Vol-08 Issue 05, May -2024 ISSN: 2456-9348
Impact Factor: 7.936

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

A SMART CONTRACT VULNERABILITY DETECTION MECHANISM BASED ON DEEP LEARNING AND EXPERT RULES

Dhatrik Sujan Kuma, Addepati Amrutha Varshini , Yasani Samara Simha Reddy,
UG scholars, Department of Computer Science Engineering (Cyber Security),
Guru Nanak Institutions Technical Campus (Autonomous), Hyderabad, India

Mrs. G.Usha Rani
Assistant Professor, Department of Computer Science Engineering (Cyber Security),
Guru Nanak Institutions Technical Campus (Autonomous), Hyderabad, India

ABSTRACT

Traditional techniques for smart contract vulnerability detection rely on fixed expert criteria to discover vulnerabilities, which are less generalizable, scalable, and accurate. Deep learning algorithms help to address these issues, but most fail to encode true expert knowledge and remain interpretable. In this paper, we present a smart contract vulnerability detection mechanism that operates in phases with graph neural networks and expert patterns in deep learning to mutually address the deficiencies of the two detection approaches and improve smart contract vulnerability detection capabilities. Experiments show that our vulnerability detection mechanism outperforms the original deep learning model by an average of 6 points in detecting vulnerabilities and that the second stage of the checking mechanism can also block contract transactions containing dangerous actions at the Ethernet Virtual Machine (EVM) level and generate error reports for submission. This strategy helps to construct more stable smart contracts and to create a secure environment for smart contracts

Keywords:
Smart Contract Vulnerabilities, Detection Methods, Conventional Approaches, Expert Criteria, Deep Learning Algorithms, Neural Networks, Graph Models, Vulnerability Detection Mechanism, Testing Results, Risky Activities, Transaction Blocking, Ethereum Network, Verification Mechanism, Scalability, Accuracy, Generalizability.

1.INTRODUCTION

Blockchain and digital currency have garnered significant attention due to the maturation of blockchain technology and the rapid development of cryptocurrencies. The synergy of decentralized consensus protocols and proof-of-work mechanisms ensures decentralized operations, transaction transparency, and tamper-evident properties. These advancements have led to the creation of smart contracts, which facilitate a wide range of applications and services. However, the increasing complexity of smart contracts has introduced significant security challenges due to frequent vulnerabilities. Despite mechanisms like Ethereum's Create2 that allow for contract updates, the maintenance costs and irreversible damage from attacks necessitate thorough security evaluations of smart contracts before deployment. And Current vulnerability detection methods are divided into traditional approaches based on expert rules and automated tools, and deep learning-based methods. Traditional methods, exemplified by tools like Oyente and Securify, rely on predefined guidelines that are prone to error and struggle to meet growing demands. Deep learning models, although more efficient and generalizable, suffer from dependency on datasets and lack interpretability. To improve detection using graph neural networks (GNNs) alongside expert models to detect smart contract vulnerabilities and block dangerous transactions at the Ethereum Virtual Machine (EVM) level. This approach aims to create a comprehensive vulnerability detection mechanism with enhanced detection capabilities and ability to prevent risky operations. This paved the way for smart contracts that have enabled numerous applications and services. Unfortunately, smart contracts have come under the limelight for all the wrong reasons with increasing sophistication in their architecture leading to repeated vulnerabilities. Even though Ethereum recently introduced Create2, a mechanism that does enable the smart contract to be updated after it has been created, the maintenance fees and collateral damage from being compromised in an attack combined with irreversible changes pushes for a full-fledge security audit of their smart contracts before going live. The existing methods of vulnerability detection are classically traditional

2. LITERATURE SURVEY

A. Joshi, H. Kebriaei, V. Mariani, and L. Glielmo propose a method to detect smart contract vulnerabilities by combining graph neural networks with expert knowledge. They explain how they transform the source code into a contract graph, normalize it, and then employ a temporal message propagation network to extract features. Results show significant accuracy improvements over existing methods, especially for reentrancy, timestamp dependence, and infinite loop vulnerabilities.

