International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 22 No. 3 (2026)

ijerst.editor@gmail.com

editor@ijerst.com

ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

DYNAMIC RANSOMWARE DETECTION USING TIME-BASED API CALLING ANALYSIS

Juveriya Rasheed[1], Dr. Umar Farooq[2]
[1]PG Scholar, Department of AI&DS, Shadan Women's College of Engineering and Technology, Hyderabad, juveriyarasheed2001@gmail.com
[2]Assoc. Professor, Department of CSE, Shadan Women's College of Engineering and Technology Umarfarooq.mohd@gmail.com

ABSTRACT
Numerous malware varieties, such as ransomware, adware, keyloggers, rootkits, and botnets, have emerged as a result of the quick evolution of cyberthreats and present serious risks to digital systems. The machine learning-based multi-class malware detection system presented in this research is intended to correctly categorise various kinds of harmful software. Before being fed into many machine learning models, such as Random Forest, XGBoost, and Long Short-Term Memory (LSTM), the system uses a feature-based dataset that has been preprocessed and normalised. To assess these models' performance in terms of classification accuracy and dependability, a comparison analysis is carried out. Because XGBoost can efficiently capture complicated feature interactions and minimise overfitting, it was the model with the best accuracy. A Flask-based web application is used to implement the suggested system, allowing users to enter feature values and obtain real-time forecasts for each form of malware along with thorough explanations and countermeasures. The findings show that ensemble learning techniques, especially XGBoost, are very successful in classifying multi-class malware and can be used in practical cybersecurity systems

1. INTRODUCTION

Modern computing systems have greatly benefited from the quick development of digital technology and the rise in internet traffic, but they have also created substantial cybersecurity risks. Malware, or malicious software intended to interfere with, harm, or obtain unauthorised access to systems, has grown more complex and pervasive. Data security, user privacy, and system integrity are seriously threatened by a variety of malware types, including ransomware, adware, keyloggers, rootkits, and botnets. Intelligent and adaptive detection systems are crucial since traditional signature-based detection techniques are no longer adequate to identify newly emerging and developing malware strains. Because machine learning algorithms can recognise hidden risks and understand patterns from data, they have become more prominent in cybersecurity in recent years. The main goal of this project is to apply machine learning methods to create a multi-class malware detection system. Based on attributes that were retrieved from the dataset, the system is intended to categorise various forms of malware. To improve the accuracy and performance of the model, data preprocessing and normalisation techniques are used. A number of models, including as Random Forest, XGBoost, and Long Short-Term Memory (LSTM), are put into practice and contrasted to assess how well they classify malware. Because of its scalability, robustness, and capacity to effectively manage intricate feature interactions, XGBoost has demonstrated outstanding performance among them. The technology is further incorporated into an easy-to-use Flask-based web application that allows for real-time prediction and offers comprehensive details about the sorts of malware that have been found, along with preventative measures. In addition to increasing detection accuracy, our method offers a scalable and useful solution for practical cybersecurity applications, assisting users and organisations in defending their systems against a variety of malware threats.

2. EXISTING SYSTEM OF PROJECT

To identify known threats, traditional malware detection systems mostly use signature-based methods. To identify harmful activity, these systems check programs and files against a database of known malware signatures. This method works well for malware that has already been detected, but it is unable to identify novel and unidentified variants, particularly those that employ obfuscation and polymorphism tactics. Attackers constantly alter malware signatures as cyberthreats change quickly, making it challenging for conventional systems to stay up. Furthermore, signature-based approaches necessitate regular database updates, which may cause delays in identifying new threats. Heuristic and rule-based detection, which looks for suspicious activity based on predetermined rules, is another popular method. This approach frequently produces large false

positive rates and lacks adaptability, even though it can identify some unknown dangers. These systems' capacity to learn from fresh data is constrained, and they are unable to generalise to previously unidentified malware patterns. Additionally, the majority of conventional systems are unable to distinguish between different kinds of malware since they do not enable multi-class classification. Because of this, they offer little information about the attack's nature, which makes it more difficult to put the right preventive measures in place.

3. RELATED WORK

Numerous cutting-edge methods and systems for malware analysis have been created [12], [14], [16], [17], and [28]. In order to understand the behaviour of malware in general and ransomware in particular, research has been done to apply current methods and create new ones. However, in order to keep up with the complexity of analysis and the quickly changing nature of ransomware, malware analysis needs be continuously improved.

