International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 9, Special Issue ICCIS-2025, pp 48-54, May-2025
https://doi.org/10.47001/IRJIET/2025.ICCIS-202507
Second International Conference on Computing and Intelligent Systems (ICCIS-2025)

Securing Smart Contracts in Fog Computing: Machine Learning-Based Attack Detection for Registration and Resource Access Granting

[1]A.Gowtham, [2]K.Nanda Kishore Reddy, [3]M.Somu Sekhar Naik

[1]Assistant Professor, Department of Computer Science and Engineering and Cyber Security (UG), Madanapalle Institute of Technology & Science (Autonomous), Madanapalle, India
[2,3]UG Scholar, Department of Computer Science and Engineering and Cyber Security (UG), Madanapalle Institute of Technology & Science (Autonomous), Madanapalle, India
E-mails: [1]gowthama@mits.ac.in, [2]nkishore9850@gmail.com, [3]somusekharnaik316@gmail.com

***Abstract* - Attack resistance in smart contracts has become one of the paramount concerns in this fog computing evolving landscape. Based on this problem, this study utilizes machine learning for the detection of attacks from the analysis of Ethereum transaction data and smart contract interactions. There are different methods for feature extraction: Term Frequency-Inverse Document Frequency (TF-IDF), Bag of Words (BoW), and N-gram methods for converting raw data into readable formats by machine. Several machine learning classifiers, including XGBoost, Random Forest, Light Gradient Boosting, and Extra Trees, are applied to detect the attacks. These models should offer high detection accuracy with low computational complexity, enabling scalability for real-time fog computing applications. Experimental results show that Extra Trees, with N-gram features, outperformed other models by achieving an accuracy of 83%. The proposed system is promising toward the efficient detection and classification of attacks in a dynamic fog environment.**

Keywords: Fog Computing, Smart Contracts, Machine Learning, Attack Detection, Ethereum, Feature Extraction, Computational Efficiency.

I. INTRODUCTION

In the recent past, fog computing has been an emerging promising paradigm for the extension of cloud computing capabilities toward the edge of networks. The enabling decentralized processing of data close to the devices of ends decreases latency and bandwidth usage, enhancing overall system efficiency. With an increase in IoT devices and smart contracts, however, there has been an intense need for the secure and reliable systems of fog computing. Security within the realm [1] of fog computing in smart contracts is another challenging aspect. Smart contracts in a blockchain-based system are the self-executing agreement coded into such systems. Decentralized applications rely heavily upon such contracts as they enable trustless and automated transactions. The higher the number of users consuming blockchain-based applications, the more susceptible the application would become to various types of attacks, which include reentrancy, overflow, underflow, and front-running attacks.

As the integrity of the system largely relies on smart contracts, there is a crucial need for robust and scalable detection mechanisms for the protection [2] of systems. Traditional security measures often fail to handle the dynamic and decentralized nature of the fog computing environment. Because of the resource constraints and the geographical distribution of the devices within fog networks, the security protocols that are effective in centralized cloud systems do not work in these environments. Manual detection of malicious activities in smart contracts can be time-consuming and error-prone, especially when the volume of transactions is large. Thus, there is a need for automated systems that can efficiently detect malicious behavior in smart contracts without introducing significant overhead.

Machine learning is promising for this problem, as it can analyze vast amounts of data, identify patterns, and make predictions based on historical behavior. By using the machine learning algorithm to identify anomalies in smart contract execution, which may [3] imply attacks. Such systems learn from data and evolve over time, especially in dynamic environments such as fog computing. Sources of data for training and testing machine learning models include various ones, and blockchain transaction data is one of the most reliable. It is, in fact one of the most widely used blockchain platforms and it has a massive amount of transaction data, which can be mined to detect an attack. Tools like Etherscan.io, which are publicly available, provide information on account histories of transactions, contract code,

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 9, Special Issue ICCIS-2025, pp 48-54, May-2025
https://doi.org/10.47001/IRJIET/2025.ICCIS-202507
Second International Conference on Computing and Intelligent Systems (ICCIS-2025)

and execution details that can be used for extracting meaningful features related to machine learning models.

