ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 9, September 2024

Optimizing Smart Contract Security with Machine Learning: Tools and Techniques

[1] Jasvant Mandloi1*, [2] Pratosh Bansal

[1] [2] Department of Information Technology, Institute of Engineering and Technology, Devi Ahilya Vishwavidyalaya, Indore, India

Corresponding Author Email: [1] jasvant28284@gmail.com

Abstract*— As the adoption of Blockchain technology continues to grow, ensuring the security of smart contracts has become a critical concern. Traditional methods of securing smart contracts, such as static analysis and formal verification, have shown limitations in terms of scalability, accuracy, and the ability to adapt to evolving threat landscapes. This paper explores the innovative application of machine learning (ML) techniques to enhance the security of smart contracts. We investigate several tools and techniques that apply machine learning to smart contract vulnerability identification anomalous behavior detection and threat prediction. We investigate a range of ML tools and methodologies, including anomaly detection, classification algorithms, and reinforcement learning, to identify and mitigate vulnerabilities more effectively. Additionally, we discuss the integration of ML-based tools with existing security frameworks to create a comprehensive, adaptive security system for smart contracts. At last is suggested that now focus should be more on developing an ML model that integrates with the present tools to give better results.

***Index Terms*— Machine Learning, Security, Front-Running, Bad Randomness, Vulnerability Detection and Smart Contracts.**

I. INTRODUCTION

Smart Contracts have gained significant attention and attraction due to their immense potential within Blockchain technology. They are self-executed tiny computer programs that reside on the Blockchain. This feature ensures the immutability of the data as long as the integrity of the Blockchain remains intact. But before these hopes may come true, there are still a lot of obstacles to overcome. First, in a distributed setting where certainty is not guaranteed, it is challenging for a single consensus mechanism to ensure strong security and dependability in a decentralized transaction verification network. Second, the difficulty of Blockchain technology to scale has seriously impeded its adoption and spread into new industries and businesses. The immense potential of smart contracts within Blockchain technology has garnered significant attention and enthusiasm. They are tiny computer programs that are located on the Blockchain. This feature guarantees the data's immutability as long as the blockchain's integrity is preserved. Therefore, users can be assured that the program remains unaltered and untampered with. This eliminates the necessity for individuals to depend on other nodes or a central third party, thereby instilling confidence in a decentralized environment. A recent report estimates that the smart contract industry will be valued at USD 144. 95 million in 2020 and will grow at a compound annual growth rate (CAGR) of 24. 55 percent between 2021 and 2028 to reach USD 770. 52 million [1][2].

Utilizing machine learning techniques is a highly effective method for identifying smart contract security flaws. In recent years, there has been remarkable advancement in the field of machine learning, establishing it as a prominent discipline in contemporary computer science. Developing algorithms that can learn from data on their own and use that knowledge to make decisions predictions or classifications when faced with new data is the primary goal of machine learning [3]. Machine learning algorithms have been significantly used in advanced smart-contract security detection. Unfortunately, there is a lack of comprehensive research on applying machine learning to find security threats in smart contracts in the present scenario. To bridge this knowledge gap, we have decided to compose an extensive review paper on it. We hope to offer a thorough overview of the obstacles faced today by smart contracts as well as future directions where machine-learning approaches are used for detection and development. This work will resource for academics, the research community, and professionals in business. In section 2 we have done a related work review study to get an idea about present work in the field. In section 3 evaluation and adoption of smart contracts have been covered. After that in section 4 security landscape of the smart contract is discussed and in section 5 some of the recognized attacks on the smart contract with security measures are covered and details about the machine learning tools and techniques used are also placed.

II. RELATED WORK

We have thoroughly examined the latest research articles in the area of identifying vulnerabilities in smart contracts, as well as studies focusing on using machine learning for vulnerability detection. Though machine learning has made significant strides in identifying vulnerabilities in smart contracts we noticed an interesting phenomenon during this process. Notwithstanding the notable progress machine learning has made in identifying vulnerabilities in smart