To assist analysts in the malware analysis process, research has been done on creating malware visual analytics systems [12], [16], and [17]. MalView, an interactive visualisation system that employs pattern-matching approaches for hybrid analysis of malware, was created by Nguyen et al. [12].MalView concentrates on the domain names that interact with the sample, Dynamic Linked Libraries (DLL) calls, and process activity. MalView features a network to show the dependencies between processes, a library matrix to show the DLL calls for each process, and a time-series process visualisation. The goal of each of these visualisations is to assist the analyst in identifying anomalies in a sample's process frequency as well as malicious process signatures. In order to ascertain whether the domain names that interacted with the sample are malicious or benign, MalView employs VirusTotal [29].

Instead of using internal analysis, the system makes use of external analysis like VirusTotal and IPstack [30]. Because the system depends on other systems to improve their analytic methodologies to handle new variants, this is a disadvantage that could lessen the system's efficacy against new hostile domains. The purpose of Eventpad is to enable analysts to choose their desired characteristics and trends. They used tshark to capture packets and build up a virtual network [31]. Eventpad was used to turn the recorded network activity into blocks. An analyst may visually encode the blocks and choose the relevant attributes once they were captured. To identify malicious behaviour and patterns, the analyst use both visual encoding and pattern matching. Eventpad's efficacy as a malware analysis system is limited because it does not employ a network activity dataset to train a model to categorise dangerous samples or behaviour. A visualisation system named KAMAS, which focuses on malware behaviour analysis, was developed by Wagner et al. [16].

4. PROPOSED SYSTEM

The suggested solution overcomes the drawbacks of conventional signature-based techniques by introducing a machine learning-based strategy for multi-class malware identification. It makes use of a structured dataset from which attributes reflecting various malware characteristics have been retrieved. To guarantee consistency and enhance model performance, the data is preprocessed and normalised. Several machine learning methods are used to categorise different kinds of malware, including ransomware, adware, keyloggers, rootkits, and botnets. These algorithms include Random Forest, XGBoost, and Long Short-Term Memory (LSTM). To determine the best method for precise categorisation, these models are trained and assessed. Because XGBoost can manage complicated feature interactions and avoid overfitting through boosting approaches, it performs better than the other implemented models. The system can successfully differentiate between various types of malware since it allows multi-class categorisation. Furthermore, a Flask-based web application incorporates the suggested approach, enabling users to enter feature values and obtain predictions in real time. Additionally, it offers comprehensive details regarding the malware that has been found, including its effects and countermeasures. This method improves usability, scalability, and accuracy, which makes it appropriate for real-world cybersecurity applications.

5. MODULES

1. User Authentication Module:

The system's user registration and login processes are managed by the User Authentication Module. It guarantees validation by looking for duplicate users and matching credentials, and it enables new users to create an account by entering their username and password. In order to restrict access to secured pages, the system maintains session management while confirming user credentials during login. The prediction interface cannot be used by unauthorised users. In order to securely end user sessions, the module also has logout capabilities. This guarantees restricted access to the application and basic security.

2. Data Preprocessing and Feature Extraction Module:

The preparation of the dataset for efficient model training and prediction is the main goal of this module. It entails addressing missing values, cleansing the data, and transforming inputs into a standard numerical representation. To enhance model performance, a scaler

is used to apply feature scaling approaches like normalisation or standardisation. Based on predetermined feature names, the system makes sure that input features are sorted in the proper order. The accuracy of machine learning models can be increased and noise can be reduced with proper preprocessing. This module is essential to preserving the consistency and quality of the data.

3. Model Training and Prediction Module:

Machine learning models are created and assessed via the Model Training and Prediction Module. It entails using the preprocessed dataset to train several models, including Random Forest, XGBoost, and LSTM. To determine which model is the most accurate, each model is assessed using performance metrics. The best-performing model is chosen and stored for later use after training is finished. In order to produce predictions, user-provided inputs are analysed and run through the trained model. This module guarantees effective and precise classification of malware.

4. Flask-Based Web Interface Module:

This module offers an easy-to-use web interface for system interaction. It was created with Flask and makes it simple for users to browse the signup, login, and prediction pages. Users' input feature values are gathered by the interface and forwarded to the backend for processing. Depending on what the user does, it dynamically shows messages and results. To create visually appealing and responsive pages, templates are utilised. This module makes the system interactive and accessible by bridging the gap between the end user and the machine learning model.