Feature extraction-This is a preprocessing step that transforms raw data into a format that can now be fed into the algorithm of machine learning [4] algorithms. Techniques such as Term Frequency-Inverse Document Frequency (TF-IDF), Bag of Words (BoW), and N-gram are popular methods for representing textual data and contract interactions in structured format. These methods may capture hidden relations between different pieces of smart contracts and transactions, which can machine learning models identify as patterns showing malicious activity. After preprocessing of data, machine learning classifiers are ready to classify attacks. Different algorithms have been tested for the purpose, including XGB, RF, Light Gradient Boosting, and Extra Trees, in detecting malicious activities on smart contracts. Each of these classifiers has its strengths and weaknesses in terms of accuracy [5], computational complexity, and scalability. For example, XGB and RF are known for their high accuracy in classification tasks, while Extra Trees often performs well with large, high-dimensional datasets. However, high accuracy may come at the cost of increased computational complexity, which can be a concern in fog computing environments where resources are limited. Therefore, the balance between detection accuracy and computational efficiency is necessary to ensure that the system can scale in real-time scenarios.

The main objective of this research is to develop a machine learning-based attack detection system that can effectively identify and classify malicious activities targeting smart contracts in fog computing environments. The proposed system aims [6] to leverage various feature extraction techniques and machine learning models to achieve high detection accuracy while minimizing computational overhead. By addressing both the security and scalability concerns, the system aims to provide a practical solution that can be deployed in real-time environments, ensuring the safety and reliability of smart contract execution. The findings from this research could significantly enhance the security posture of fog computing systems, enabling them to safely host decentralized applications and services.

This work is organized as Section II presenting a review of the literature survey. Section III describes the methodology, highlighting its key features and functionality. Section IV discusses the results, analysing the system's effectiveness. Lastly, Section V concludes with the main findings and explores future implications.

II. LITERATURE SURVEY

The most recent advances in attack detection for smart contracts have been in using static analysis methods, which examine the code of smart contracts without executing them. These methods can identify vulnerabilities such as reentrancy and overflow errors. However, they usually fail to detect runtime issues or more complex attacks that emerge from dynamic interactions between contracts. In a bid to supplement static analysis with dynamic detection, machine learning has been proposed in identifying malicious behavior through transaction data analysis. Approaches in such a nature have the ability to detect patterns and anomalies indicative of attack strategies not captured by more traditional methods. Machine learning-based detection systems of smart contracts most often rely on data from platforms like Ethereum's blockchain for model training.

These systems apply various feature extraction techniques to represent raw blockchain data in a [7] machine-readable format. This transforms transaction data and contract interactions into structured inputs that machine learning algorithms can use to identify abnormal patterns or malicious activities within the smart contract environment. The methods can be adapted to specific attack types, thus enhancing detection accuracy over time. Recent research has focused on comparing the performance of various machine learning algorithms in detecting smart contract vulnerabilities. Algorithms have been evaluated in terms of their ability to detect known attacks in smart contracts. These models often show high accuracy in attack classification tasks [8], with some outperforming traditional methods. But a trade-off in terms of accuracy of detection versus computational efficiency will always remain as a challenge, especially in fog computing, as low latency with minimal overhead are the essential criteria for real-time operation. Other than supervised learning, there has also been interest in unsupervised techniques in detecting attacks in smart contracts.

Unsupervised models like clustering and anomaly detection algorithms are of value when labelled data is not available or in short supply. These work based on an analysis of the normal behavior of smart contracts, looking for anomalies that could potentially represent malicious activity. Unsupervised [9] learning might identify novel attacks, which had not been previously known, although its precision will often not match that of the supervised models. The hybrid models combining both approaches do hold some promise for improvement in detection. The challenge with the application of machine learning for the detection of attacks in smart contracts is that the dataset is highly imbalanced. There are thousands of normal transactions for every single attack transaction.

The problem of such imbalance is biased models, which do not work well for detecting rare scenarios of attacks. Techniques like oversampling [10], undersampling, and cost-sensitive learning have been proposed for overcoming this issue. Such approaches seek to ensure that the model gives due attention to rare cases of attacks without sacrificing general performance. Researchers in this field focus on improving the robustness of machine learning models in imbalanced settings. Another significant concern with applying machine learning models in real-time smart contract attack detection is their computational complexity.