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 9, September 2024

contracts we noticed an interesting phenomenon during this process. There is a noticeable absence of literature that specifically focuses on the use of machine learning for this purpose. The purpose of this paper is to bridge this gap by offering a more thorough and well-structured theoretical framework as well as a helpful practical reference for researchers and developers involved in this field. This work presents a research investigation on attacks targeting Ethereum smart contracts. Their research delved into different types of attacks and real-life examples, along with methods to enhance the safety of smart contracts. Their research centered on analyzing different attack tactics and provided developers with useful advice and tools to protect against these kinds of attacks [4]. The author examined and summarized the Blockchain smart contract security verification procedure. Exploring different approaches and strategies for guaranteeing the security of smart contracts was the main focus of their research [5]. In this study, a survey on the use of deep learning techniques for software vulnerability analysis was examined. This paper primarily examines the suitability and efficacy of deep learning techniques for the identification and analysis of software vulnerabilities. This study did not focus on any one particular technology. Rather it examined the target technology as part of the classification scheme [6,7].

III. EVOLUTION AND ADOPTION OF SMART CONTRACT TECHNOLOGY

Since Nick Szabo first introduced the concept of smart contract technology in 1994 the technology has advanced remarkably. Smart contracts which were first developed as theoretical ideas have experienced a remarkable transformation and are now providing a digital age revolution for conventional contract procedures. We examine the evolution of smart contracts in the following table highlighting any potential drawbacks and analyzing their salient characteristics as well as their application in different industries.

Table I: Evolution of Smart Contract & Its Application [8]

Year	Version	Application	Key Developments	Shortcomings
1994	Conceptual	Theoretical Foundations	Introduction of the concept of self-executing contracts by Nick Szabo.	Limited technological infrastructure for practical implementation.
2015	Ethereum 1.0	Decentralized Platforms	Launch of Ethereum, enabling the execution of complex smart contracts.	The DAO incident highlights vulnerabilities and security concerns.
2020	Ethereum 2.0	Scalability Improvements	Transition to Ethereum 2.0 to enhance scalability through Proof-of-Stake.	Ongoing transition may lead to compatibility issues with existing contracts.
2021	Layer 2 Solutions	Improved Throughput	Introduction of Layer 2 scaling solutions to address scalability challenges.	Limited adoption due to the complexity of implementing Layer 2 solutions.
2022	Polkadot, Cosmos	Interoperability	Emergence of platforms emphasizing interoperability between blockchains.	Potential security risks in cross-chain transactions.
2023	Defi Platforms	Financial Services	Rise of DeFi platforms offering automated lending, borrowing, and trading.	Vulnerabilities in smart contracts give rise to financial losses and exploits.

IV. SMART CONTRACT SECURITY LANDSCAPE

With a focus on Blockchain networks such as Ethereum research has devoted a great deal of attention to smart contract security. There is a great deal of concern about this area which has led to a lot of research. Smart contracts are those that run on a predefined set of rules. It is crucial to guarantee the confidentiality and security of these contracts to stop any weaknesses or possible exploits. The countermeasures available to address these concerns are covered in this section.

A. Recognized Security and Privacy Attacks in Smart Contracts

Table II: Details some of the Popular threats and attacks on Smart contracts [9,10]

Vulnerability/ Attack	Real-World Incident	Cause	Resultant	Potential Remedies

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 9, September 2024

Reentrancy Attack	DAO (2016)	inappropriate access control	substantial loss of Ethereum	Use the checks-effects-interactions pattern and put access control into practice. For ether transfers utilize the withdraw pattern.
Underflow/Overflow of Uninitialized Integer	King of the Ether Throne (2016)	Arithmetic operations that are badly designed.	Token loss or contract control loss	To stop integer overflow or underflow use SafeMath or other comparable libraries.
Unsettled Variables	Various	Improper initialization of variables	Variable contract behavior	Before use make sure all variables are initialized.
Access-Control	Batch Overflow (2018)	Incorrect settings for access control	Modifying the terms of the contract without consent.	Put strong access control measures in place.
Front-Running Attack	Bancor (2018)	Tricking the ordering of a transaction.	financial loss as a result of front-running attacks.	To reduce front-running apply techniques such as commit-reveal.
Dependence of Timestamps	Many such Incidents	Using block timestamps only	Sensitivity to manipulation by miners	less reliance on timestamps in blocks.
Vulnerabilities related to Gas Limit	Parity Multisig Wallet (2017)	Incorrect calculation of gas.	Transaction execution is not possible.	Take gas refunds into consideration and estimate gas limits carefully.
Uncertain External Information.	DeFi Abuse (Multiple)	Untrustworthy external data sources or oracles	Inconsistent data impacts contract behavior.	Employ several oracles and put procedures in place for data verification.
Absence of Error Management	Many Events	Neglecting to deal with errors or exceptions.	Contract vulnerabilities ignored	Put strong error-handling and fail-safe measures in place.

