels.

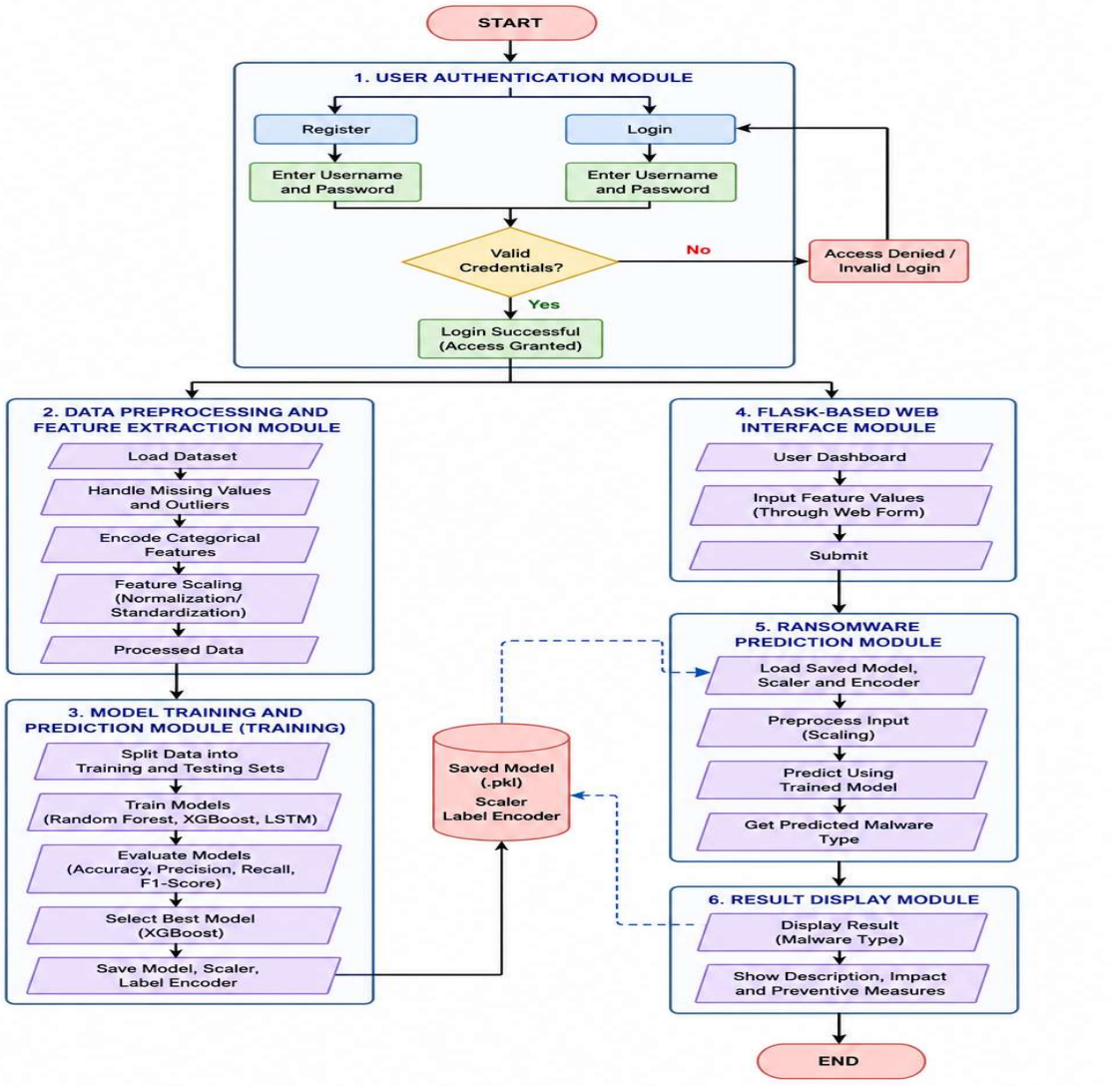
Output: Predicted malware type with description and preventive measures.