In fog computing environments, where processing resources are limited and distributed across various edge devices, it is essential to optimize the models to run efficiently without overloading the system. Feature selection and dimensionality reduction [11] techniques, such as Principal Component Analysis (PCA) and Recursive Feature Elimination (RFE), have been investigated to improve the performance of models by reducing the number of features without sacrificing accuracy. These approaches try to optimize the process of detection to make it suitable for deployment on resource-constrained devices. Over the past couple of years, ensemble learning approaches for smart contract attack detection have started to emerge.

Methods like Bagging, Boosting, and Stacking are based on the idea that by combining predictions from multiple models, the final overall detection capability will be higher. Ensemble-based methods are thus highly beneficial to increase accuracy as [12] well as robustness by utilizing the strengths of various algorithms put together. However, increased complexity of ensemble models raises issues of computational efficiency, and thus their application in fog computing environments is an area of ongoing research. Deep learning techniques have also been explored in recent research to ensure smart contract security. It applies [13] neural networks, especially recurrent neural networks (RNNs) and long short-term memory (LSTM) networks, for the analysis of transaction sequences in order to identify complex attack patterns. These models are able to learn temporal dependencies within the data, making them well-suited for identifying sophisticated, multi-step attacks that may unfold over time. The methods based on deep learning may also have high accuracy for detection purposes, but usually require a vast amount of labeled data [14] and intense computational resources; thus, being impractical for certain environments. Due to decentralized finance (DeFi) and DApps-based smart contract-based attacks, this work focuses more on the crucial aspect of designing a robust system for detection in this field.

The research focuses on finding specific attack vectors that target the financial components of smart contracts, like price manipulation, flash loans, and oracle vulnerabilities. Such attacks exploit weaknesses [15] in the interaction between the contract and external data sources, thereby causing significant financial effects. It demands the analysis of both the contract code and external factors such as transaction patterns and market behavior for detection. Evaluation metrics are important for the understanding of how effective the machine learning models are in smart contract attack detection. They give an insight into how good the model is at detecting true positives without generating false positives and false negatives. Cross-validation techniques [16] are used to assess the generalizability of the model and prevent overfitting. Researchers have demonstrated that while accuracy is a good metric, precision and recall are sometimes more informative, especially when the attack scenarios are rare or imbalanced. This multi-metric evaluation of models provides a better view of the performance of models. Another challenge in applying [17] to smart contract attack detection is the changing nature of attack strategies.

The detection system has to be capable of identifying attack types that it has not previously seen. Therefore, researchers have been studying online learning and adaptive models that could update their parameters based on new data, which would enable them to adapt to emerging threats. This ensures the effectiveness of the detection system because attack [18] techniques will evolve. In order to maintain high detection accuracy over time, it is possible to use real-time feedback and continuously retrain models on fresh data. The integration of machine learning-based attack detection systems into fog computing environments has been explored as a potential solution for real-time threat monitoring.

Fog computing offers distributed and decentralized computing resources, which are well-suited for deploying lightweight detection systems that can process data closer to the source. This reduces the response time and has minimal impacts from attacks, though edge devices provide limited computational resources [19] and necessitate low-latency detection. Other efforts include optimizing and developing strategies for the deployment of efficient models. Many other research works focus on the hybrid attack detection systems, where more than one algorithm is combined, along with a variety of techniques for feature extraction. In order to take the best from different models, researchers try to combine them in a way that their strengths are enhanced [20] to increase detection accuracy and robustness. For example, integrating traditional models like Random Forest with advanced techniques like deep learning will create a system that can handle simple as well as complex attack scenarios. These hybrid systems are promising solutions to the diverse and evolving threats that target smart contracts in fog computing environments.

III. METHODOLOGY

With the wide spread adoption of smart contracts within the decentralized system such as that found in Ethereum, there comes with it various challenges, specifically with regard to security in a fog computing environment. Fog computing extends cloud computing capability down to the edge of the network; its distributed nature coupled with being resource-constrained leaves it prone to attacks especially when targeting smart contracts. As smart contracts execute automatically, they are vulnerable to different types of attacks, including reentrancy, overflow, and front-running, which may result in huge financial and data breaches. Therefore, it is crucial to develop efficient and scalable attack detection systems. Machine learning has emerged as a promising solution for identifying patterns of malicious behavior in smart contracts. This research aims to explore the possibility of machine learning algorithms in detecting attacks on smart contracts in fog computing, while maintaining high detection accuracy and low computational overhead for real-time applications.