5. Ransomware Prediction Module:

Finding out if the input data is indicative of ransomware or other malware is the main goal of the Ransomware Prediction Module. The input is categorised into pre-established malware groups using the trained machine learning model. This module focuses on identifying ransomware because of its significant danger and impact, even though the system supports a variety of malware kinds. It determines the probability of ransomware presence by analysing input features. The module aids in the prevention of data encryption assaults and early detection. It is essential for improving system security.

6. Result Display Module:

The prediction results are presented in an understandable and educational way via the Result Display Module. Once the input has been processed, the system shows the type of malware that was found along with the classification label that goes with it. For improved user comprehension, it also offers extra information including description, impact, and preventative measures. The outcomes are displayed on a special webpage with appropriate formatting and an easy-to-use layout. This module increases user awareness and aids in responding appropriately to risks that are identified. It guarantees that the results are easy to understand and relevant.

6. TECHNIQUES USED IN PROJECT

6.1 EXISTING TECHNIQUE: -

One of the most popular conventional methods for detecting malware is signature-based detection. It compares programs and files to a database of known malware signatures. The system labels the file as harmful if a match is discovered. This method is frequently employed in antivirus software and is quite successful in identifying dangers that have already been identified. However, in order to continue being successful against new malware strains, the signature database must be updated on a regular basis.

Another method that looks for unknown viruses by examining suspicious behaviour patterns is heuristic-based detection. It use predetermined criteria and characteristics to identify potentially dangerous activity rather than depending on known signs. Although this approach can identify new and altered malware strains, it frequently has large false positive rates. It is also less effective against quickly changing cyberthreats since it cannot learn from fresh data.

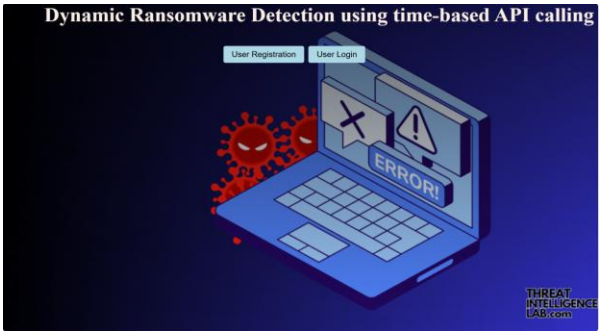
6.2 ALGORITHM USED:

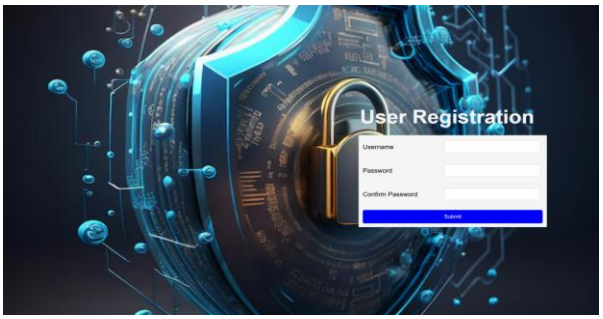
In order to increase classification accuracy, Random Forest, an ensemble machine learning technique, constructs several decision trees and aggregates their outputs. By averaging the outcomes of various trees, it lessens overfitting and offers reliable performance on structured datasets. By gradually fixing mistakes caused by earlier models, the sophisticated boosting method XGBoost improves model performance. It is the best-performing model in this project because it is very effective, scalable, and able to handle intricate feature interactions.

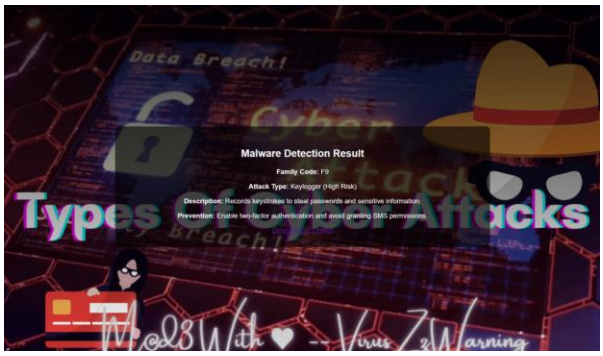
Recurrent neural networks, such as LSTM, are made to identify temporal and sequential patterns in data. While time-series and sequential data analysis is its primary use, This project incorporates it for comparative analysis. These methods work together to provide efficient multi-class malware categorisation. XGBoost outperforms the others in handling feature-based malware detection tasks, as evidenced by its greatest accuracy.

All things considered, the experiment shows how intelligent systems can enhance malware detection and emphasises the significance of machine learning in contemporary cybersecurity. It provides a solid basis for upcoming studies and advancements in threat identification and malware analysis. The suggested solution helps create more dependable and secure digital environments by fusing accuracy, usability, and scalability.