B. Existing Security Measures in Smart Contracts

Ethereum smart contracts have the potential to revolutionize various industries by streamlining transactions automating procedures and enabling businesses. However, their inherent vulnerability to cyberattacks necessitates a nuanced understanding and implementation of robust security measures to protect user funds, maintain trust in the ecosystem, and unlock the full potential of this technology. The consequences restrictions and advancements of Ethereum smart contract security mechanisms are explored in this paper.

Threat landscape by quantifying risks

Ethereum's decentralized structure provides it with resistance to censorship, yet it also makes it vulnerable to malicious individuals. There has been a concerning increase in reentrancy attacks, which involve exploiting transaction flow to steal funds. According to DeFi Pulse, these mishaps cost at least $1. 46 billion between 2020 and 2023. $120 million was spent on the most recent Badger DAO incident. Other risks to be aware of are denial-of-service attacks front-running and integer overflows and underflows that could prevent users from accessing the system. The DAO Hack (2016) demonstrates how reentrancy can cause a $60 million loss as a result of lax security.

Solid foundations are the first defense

Well-written code is essential to creating safe smart contracts. Access control error handling and input validation

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 9, September 2024

can significantly reduce the risk of malicious attacks. Errors and duplication are decreased by using well-known libraries like OpenZeppelin Contracts which provide pre-audited modules. Over 10000 vulnerabilities in contracts created with these frameworks are systematically listed in the Consensys Diligence Smart Contract Weakness List. The need for continued monitoring and improved safety protocols is evident.

Testing and Sandboxing Proactive Flaw Detection

Testing in multiple scenarios and deployment in controlled environments might reveal unusual situations and flaws before real-world implementation. Truffle and Hardhat are dependable testing frameworks, while Remix allows secure experimentation. Contracts with 10-20% non-audits, according to Ethereum Security Project research. This highlights a lack of proactive security [11,12].

C. Tools and methodologies for Smart Contract security and privacy analysis

Without a thorough code review, vulnerabilities could cause money losses and privacy violations. Researchers are continually creating tools and methods to secure and protect these vital components. Static analysis, dynamic analysis, and formal verification have produced several popular tools.

Table III: The prevailing static tools employed for smart contracts are [6][10][13][14]

Tool Name	EVM-code Input	Solidity-Code was taken as Input	Support both	Platform on	The method utilized for Analysis
ESCORT	Yes	No	Yes	-	By Machine-learning
Ether(S-GRAM)	Yes	No	Yes	On Python	By Machine -learning
GASOL	Yes	Yes	Yes	-	By Code-Instrumentation
SAFEVM	Yes	Yes	Yes	On Python	By Constraint-Solving, Symbolic-Execution
SIF	No	Yes	No	On C++	By Code-Instrumentation
Slither	No	Yes	No	On Python	By Constraint-Solving, Code -Transformation
Smartbugs	No	Yes	No	On Python	By Machine-learning
Smartcheck	No	Yes	No	On Java	By Code-Transformation
SmartEmbed	No	Yes	No	On JavaScript	By Code-Transformation, Code-Instrumentation
E-EVM	Yes	No	No	On Python	By Symbolic-Execution
Mythril	Yes	No	No	On Python	By Constraint-Solving, Symbolic-Execution
Octopus	Yes	No	No	On Python	By Symbolic-Execution, Fuzz-Testing
Osiris	Yes	No	No	On Python	By Symbolic-Execution
Oyenet	Yes	No	No	On Python	By Symbolic-Execution

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 9, September 2024

Securify	Yes	No	No	On Java	By Abstract-Interpretation

Table IV: The prevailing dynamic tools utilized for smart contracts are [6,10,13,14,15]