A. Data Collection

The first step in this methodology is data collection from Ethereum's official blockchain explorer, etherscan.io. This platform offers access to all comprehensive details regarding Ethereum transactions, smart contract interactions, and other related metadata. Data was selected to reflect a wide coverage of Ethereum transactions, both regular and attack related. Transactions relating to smart contracts are targeted in this research, as they stand to be affected the most in malicious activities. The raw data includes relevant features such as amounts of transactions, gas usage, sender and receiver addresses, addresses of contracts and functions called in the contract. The data is collected over a specified time frame to ensure diverse transaction patterns, both normal and attack-related. This dataset is vital for training machine learning models and detecting possible threats to smart contracts in fog computing environments.

B. Feature Extraction

Once the data is collected, it undergoes preprocessing and feature extraction to transform it into a form suitable for machine learning. Multiple techniques are applied in feature extraction: Term Frequency-Inverse Document Frequency, Bag of Words, and N-gram. TF-IDF is applied for weighing terms or tokens based on their importance within the interactions with the contract that reflects significance for identifying attack-related patterns. BoW reduces the data to a word level, thus treating the data into a bag or collection of tokens. From there, these bags of tokens determine the reappearing patterns that exist within that data. Through the N-gram technique, sequences of neighboring words are used to obtain insight into how certain terms within contract data interplay with one another. These methods of feature extraction enable the data to be represented numerically, thus making it ready for inputting into machine learning models. The objective is the extraction of meaningful features for enhancing attack detection performance.

C. Training of Models

After feature extraction, the next step involved in this process is the training of the machine learning models. Four of the most prominent classifiers have been chosen for this study: Extreme Gradient Boosting (XGB), Random Forest (RF), Light Gradient Boosting, and Extra Trees. These models have been picked because of their proven capability to handle large datasets and robust performance in classification tasks. The output obtained by extracting the features from Ethereum smart contract interactions forms the input for the training models. In the course of the training, each classifier gets trained to identify patterns of malicious activity pertaining to transactions. The models are tuned to hyperparameter optimization through the use of cross-validation and other methods, and in this way, predictive accuracy is maximized. Cross-validation techniques are used to avoid overfitting and ensure generalization of the models. This training phase allows the models to adapt to various attack scenarios, equipping them with the ability to distinguish between normal and malicious transactions effectively.

D. Model Evaluation

After training the models, the next step is to evaluate their performance using a variety of metrics. These include accuracy, precision, recall, F1 score, and computational time, all of which are crucial for the evaluation of the models' ability to detect attacks on smart contracts. Accuracy is the overall correctness of the model. Precision and recall give insight into the model's ability to correctly identify attacks as well as not attacks. The F1 score balances precision and recall to provide a holistic measure of model performance. Cross-validation is applied during the testing phase to ensure that the models will have a consistent behavior between data splits. The time it takes to train the models is used to check if they can effectively operate in real-time computing fog environments for timely decisions. The ultimate aim of this testing phase is identifying the best performing model both in terms of accuracy and efficiency in order to detect attacks on smart contracts.

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 9, Special Issue ICCIS-2025, pp 48-54, May-2025
https://doi.org/10.47001/IRJIET/2025.ICCIS-202507
Second International Conference on Computing and Intelligent Systems (ICCIS-2025)

E. System Implementation

The identified model is then incorporated into a real-time attack detection system for a fog computing environment. The trained machine learning model is used by the system in real-time to analyze incoming Ethereum transactions and interactions with smart contracts. The scalability of the system is implemented because it will operate in dynamic fog computing environments with limited computational resources. The chosen model, Extra Trees with N-gram features, is best suited for this application because of its high accuracy and low computational complexity. The system can identify reentrancy, overflow, and front-running attacks while minimizing the resources required to operate. It continuously monitors transactions in smart contracts, alerting stakeholders to any suspicious activity and enabling timely interventions. This real-time implementation ensures that the attack detection system is both practical and efficient for use in fog computing environments.