Training Phase

1. Load the malware dataset.
2. Handle missing values and clean the data.
3. Separate input features (**X**) and target labels (**y**).
4. Encode malware labels using LabelEncoder.
5. Normalize feature values using MinMaxScaler.
6. Split the dataset into training and testing sets.
7. Train three models: Random Forest, XGBoost, and LSTM.
8. Evaluate each model using Accuracy, Precision, Recall, and F1-Score.
9. Select the model with the highest accuracy (XGBoost).
10. Save the trained XGBoost model, scaler, and label encoder.

Prediction Phase

11. User logs into the Flask web application.
12. User enters feature values through the input form.
13. Load the saved model, scaler, and label encoder.
14. Preprocess the input using the same scaler.
15. Predict the malware class using XGBoost.
16. Convert the predicted label to the original malware name.
17. Display the malware type, description, impact, and preventive measures.



TESTING

Software testing is performed to identify errors and verify that the machine learning-based multi-class malware detection system functions correctly. It ensures that the application meets user requirements, produces accurate predictions, and operates reliably without failures.

Developing Methodologies

A structured testing plan was prepared to validate all modules of the system, including data preprocessing, model prediction, user authentication, and the Flask web interface. Testing was conducted under strict quality control to ensure that the system is bug-free and meets the specified requirements.

Types of Tests

Unit Testing

Each module, such as preprocessing, model loading, and prediction, was tested individually to verify that it produces the expected output.

Functional Testing

This testing ensured that all functions, including login, feature input, prediction, and result display, worked according to the requirements.

System Testing

The complete integrated system was tested to confirm that all components work together correctly and provide accurate results.

Performance Testing

The system was evaluated for prediction speed, response time, and memory usage. The XGBoost model provided fast and efficient predictions.

Integration Testing

This testing verified smooth interaction between the machine learning model, Flask backend, and user interface.

Acceptance Testing

End users tested the application to ensure that the interface was easy to use and that the predictions were understandable and reliable.

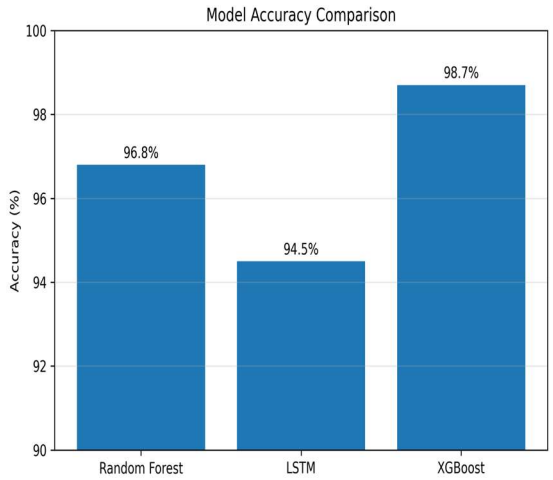
Build the Test Plan

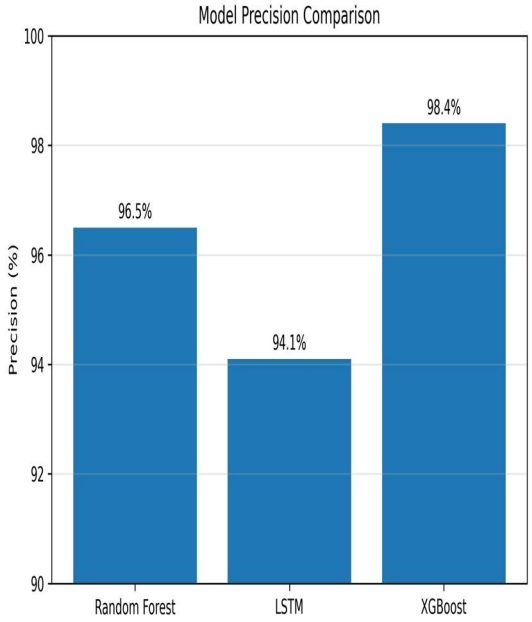
A comprehensive test plan was developed by dividing the project into individual modules and testing each module separately before integrating them. This approach helped identify and correct errors efficiently, ensuring that the final system operates accurately and reliably.