N. V. Chawla and D. A. Davis introduce Hessian graph convolutional networks (HesGCN), addressing the poor extrapolating ability of traditional GCN due to unchanged null space of Laplacian along the manifold. HesGCN optimizes one-order spectral graph Hessian convolutions, incorporating richer null space information to improve feature learning. Experimental results validate its superiority over existing methods in semi-supervised learning tasks.

R. Sanchez-Marquez and J. M. J. Vivas systematically review security vulnerabilities in Ethereum blockchain smart contracts, emphasizing their significance due to potential financial losses. They discuss detection tools, real-life attacks, and preventive mechanisms. Comparisons among analysis tools are provided, and future research directions are suggested to address issues in Ethereum blockchain-based smart contracts

S. He, Y. Lu, Q. Tang, G. Wang, and C. Q. Wu provide an overview of blockchain technology, focusing on its decentralization, security, and trustworthiness. They introduce blockchain classification and architecture, analysing its core principles and function in each layer, particularly in relation to Bitcoin. The paper discusses the current level of blockchain technology development and applications, along with the challenges and future trends.

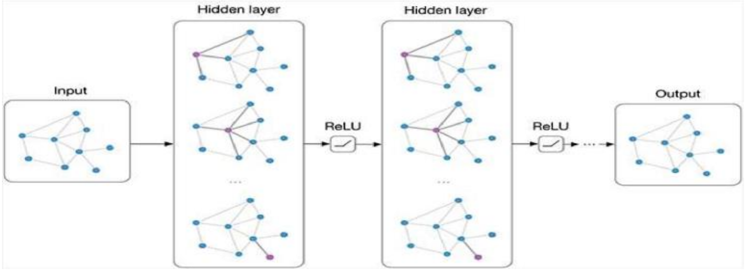
Y. Ni, C. Zhang, and T. Yin address security vulnerabilities in smart contracts, crucial for driving Web 3.0 in areas like digital finance. They categorize vulnerabilities and analyse detection techniques, comparing existing tools based on various criteria. The paper presents the Co-Governed Sovereignty Multi-Identifier Network (CoG-MIN) as a case study, emphasizing the importance of smart contract security in Web 3.0, and suggests future research directions.

Y. Zhang and J. Ma focus on Ethereum Solidity smart contracts, outlining their mechanisms, common vulnerabilities, and traditional detection tools such as symbolic execution and formal verification. They also discuss recent machine learning-based vulnerability detection methods and suggest improving detection efficiency and standardizing information databases.

3. PROPOSED SYSTEM

Our proposed system leverages Graph Neural Networks (GNN) in deep learning to enhance the detection of vulnerabilities in smart contracts. By collaborating with expert models, this approach aims to improve both detection efficiency and generalization. Combining traditional expert rules with deep learning methods helps to address the interpretability concerns often associated with deep learning. In this system, the GNN model and expert model work synergistically to provide a more robust screening of smart contracts. Additionally, specific constraint rules are introduced for certain vulnerability types, enabling the system to block dangerous transactions at the Ethereum Virtual Machine (EVM) level. This integration of GNNs with expert patterns offers a comprehensive solution to smart contract vulnerability detection and mitigation.

4. SYSTEM ARCHITECTURE



IJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

5. REQUIREMENTS

The hardware requirements serve as a comprehensive specification for the system implementation and are essential for initiating the system design process. They outline what the system should encompass rather than dictating how it should be executed.

- Processor: Dual-core Intel Core 2 Duo
- RAM: 4GB DDR RAM
- Hard Disk: 250GB Storage Capacity

6. SOFTWARE REQUIREMENTS

The software requirements document serves as a specification for the system, outlining what the system should accomplish rather than detailing the methods for achieving it. It provides a foundation for creating the software requirements specification and is instrumental in estimating costs, planning team activities, performing tasks, and tracking progress throughout the development process.