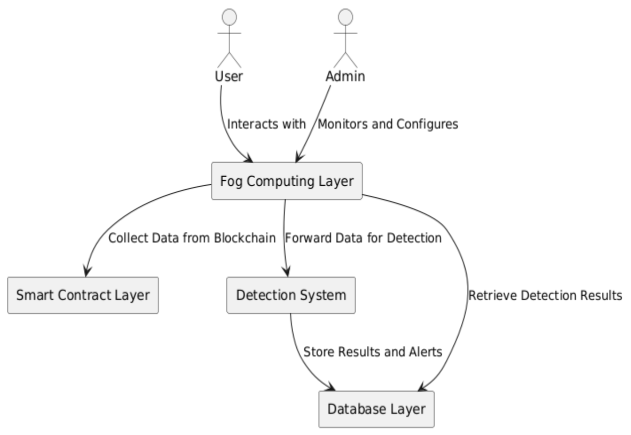

Figure 1: Architecture Diagram

IV. RESULT AND DISCUSSION

Thus, the machine learning models showed an evaluation that differentiated their capabilities against attacks in a fog computing scenario on smart contracts. In one of the implemented models, among the three under consideration, and using N-gram for extracting features, this model's efficiency was observed in achieving accuracy 83%. This performance was much more impressive than the other classifiers, such as Extreme Gradient Boosting (XGB), Random Forest (RF), and Light Gradient Boosting, which achieved an accuracy of about 80%, 80%, and 81%, respectively, with Term Frequency-Inverse Document Frequency (TF-IDF) and Bag of Words (BoW). It's obvious that Extra Trees models work much better with the model in terms of the capability of working well with high-dimensional data, which is one of the crucial needs of the smart contract analysis environment because all features are required to be processed simultaneously. N-gram also helps capture the sequential pattern inside the data; thus, more context helps to better distinguish normal and malicious activity.

Although the TF-IDF approach gave slightly lower accuracy than the N-gram, it was more computationally efficient and, therefore, much more attractive for real-time computing in the fog computing environment. The lower computationally overhead that can be given by the TF-IDF allows faster processing and less consumption of resources, which in real-time scenarios is necessary because of the limitations in the computational capability of the nodes in the fog architecture. Such a trade-off between accuracy and computational time is an important consideration for real-time attack detection systems in real-time deployments.

Precision, recall, and F1 scores are further measures to show how effective Extra Trees are in the attack detection. A high precision rate was demonstrated, meaning most threats identified were actually threats, hence reducing false positives and making the system more reliable. Recall was also high but, at some points, not as good as XGB and RF. XGB and RF were also shown to be more sensitive to attacks in a way that varies based on patterns, and, interestingly, these models outperform the other in terms of their F1 scores, showing balance between precision and recall.

The computational time, when compared, was found to be much less complex in the TF-IDF approach. In contrast, other models that were based on BoW and N-gram methods took much more time for feature extraction and training of the model, which would be a major limitation in places where response time is of importance. In this regard, however, the N-gram and BoW models were more inclined to understand the contextual patterns. This is because contextual patterns can be used for detecting more subtle attacks that do not easily raise suspicion.

From the results obtained in this work, it appears that the Extra Trees model, with N-gram features, is the best model for detecting attacks while remaining computationally feasible. However, in real-time implementation within resource-constrained fog environments, the TF-IDF approach strikes a nice balance between detection performance and computational cost. Which of these approaches would be adopted depends on the application requirements of fog computing; that is, it depends on the need for quick detection or on the requirement to detect more attacks.

The findings from this research can be applied to improve security in smart contract systems, especially in the fog computing environment, where the balance between high accuracy and low computational overhead is essential. Future work could involve fine-tuning these models to get even better performance or investigating further feature extraction techniques to further improve detection capabilities. In addition, incorporating these models into a real-time monitoring system will ensure the security of smart contracts in fog computing against the ever-evolving threats.