7. RESULTS







8. CONCLUSION

To sum up, this study effectively solves the shortcomings of conventional malware detection methods by presenting a machine learning-based multi-class malware detection system. Using feature-based categorisation techniques, the system can correctly identify many forms of malware, including ransomware, adware, keyloggers, rootkits, and botnets. The research illustrates the usefulness of ensemble and deep learning approaches in cybersecurity applications by building and comparing several machine learning algorithms, such as Random Forest, XGBoost, and LSTM. Because of its capacity to manage intricate feature interactions and minimise overfitting, XGBoost proved to be the most effective algorithm for this task, achieving the best accuracy among these models.

The system becomes practical and easy to use in real time when the trained model is integrated into a Flask-based web application. In addition to receiving predictions and comprehensive information about the discovered virus, including its impact and preventative strategies, users can simply enter feature settings. This raises user awareness of various cyberthreats in addition to improving detection accuracy. The system offers a scalable and effective solution that may be expanded for practical use.

9. FUTURE ENHANCEMENT

To increase its effectiveness and practicality, the suggested multi-class malware detection system can be improved in a number of ways. The incorporation of real-time data analysis utilising system logs and API call sequences for dynamic malware detection is one of the main enhancements. For increased scalability and performance, the system can be implemented in cloud environments and expanded to handle massive datasets. Deep learning models, such as Transformer-based architectures or sophisticated LSTM, can enhance the identification of intricate and dynamic malware patterns. Adding real-time network traffic monitoring can make it easier to spot suspicious activity. Integrating automatic alarm systems to rapidly notify users of possible hazards is another way to improve the system. Putting in place a database to retain user behaviour and

prediction history for further study is another improvement. The user experience can be enhanced by adding interactive dashboards and visual analytics to the user interface. For greater accessibility, mobile applications can be integrated with the system. Adapting to new malware strains can also be aided by ongoing model training with updated datasets. Encryption and safe authentication techniques can be used to further improve security. A hybrid detection strategy may be offered by integration with current antivirus programs. For improved usability, the system may also accommodate multilingual interfaces. All things considered, these improvements will increase the system's resilience, scalability, and suitability for practical cybersecurity applications.

10.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.
[13] M. Alazab, S. Venkataraman, and P. Watters, ''Towards understanding malware behaviour by the extraction of API calls,'' in Proc. 2nd Cybercrime Trustworthy Comput. Workshop, Jul. 2010, pp. 52–59.
[14] D. Sgandurra, L. Muñoz-González, R. Mohsen, and E. C. Lupu, ''Automated dynamic analysis of ransomware: Benefits, limitations and use for detection,'' 2016, arXiv:1609.03020.
[15] T. Munzner, ''Visualization analysis and design-presentation,'' Vis. Anal. Design, vol. 16, pp. 1–3, Aug. 2014.
[16] M. Wagner, A. Rind, N. Thür, and W. Aigner, ''A knowledge-assisted visual malware analysis system: Design, validation, and reflection of KAMAS,'' Comput. Secur., vol. 67, pp. 1–15, Jun. 2017.
[17] B. C. M. Cappers, P. N. Meessen, S. Etalle, and J. J. Van Wijk, ''Eventpad: Rapid malware analysis and reverse engineering using visual analytics,'' in Proc. IEEE Symp. Vis. Cyber Secur. (VizSec), Oct. 2018, pp. 1–8.
[18] M. Wagner, F. Fischer, R. Luh, A. Haberson, A. Rind, D. A. Keim, and W. Aigner, ''A survey of visualization systems for malware analysis,''in Proc. Eurograph. Conf. Vis.-State Art Rep., EuroVis-STAR, Jan. 2015,pp. 105–125.
[19] S. Poudyal, K. P. Subedi, and D. Dasgupta, ''A framework for analyzing ransomware using machine learning,'' in Proc. IEEE Symp. Ser. Comput. Intell. (SSCI), Nov. 2018, pp. 1692–1699.
[20] J. Rafapa and A. Konokix, ''Ransomware detection using aggregated random forest technique with recent variants,'' Authorea, pp. 1–8,Aug. 2024.
[21] S. Razaulla, C. Fachkha, C. Markarian, A. Gawanmeh, W. Mansoor, B. C. M. Fung, and C. Assi, ''The age of ransomware: A survey on the evolution, taxonomy, and research directions,'' IEEE Access, vol. 11, pp. 40698–40723, 2023.
[22] V. Cobilean, H. S. Mavikumbure, B. J. Mcbride, B. Vaagensmith, V. K. Singh, R. Li, C. Rieger, and M. Manic, ''A review of visualization methods for cyber-physical security: Smart grid case study,'' IEEE Access, vol. 11, pp. 59788–59803, 2023.
[23] G. Richer, A. Pister, M. Abdelaal, J.-D. Fekete, M. Sedlmair, and D. Weiskopf, ''Scalability in visualization,'' IEEE Trans. Vis. Comput. Graph., vol. 30, no. 7, pp. 3314–3330, Jul. 2024.
[24] A. Ulmer, M. Angelini, J.-D. Fekete, J. Kohlhammer, and T. May, ''A survey on rogressive visualization,'' IEEE Trans. Vis. Comput. Graph., vol. 30, no. 9, pp. 6447–6467, Sep. 2024.
[25] A. Singh, A. Ikuesan, and H. Venter, ''A context-aware trigger mechanism for ransomware forensics,'' in