- Operating System: Windows 7/8/10
- Platform: Spyder3
- Programming Language: Python
- Front End: Spyder3

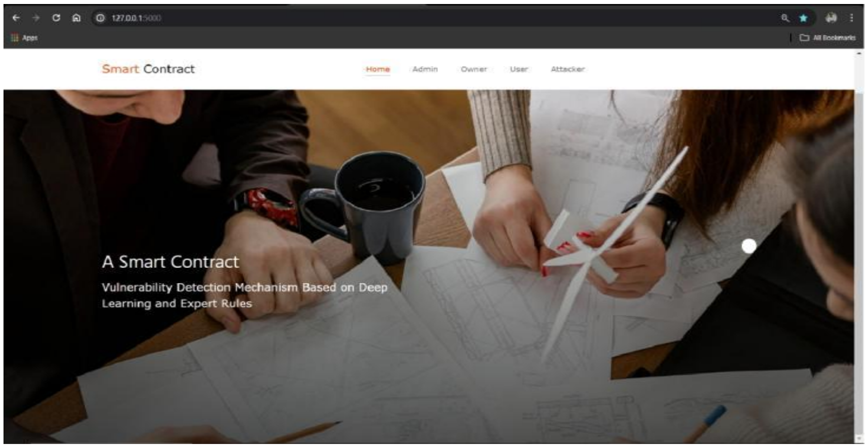
7. FUTURE ENHANCEMENTS

[1]. Advanced Graph Neural Networks (GNNs): Continued research and development in GNNs could yield more sophisticated models capable of identifying even more subtle vulnerabilities within smart contracts.

[2]. Hybrid Approaches: Integrating traditional expert rules with machine learning techniques could offer a more holistic approach to vulnerability detection. By harnessing the strengths of both methods—such as the transparency of expert rules and the efficiency of machine learning models—the overall detection capabilities could be significantly enhanced.

[3]. Incorporation of Formal Verification: Formal verification techniques could be incorporated into the vulnerability detection process to mathematically prove the correctness of smart contracts and identify potential vulnerabilities at an early stage of development.