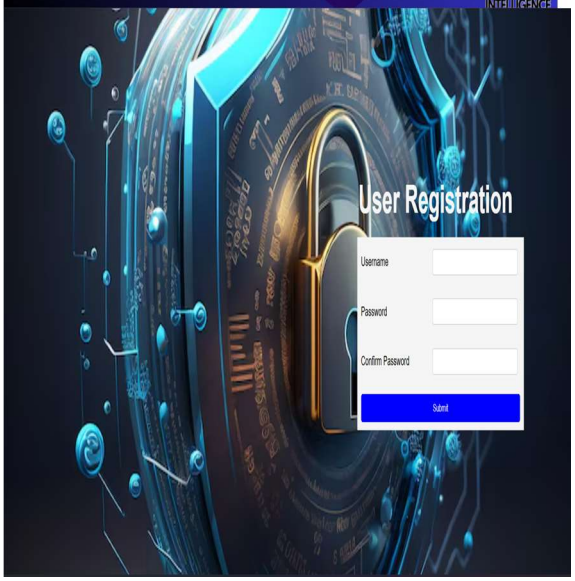
RESULTS

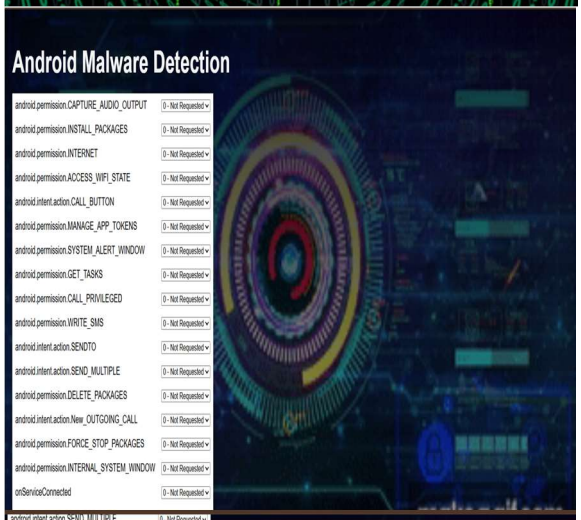
The Machine Learning-Based Multi-Class Malware Detection System was successfully developed using Python, Flask, and machine learning algorithms such as Random Forest, XGBoost, and TensorFlow LSTM. After preprocessing the dataset and training the models, XGBoost achieved the highest accuracy and was selected as the final model.

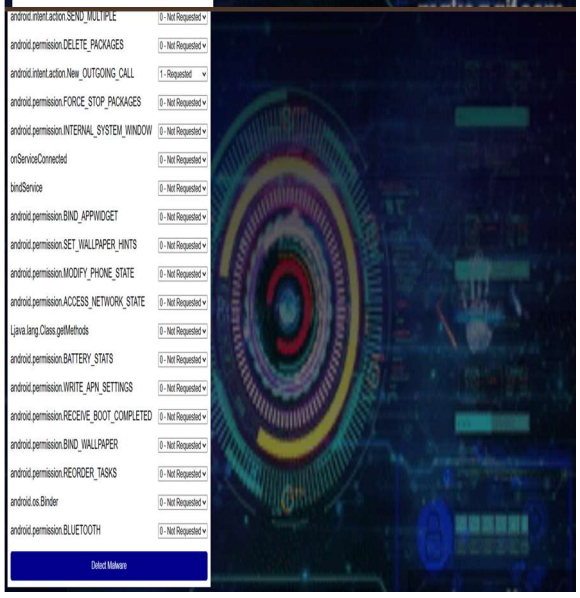
The model was integrated into a Flask web application that allows users to log in, enter feature values, and receive real-time malware predictions along with descriptions and preventive measures. Testing confirmed that the system provides accurate, fast, and reliable results. Overall, the project demonstrates that XGBoost is an effective and practical approach for detecting ransomware and other malware types.