Proc. 14th Int. Conf. Cyber Warfare Secur. (ICCWS), 2019, pp. 629–638.
[26] A. Patel and J. Tailor, ‘‘A malicious activity monitoring mechanism to detect and prevent ransomware,’’ Comput. Fraud Secur., vol. 2020, no. 1, pp. 14–19, Jan. 2020.
[27] R. Richardson and M. M. North, ‘‘Ransomware: Evolution, mitigation and prevention,’’ Authorized Administrator Digit. Commons Kennesaw State Univ., vol. 13, no. 1, pp. 10–21, 2017.
[28] K. Hammadeh and M. Kavitha, ‘‘Unraveling ransomware: Detecting threats with advanced machine learning algorithms,’’ Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 9, pp. 484–491, 2023.
[29] H. Sistemas. (2019). Virustotal Public API V2.0. Accessed: Apr. 7, 2024. [Online]. Available: https://www.virustotal.com/en/documentation/publicapi/ [30] IPStack API. Accessed: Jul. 5, 2024. [Online]. Available: https://ipstack. com/documentation
[31] Tshark. Accessed: Apr. 5, 2024. [Online]. Available: https://www. wireshark.org/docs/man-pages/tshark.html
[32] H. Dornhackl, K. Kadletz, R. Luh, and P. Tavolato, ‘‘Malicious behavior patterns,’’ in Proc. IEEE 8th Int. Symp. Service Oriented Syst. Eng., Apr. 2014, pp. 384–389.
[33] A. Singh, R. A. Ikuesan, and H. S. Venter, ‘‘Ransomware detection using process memory,’’ in Proc. Int. Conf. Cyber Warfare Secur., vol. 17, Mar. 2022, pp. 413–422.
[34] T. R. Dendere and A. Singh, ‘‘Ransomware detection using portable executable imports,’’ in Proc. Int. Conf. Cyber Warfare Secur., Mar. 2024, vol. 19, no. 1, pp. 66–74.
[35] J. Ferdous, R. Islam, A. Mahboubi, and M. Z. Islam, ‘‘AI-based ransomware detection: A comprehensive review,’’ IEEE Access, vol. 12, pp. 136666–136695, 2024.
[36] Corvus Forensics. (2019). Virus Share. Accessed: Apr. 4, 2024. [Online]. Available: https://virusshare.com/
[37] N. Naik, P. Jenkins, N. Savage, L. Yang, T. Boongoen, N. Iam-On, K. Naik, and J. Song, ‘‘Embedded YARA rules: Strengthening YARA rules utilizing fuzzy hashing and fuzzy rules for malware analysis,’’ Complex Intell. Syst., vol. 7, no. 2, pp. 687–702, Apr. 2021.
[38] A. Panaras, B. Silverstein, and S. Edwards, ‘‘Automated cooperative clustering for proactive ransomware detection and mitigation using machine learning,’’ TechRxiv, pp. 1–9, Sep. 2024.
[39] S. Zhang, T. Du, P. Shi, X. Su, and Y. Han, ‘‘Early detection and defense countermeasure inference of ransomware based on API sequence,’’ Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 10, pp. 632–641, 2023.
[40] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, ‘‘Improved training of Wasserstein GANs,’’ in Proc. Adv. Neural Inf. Process. Syst., vol. 30, Dec. 2017, pp. 5769–5779.
[41] Python. Accessed: Apr. 4, 2024. [Online]. Available: https://docs.python. org/