8. SNAPSHOTS

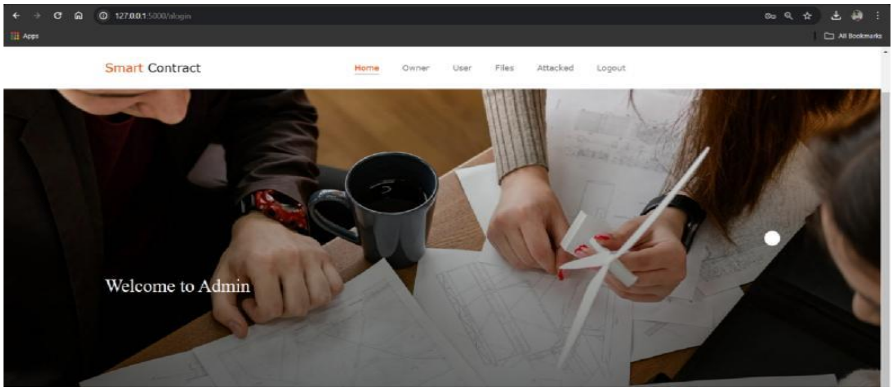


A Smart Contract

Vulnerability Detection Mechanism Based on Deep Learning and Expert Rules

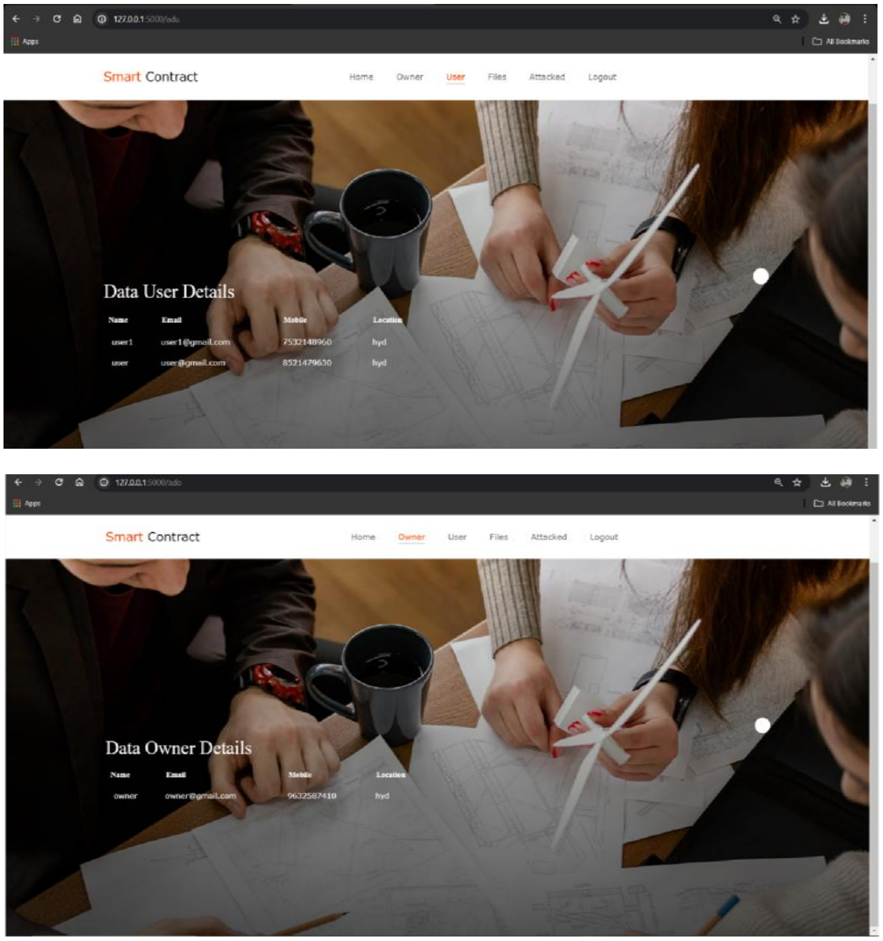
Figure 1 HOME PAGE

IJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/



Welcome to Admin

Figure 2 LOGIN PAGE



Data User Details

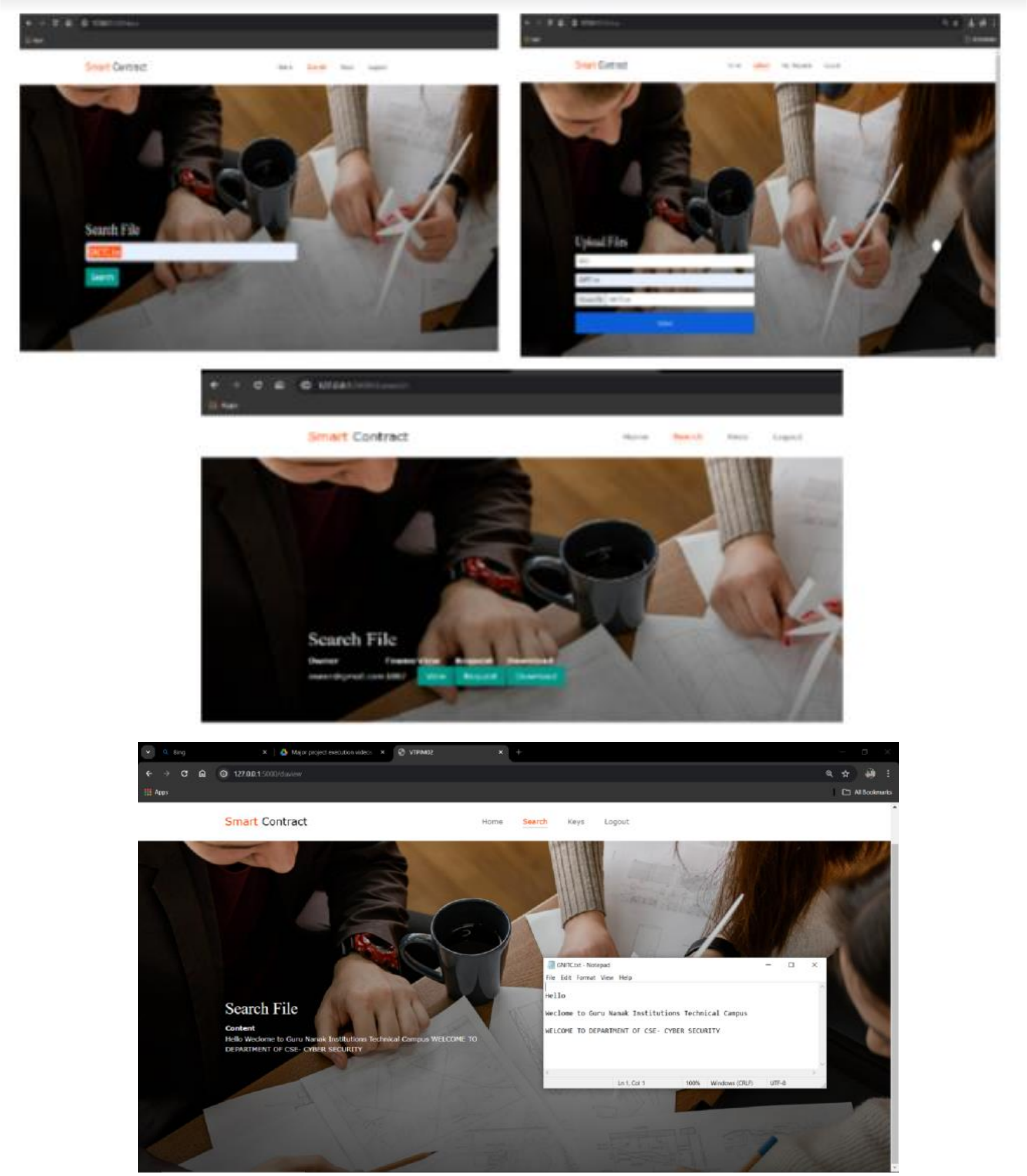
Name	Email	Mobile	Location
user1	user1@gmail.com	7532148960	hyd
user	user@gmail.com	8521479630	hyd