Conclusion

This project successfully developed a machine learning-based multi-class malware detection system capable of identifying malware types such as ransomware, adware, keyloggers, rootkits, and botnets. Multiple algorithms, including Random Forest, XGBoost, and LSTM, were implemented and compared, with XGBoost achieving the highest accuracy and best overall performance.

The selected model was integrated into a Flask web application that provides real-time predictions along with malware descriptions and preventive measures. The system is accurate, user-friendly, and scalable, demonstrating the effectiveness of machine learning in modern cybersecurity and providing a practical solution for malware detection.

Future Scope

The proposed malware detection system can be enhanced by incorporating real-time analysis of system logs, API calls, and network traffic to improve detection accuracy. Advanced deep learning models such as LSTM and Transformer-based architectures can be used to identify more complex and evolving malware patterns. The system can also be deployed in cloud environments for better scalability and integrated with automated alert mechanisms to notify users of potential threats.

Additional improvements include storing prediction history in a database, providing interactive dashboards, and supporting mobile access. Continuous model retraining with updated datasets and stronger security features such as encryption and secure authentication will further improve the system.

These enhancements will make the application more robust, scalable, and suitable for real-world cybersecurity deployment.

References

[1] P. O'Kane, S. Sezer, and D. Carlin, ''Evolution of ransomware,'' IET Netw.,vol. 7, no. 5, pp. 321–327, Jun. 2018.
[2] G. O. Gorman and G. McDonald, ''Ransomware : A growing menace,''Symantec, vol. 1, p. 16, Aug. 2012.
[3] H. Tuttle, ''Ransomware attacks pose growing threat,'' Risk Manage.,vol. 63, no. 4, pp. 4–7, 2016.
[4] Threat of Ransomware Remains at Peak With Half of Organizations FallingVictim in the Last Year, Athena Inf. Solutions Pvt. Ltd, India, New Delhi,2023, pp. 1–4.
[5] P. Chakraborty, ''Ransomware remains major threat as Sophos reports stateof cyber security in 2023,'' Gurgaon Athena Information Solutions Pvt.Ltd, India, Tech. Rep., 2023, pp. 2023–2024.
[6] M. Sunidhi, ''Elastic global threat report 2023 reveals dominance ofransomware,'' Athena Information Solutions Pvt. Ltd, India, Mumbai,Tech. Rep., 2023, pp. 3–5.
[7] CISO Research Reveals 90 % of Organisations Suffered at Least One Major Cyber Attack in the Last Year; 83 % Report Ransomware Payments, Athena Information Solutions Pvt. Ltd, India, Mumbai, 2023, pp. 1–3.
[8] J. Porter, ''Wolverine part of massive insomniac games leak afterransomware deadline passes,'' Verge New York City, USA, Tech. Rep.,2023.
[9] B. Yamany, M. S. Elsayed, A. D. Jurcut, N. Abdelbaki, and M. A. Azer,''A holistic approach to ransomware classification: Leveraging static and dynamic analysis with visualization,'' Information, vol. 15, no. 1, p. 46,Jan. 2024.
[10] Y. Wang, Z. Li, and Y. Zhang, ''Optimized ransomware detection through reverse Bayer analysis of file system activities,'' OSF Preprints, 2024.
[11] C. Beaman, A. Barkworth, T. D. Akande, S. Hakak, and M. K. Khan, ''Ransomware: Recent advances, analysis, challenges and future research directions,'' Comput. Secur., vol. 111, Dec. 2021, Art. no. 102490.
[12] H. N. Nguyen, F. Abri, V. Pham, M. Chatterjee, A. S. Namin, and T. Dang, ''MalView: Interactive visual analytics for comprehending malware behavior,'' IEEE Access, vol. 10, pp. 99909–99930, 2022.