Tools used	**Using EVM- code as Input**	**Using Solidity Code as Input**	**Support both of them**	**Platform**	**Method utilised for Analysis**
ContractLarva	No	Yes	No	Haskell and Tex	By Code Instrumentation
Ethlint	No	Yes	No	Java Script	By Code Instrumentation
Harvey	No	Yes	No	–	By Fuzz testing
ContractGuard	Yes	No	No	Java Script	By Machine Learning
EthBMC	Yes	No	No	Rust	By Symbolic Execution
EVMFuzz	Yes	No	No	Python	By Fuzz testing
Manticore	Yes	No	No	Python	By Symbolic Execution
EASYFLOW	Yes	Yes	Yes	Go	By Taint Analysis
ReGuard	Yes	Yes	Yes	Python	By Fuzz testing

V. INTEGRATING MACHINE LEARNING INTO SMART CONTRACT SECURITY

Over the past few years, the widespread adoption of Blockchain technology has brought significant changes to multiple industries, providing secure and transparent decentralized systems. Nevertheless, smart contracts which are self-executing contracts deployed on Blockchain platforms such as Ethereum, can be vulnerable to various security risks and vulnerabilities. Incorporating machine learning (ML) methods into smart contract security offers a hopeful strategy to reduce these risks and improve network security as a whole.

A. Rational to Support Machine Learning Techniques

- Machine learning techniques can adjust to evolving threats and analyze complex patterns in smart contract code and transaction data, resulting in enhanced security measures.
- ML algorithms allow for the ongoing monitoring of smart contracts, identifying irregularities, and forecasting possible security breaches.
- Machine learning models can identify patterns associated with security vulnerabilities in newly created contracts by analyzing extensive datasets of known vulnerabilities.
- ML-powered security solutions provide the capability to automate the analysis of large datasets and seamlessly adjust to the constantly evolving nature of decentralized systems, leading to enhanced scalability and efficiency [15,16].

B. Integration of ML Model to Secure Smart Contract

Machine learning (ML) models in smart contract security frameworks improve Blockchain ecosystem security risk detection and mitigation. Compared to previous approaches, machine learning-based security solutions reduced security incident detection time by 30%, according to IBM. ML models increase smart contract security by discovering vulnerabilities, abnormalities, predictive analysis, and real-time monitoring and response [17].

C. Real-time Monitoring and Response

Monitoring and response systems play a crucial role in ensuring the security of smart contracts. These systems

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 9, September 2024

enable the timely identification and resolution of security concerns in Blockchain ecosystems. As an illustration, McAfee Labs made an interesting finding regarding the effectiveness of real-time monitoring systems. By utilizing machine learning algorithms, these systems were able to achieve an impressive accuracy rate of 95% in swiftly detecting harmful activity as soon as it happened.

Table V: Tools and methodologies for Smart Contract security and privacy analysis [14,19,20,21]

Model	Decision Tree	Random Forest	Support Vector Machine (SVM)	Logistic Regression
Problem Domain	Classification of vulnerabilities	Classification of vulnerabilities	Classification of specific vulnerabilities	Binary classification (vulnerable vs. non-vulnerable)
Performance Metrics	Accuracy, Precision, Recall, F1 Score	Accuracy, Precision, Recall, F1 Score	Accuracy, Precision, Recall, F1 Score	Accuracy, Precision, Recall
Loss Function	Gini Impurity, Entropy	Cross-Entropy Loss	Hinge Loss	Logistic Loss
Training Time	Fast	Fast	Moderate	Fast
Data Efficiency	High	High	Moderate	Moderate
Overfitting/ Underfitting	Prone to overfitting	Less prone to overfitting	Susceptible to overfitting	Prone to overfitting with complex data
Production-Based Parameters	Not highly scalable, Interpretable	Scalable, Not Interpretable	Scalable (with Kernels), Interpretable (with feature selection)	Scalable, Interpretable
Deep Learning Models (RNN, CNN)	Clustering Algorithms	Generative Adversarial Networks (GANs)	
Sequence or code analysis for vulnerability detection	Grouping similar contracts (may identify anomaly-based vulnerabilities)	Anomaly detection (identify contracts deviating from "normal" patterns)	
Accuracy, Precision, Recall, F1 Score	Silhouette Score, Calinski-Harabasz Index	Inception Score, Fréchet Inception Distance	
Cross-Entropy Loss	N/A (depends on algorithm)	N/A (adversarial loss function)	
Slow	Moderate	Slow	
High	Moderate	High	
Prone to overfitting on small datasets	Not directly applicable	Requires large, clean datasets	
Moderate Scalability, Not Interpretable	Not suitable for classification, interpretable (depending on algorithm)	Not suitable for direct classification, Not Interpretable	