Data Owner Details

Name	Email	Mobile	Location
owner	owner@gmail.com	9632587410	hyd

Figure 3 OWNER & USER DETAILS

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/



Search File

Content

Hello Weclome to Guru Nanak Institutions Technical Campus WELCOME TO DEPARTMENT OF CSE- CYBER SECURITY

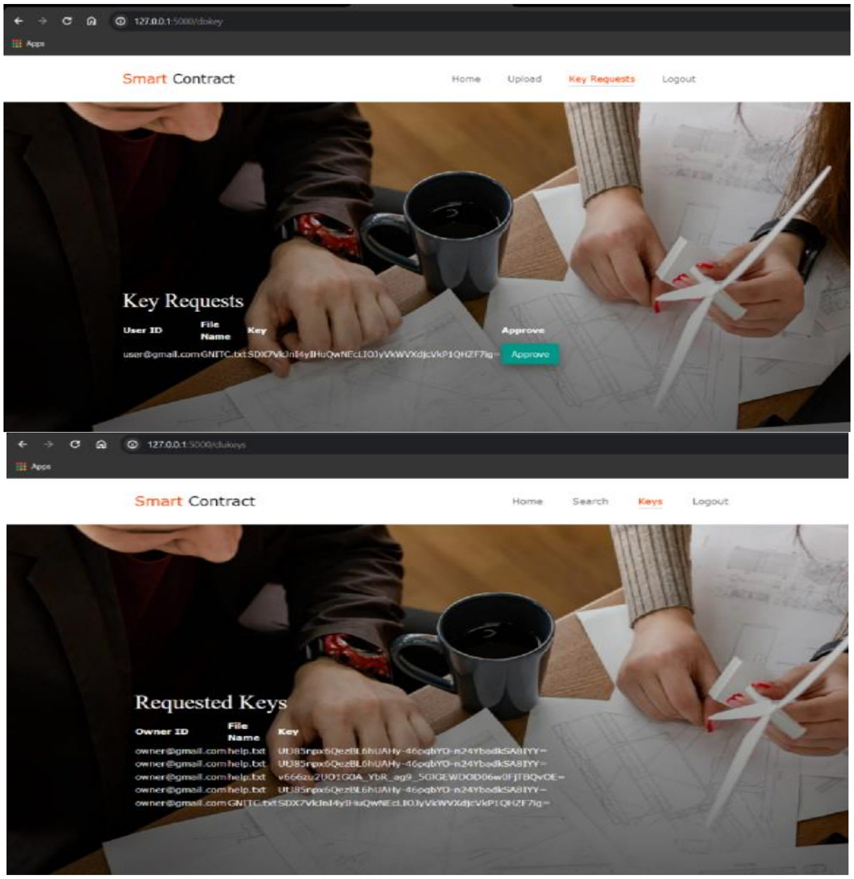
```
Hello

Weclome to Guru Nanak Institutions Technical Campus

WELCOME TO DEPARTMENT OF CSE- CYBER SECURITY
```