V. CONCLUSION

In conclusion, this study demonstrates the potential of machine learning algorithms for detecting attacks on smart contracts within fog computing environments. The results indicate that Extra Trees, using the N-gram feature extraction technique, outperformed other classifiers in terms of accuracy. This suggests that Extra Trees is particularly well-suited for identifying patterns indicative of malicious activity in smart contracts. It is apparent that the contextual relationships in the data captured by this model through the N-gram approach highly contribute to its high detection performance. However, the trade-off between accuracy and computational complexity was also evident in the study. The TF-IDF method, although achieving slightly lower accuracy, was computationally more efficient, which is a good candidate for real-time applications in resource-constrained environments like fog computing. This points out the importance of balancing detection capabilities with resource efficiency because decision-making needs to be fast, but with minimal computational resources, in fog computing environments.

The outcomes confirm that applying machine learning in the proper context with suitable feature extraction techniques is beneficial in securing decentralized systems and particularly smart contracts from attacks in a dynamic, distributed setting like fog computing. This work is valuable for adding insights to using machine learning in detecting attacks in a fog computing environment. It emphasizes the need to choose the right balance between model complexity and computational cost, so that detection systems can be run in real-time. Future research may explore ways to further optimize these models or incorporate additional features to improve detection accuracy while maintaining scalability and efficiency in real-world applications. The findings are hence a strong stepping stone toward development of strong and real-time attacking detection systems aimed at securing a smart contract.

REFERENCES

[1] F. Akbarian, W. Tärneberg, E. Fitzgerald and M. Kihl, "Attack Resilient Cloud-Based Control Systems for Industry 4.0," in IEEE Access, vol. 11, pp. 27865-27882, 2023, doi: 10.1109/ACCESS.2023.3259063.

[2] A.V. Nagarjun and S. Rajkumar, "Design of an Anomaly Detection Framework for Delay and Privacy-Aware Blockchain-Based Cloud Deployments," in IEEE Access, vol. 12, pp. 84843-84862, 2024, doi: 10.1109/ACCESS.2024.3414998.

[3] A.M. Abdallah, A. Saif Rashed Obaid Alkaabi, G. Bark Nasser Douman Alameri, S. H. Rafique, N. S. Musa and T. Murugan, "Cloud Network Anomaly Detection Using Machine and Deep Learning Techniques—Recent Research Advancements," in IEEE Access, vol. 12, pp. 56749-56773, 2024, doi: 10.1109/ACCESS.2024.3390844.

[4] U. Islam, A. Al-Atawi, H. S. Alwageed, M. Ahsan, F. A. Awwad and M. R. Abonazel, "Real-Time Detection Schemes for Memory DoS (M-DoS) Attacks on Cloud Computing Applications," in IEEE Access, vol. 11, pp. 74641-74656, 2023, doi: 10.1109/ACCESS.2023.3290910.

[5] W. Wang, X. Du, D. Shan, R. Qin and N. Wang, "Cloud Intrusion Detection Method Based on Stacked Contractive Auto-Encoder and Support Vector Machine," in IEEE Transactions on Cloud Computing, vol. 10, no. 3, pp. 1634-1646, 1 July-Sept. 2022, doi: 10.1109/TCC.2020.3001017.

[6] V. D. M. Rios, P. R. M. Inácio, D. Magoni and M. M. Freire, "Detection and Mitigation of Low-Rate Denial-of-Service Attacks: A Survey," in IEEE Access, vol. 10, pp. 76648-76668, 2022, doi: 10.1109/ACCESS.2022.3191430.

[7] J. Deng et al., "A Survey on Vehicular Cloud Network Security," in IEEE Access, vol. 11, pp. 136741-136757, 2023, doi: 10.1109/ACCESS.2023.3339192.

[8] J. M. Singh and R. Ramachandra, "3-D Face Morphing Attacks: Generation, Vulnerability and Detection," in IEEE Transactions on Biometrics, Behavior, and Identity Science, vol. 6, no. 1, pp. 103-117, Jan. 2024, doi: 10.1109/TBIOM.2023.3324684.

[9] A.B. Bhutto, X. S. Vu, E. Elmroth, W. P. Tay and M. Bhuyan, "Reinforced Transformer Learning for VSI-DDoS Detection in Edge Clouds," in IEEE Access, vol. 10, pp. 94677-94690, 2022, doi: 10.1109/ACCESS.2022.3204812.