VI. CONCLUSION & FUTURE WORK

In conclusion, this paper examines how machine learning (ML) might improve Blockchain smart contract security. We investigated a variety of techniques and methodologies that use machine learning to discover vulnerabilities, unusual behavior, and potential threats. We have investigated and provided information on machine learning-based smart contract vulnerability detection, anomaly detection, and threat prediction tools. ML can improve security processes, according to relevant literature and real-world instances. Smart contract development can be dynamic to protect against emerging threats and reduce vulnerability risks by using ML models at run time. Additional investigation is required to enhance the ML algorithms so they can effectively identify and address previously unknown vulnerabilities and complex attack techniques in smart contracts. Now more focus should be on the development of an ML framework that can analyze the behavioral patterns of smart contracts and anticipate potential security risks by leveraging historical data and adapting to evolving attack strategies.

REFERENCES

[1] Abdelfattah, F., Al Mashaikhya, N.Y., Dahleez, K.A. and El Saleh, A. 2024. A systematic review of e-learning systems adoption before and during the COVID-19. *Global Knowledge, Memory and Communication*, *73*(3), pp.292-311.

[2] Lykourentzou, I., Giannoukos, I., Nikolopoulos, V., Mpardis, G. and Loumos, V. 2009. Dropout prediction in e-learning courses through the combination of machine learning techniques. *Computers & Education*, *53*(3), pp.950-965.

[3] Ezaldeen, H., Misra, R., Alatrash, R. and Priyadarshini, R. 2019. Machine learning-based improved recommendation model for E-learning. In *2019 International Conference on Intelligent Computing and Remote Sensing (ICICRS)* (pp. 1-6). IEEE.

[4] Geeksforgeeks. 2024. Types of machine learning. Available online: https://www.geeksforgeeks.org/types-of-machine-learning/ Accessed on 2 August 2024.

[5] Khanal, S.S., Prasad, P.W.C., Alsadoon, A. and Maag, A. 2020. A systematic review: machine learning based recommendation systems for e-learning. *Education and Information Technologies*, *25*(4), pp.2635-2664.

[6] Aymane, E.Z.Z.A.I.M., Aziz, D.A.H.B.I., Abdelfatteh, H.A.I.D.I.N.E. and Abdelhak, A.Q.Q.A.L. 2024. Enabling Sustainable Learning: A Machine Learning Approach for an Eco-friendly Multi-factor Adaptive E-Learning System. *Procedia Computer Science*, *236*, pp.533-540.

[7] Sekeroglu, B., Dimililer, K., & Tuncal, K. 2019. Student performance prediction and classification using machine learning algorithms. In Proceedings of the 2019 8th International Conference on Educational and Information Technology (pp. 7-11). ACM.

[8] Farhat, R., Mourali, Y., Jemni, M. and Ezzedine, H. 2020. An overview of machine learning technologies and their use in e-learning. In *2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA)* (pp. 1-4). IEEE.

[9] Lu, D.N., Le, H.Q. and Vu, T.H. 2020. The factors affecting acceptance of e-learning: A machine learning algorithm approach. *Education Sciences*, *10*(10), p.270.

[10] Liu, X. and Ardakani, S.P. 2022. A machine learning-enabled effective E-learning system model. *Education and Information Technologies*, *27*(7), pp.9913-9934.

[11] Krippendorff, K. 2012. Content Analysis: An introduction to its methodology (3rd ed.). Thousand Oaks, CA: Sage.

[12] Stemler, S.E. 2015. Content analysis. *Emerging trends in the social and behavioural sciences: An Interdisciplinary, Searchable, and Linkable Resource*, pp.1-14.

[13] Krippendorff, K. 1989. Content analysis. *International encyclopedia of communication*, *1*(1), pp.403-407.

[14] Ahmed, E. 2024. Student Performance Prediction Using Machine Learning Algorithms. *Applied Computational Intelligence and Soft Computing*, *2024*(1), p.4067721.