Figure 4 UPLOAD, VIEW & DOWNLOAD OF FILE

IJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/



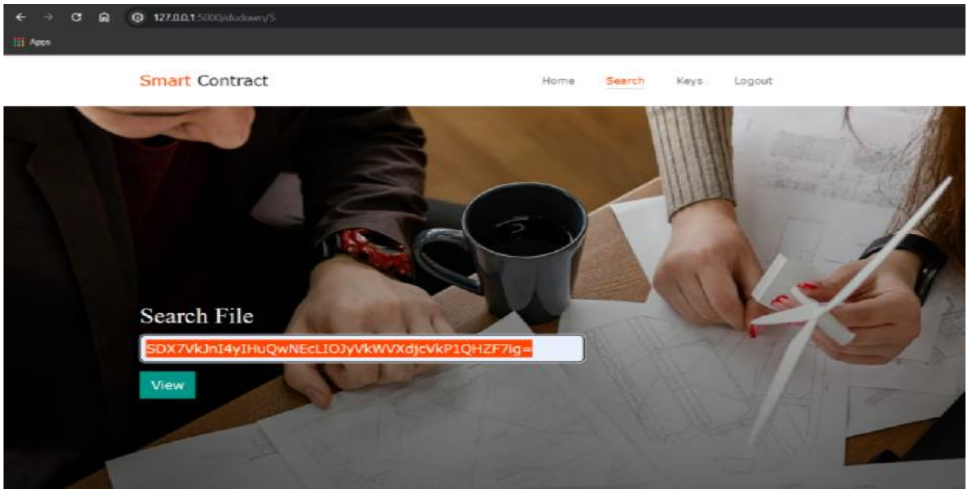
Key Requests

User ID	File Name	Key	Approve
user@gmail.com	GNITC.txt	SDX7VkJnI4yIHuQwNEcLIOJyVkWVXdjcVkP1QHZF7ig=	Approve

Requested Keys

Owner ID	File Name	Key
owner@gmail.com	help.txt	Ut385npx6QezBL6hUAHy-46pqbYO-n24YbadkSA8IYY=
owner@gmail.com	help.txt	Ut385npx6QezBL6hUAHy-46pqbYO-n24YbadkSA8IYY=
owner@gmail.com	help.txt	v666zu2UO1G0A_YbR_ag9_5GIGEWDOD06w0FJTBQvOE=
owner@gmail.com	help.txt	Ut385npx6QezBL6hUAHy-46pqbYO-n24YbadkSA8IYY=
owner@gmail.com	GNITC.txt	SDX7VkJnI4yIHuQwNEcLIOJyVkWVXdjcVkP1QHZF7ig=

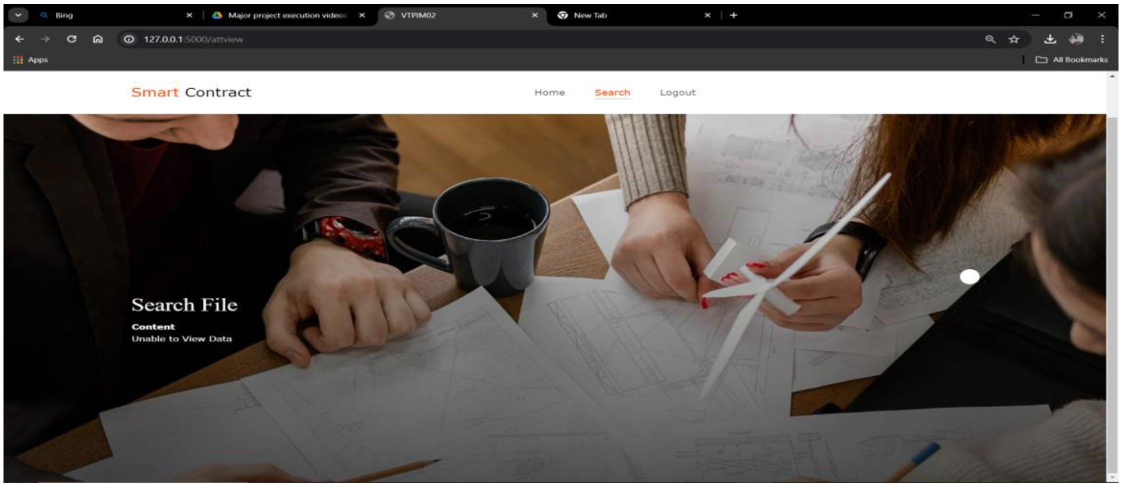
Figure 5 APPROVE REQUEST AND 6 KEY DETAILS