[10] M. Mehmood, R. Amin, M. M. A. Muslam, J. Xie and H. Aldabbas, "Privilege Escalation Attack Detection and Mitigation in Cloud Using Machine Learning," in IEEE Access, vol. 11, pp. 46561-46576, 2023, doi: 10.1109/ACCESS.2023.3273895.

[11] H. Attou, A. Guezzaz, S. Benkirane, M. Azrour and Y. Farhaoui, "Cloud-Based Intrusion Detection Approach Using Machine Learning Techniques," in Big Data Mining and Analytics, vol. 6, no. 3, pp. 311-320, September 2023, doi: 10.26599/BDMA.2022.9020038.

[12] A.Ahmim, F. Maazouzi, M. Ahmim, S. Namane and I. B. Dhaou, "Distributed Denial of Service Attack

Detection for the Internet of Things Using Hybrid Deep Learning Model," in IEEE Access, vol. 11, pp. 119862-119875, 2023, doi: 10.1109/ACCESS.2023.3327620.

[13] E. Taherian-Fard, T. Niknam, R. Sahebi, M. Javidsharifi, A. Kavousi-Fard and J. Aghaei, "A Software Defined Networking Architecture for DDoS-Attack in the Storage of Multimicrogrids," in IEEE Access, vol. 10, pp. 83802-83812, 2022, doi: 10.1109/ACCESS.2022.3197283.

[14] J. Jasmine Shirley and M. Priya, "Hybrid MRMR-PCA BagDT—An Effective Feature Selection Based Ensemble Model for Real-Time Intrusion Detection in IoT Environment," in IEEE Access, vol. 12, pp. 144230-144248, 2024, doi: 10.1109/ACCESS.2024.3468897.

[15] M. F. Saiyedand and I. Al-Anbagi, "Deep Ensemble Learning With Pruning for DDoS Attack Detection in IoT Networks," in IEEE Transactions on Machine Learning in Communications and Networking, vol. 2, pp. 596-616, 2024, doi: 10.1109/TMLCN.2024.3395419.

[16] L. Almuqren, H. Alqahtani, S. S. Aljameel, A. S. Salama, I. Yaseen and A. A. Alneil, "Hybrid Metaheuristics With Machine Learning Based Botnet Detection in Cloud Assisted Internet of Things Environment," in IEEE Access, vol. 11, pp. 115668-115676, 2023, doi: 10.1109/ACCESS.2023.3322369.

[17] S. An, A. Leung, J. B. Hong, T. Eom and J. S. Park, "Toward Automated Security Analysis and Enforcement for Cloud Computing Using Graphical Models for Security," in IEEE Access, vol. 10, pp. 75117-75134, 2022, doi: 10.1109/ACCESS.2022.3190545.

[18] Y. Zhang et al., "Pattern Corruption-Assisted Physical Attacks Against Object Detection in UAV Remote Sensing," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 17, pp. 12931-12944, 2024, doi: 10.1109/JSTARS.2024.3422377.

[19] R. Flowers, "A Zero-Day Cloud Timing Channel Attack," in IEEE Access, vol. 10, pp. 128177-128186, 2022, doi: 10.1109/ACCESS.2022.3227420.

[20] P. Rana, I. Batra, A. Malik, I. -H. Ra, O. -S. Lee and A. S. M. Sanwar Hosen, "Efficacious Novel Intrusion Detection System for Cloud Computing Environment," in IEEE Access, vol. 12, pp. 99223-99239, 2024, doi: 10.1109/ACCESS.2024.3424528.

Citation of this Article:

A.Gowtham, K.Nanda Kishore Reddy, & M.Somu Sekhar Naik. (2025). Securing Smart Contracts in Fog Computing: Machine Learning-Based Attack Detection for Registration and Resource Access Granting. In proceeding of Second International Conference on Computing and Intelligent Systems (ICCIS-2025), published in *IRJIET*, Volume 9, Special Issue ICCIS-2025, pp 48-54. Article DOI https://doi.org/10.47001/IRJIET/2025.ICCIS-202507

****