Search File

SDX7VkJnI4yIHuQwNEcLIOJyVkWVXdjcVkP1QHZF7ig=

Figure 6 ATTACKER – TRY TO ACCESS THE DATA WITH KEY

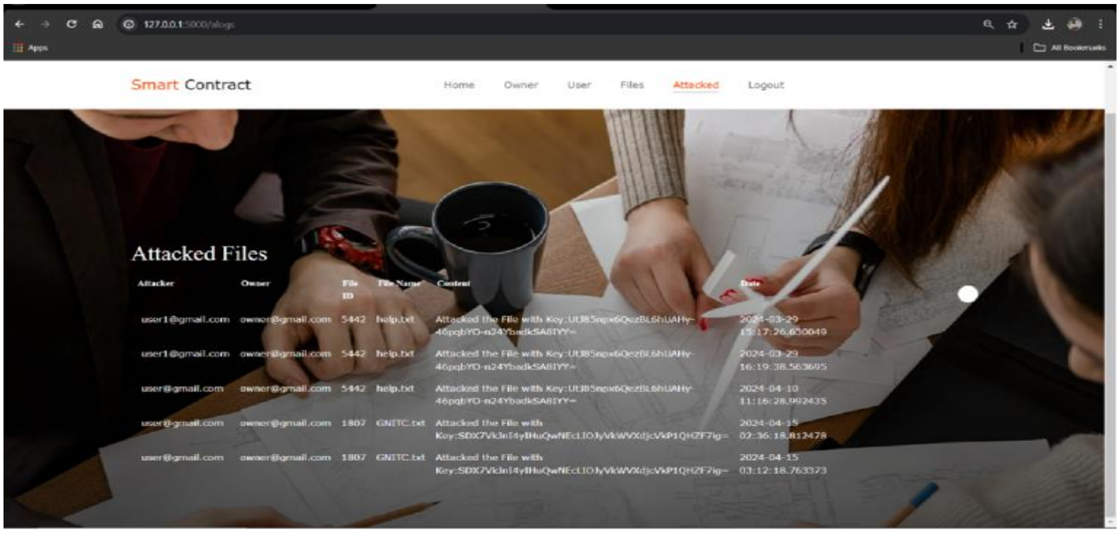


Search File

Content

Unable to View Data

Figure 7 AFTER ATTACKER KEY REQUEST



Attacked Files

Attacker	Owner	File ID	File Name	Content	Date
user1@gmail.com	owner@gmail.com	5442	help.txt	Attacked the File with Key:UtJ85npx6QezBL6hUAHy-46pqbYO-n24YbadkSA8IYY=	2024-03-29 15:17:26.630049
user1@gmail.com	owner@gmail.com	5442	help.txt	Attacked the File with Key:UtJ85npx6QezBL6hUAHy-46pqbYO-n24YbadkSA8IYY=	2024-03-29 16:19:38.563695
user@gmail.com	owner@gmail.com	5442	help.txt	Attacked the File with Key:UtJ85npx6QezBL6hUAHy-46pqbYO-n24YbadkSA8IYY=	2024-04-10 11:16:28.992435
user@gmail.com	owner@gmail.com	1807	GNITC.txt	Attacked the File with Key:SDX7VkJnI4yIHuQwNEcLIOJyVkWVXdjcVkP1QHZF7ig=	2024-04-15 02:36:18.812478
user@gmail.com	owner@gmail.com	1807	GNITC.txt	Attacked the File with Key:SDX7VkJnI4yIHuQwNEcLIOJyVkWVXdjcVkP1QHZF7ig=	2024-04-15 03:12:18.763373

Figure 8 ADMIN ALERT

9. CONCLUSION

In this paper, we study the work of combining GNN models with expert models working at the EVM level for smart contract vulnerability detection. The smart contract vulnerability detection technique presented in this work improves both its detection capabilities as a vulnerability detection tool and its capacity to function as a stopper for contracts performing risky operations. Although the time overhead will be more than the EVM without the second phase of the protection mechanism, the mechanism provides the user with the option of whether or not to activate the second phase of the protection mechanism to prevent wasting resources like time and gas. We believe that the work in this paper will play a significant role in the future development of a more secure and reliable smart contract environment**.**

10. REFERENCES

[1] J. Cheng, ‘‘Current status and prospects of blockchain technology,’’ in Proc. Int. Conf. Artif. Intell. Secur. Singapore: Springer, 2020, pp. 674–684.

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

[2] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, ‘‘HotStuff: BFT consensus with linearity and responsiveness,’’ in Proc. ACM Symp. Princ. Distrib. Comput., Jul. 2019, pp. 347–356.
[3] Y. Ni, C. Zhang, and T. Yin, ‘‘A review of research on smart contract security vulnerabilities,’’ J. Inf. Secur., vol. 5, no. 3, pp. 78–99, 2020.
[4] V. Buterin, ‘‘A next-generation smart contract and decentralized application platform,’’ White Paper, vol. 3, no. 37, p. 37, 2014.
[5] S. Badruddoja, R. Dantu, Y. He, K. Upadhayay, and M. Thompson, ‘‘Making smart contracts smarter,’’ in Proc. IEEE Int. Conf. Blockchain Cryptocurrency (ICBC), May 2021, pp. 1–3.
[6] P. Tsankov, A. Dan, D. Drachsler-Cohen, A. Gervais, F. Bünzli, and M. Vechev, ‘‘Securify: Practical security analysis of smart contracts,’’ in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., Oct. 2018, pp. 67–82.
[7] A. Warnecke, D. Arp, C. Wressnegger, and K. Rieck, ‘‘Evaluating explanation methods for deep learning in security,’’ in Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P), Sep. 2020, pp. 158–174.
[8] Y. Zhuang, ‘‘Smart contract vulnerability detection using graph neural network,’’ in Proc. IJCAI, 2020, pp. 3283–3290.
[9] J. Feist, G. Grieco, and A. Groce, ‘‘Slither: A static analysis framework for smart contracts,’’ in Proc. IEEE/ACM 2nd Int. Workshop Emerg. Trends Softw. Eng. Blockchain (WETSEB), May 2019, pp. 8–15.
[10] J. Gao, H. Liu, C. Liu, Q. Li, Z. Guan, and Z. Chen, ‘‘EASYFLOW: Keep Ethereum away from overflow,’’ in roc. IEEE/ACM 41st Int. Conf. Softw. Eng., Companion (ICSE-Companion), May 2019, pp. 23–26.
[11] C. F. Torres and M. Steichen, ‘‘The art of the scam: Demystifying honeypots in Ethereum smart contracts,’’ in Proc. 28th USENIX Secur. Symp. (USENIX Secur.) Santa Clara, CA, USA: USENIX Assoc., 2019, pp. 1591–1607.
[12] R. Baldoni, E. Coppa, D. C. D’elia, C. Demetrescu, and I. Finocchi, ‘‘A survey of symbolic execution techniques,’’ ACM Comput. Surv., vol. 51, no. 3, pp. 1–39, May 2019. 61 Dept of CSE – CYBER SECURITY
[13] Y. Fu, M. Ren, F. Ma, H. Shi, X. Yang, Y. Jiang, H. Li, and X. Shi, ‘‘EVMFuzzer: Detect EVM vulnerabilities via fuzz testing,’’ in Proc. 27th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng., Aug. 2019, pp. 1110–1114.
[14] N. Ashizawa, N. Yanai, J. P. Cruz, and S. Okamura, ‘‘Eth2 Vec: Learning contract-wide code representations for vulnerability detection on Ethereum smart contracts,’’ in Proc. 3rd ACM Int. Symp. Blockchain Secure Crit. Infrastruct., May 2021, pp. 47–59.
[15] W. Wang, J. Song, G. Xu, Y. Li, H. Wang, and C. Su, ‘‘ContractWard: Automated vulnerability detection models for Ethereum smart contracts,’’ IEEE Trans. Netw. Sci. Eng., vol. 8, no. 2, pp. 1133–1144, Apr. 2021