Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Human-in-the-Loop Framework for Detecting Logical Vulnerabilities in Smart Contracts

Alireza Assadzadeh

alireza.asadzadeh@iaut.ac.ir

Islamic Azad University Tabriz Branch https://orcid.org/0009-0008-1797-3606

Research Article

Keywords: Blockchain security, Consensus protocols, Smart contracts, Logical vulnerabilities, Human-in-the-Loop

Posted Date: November 10th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11657430/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

A Human-in-the-Loop Framework for Detecting Logical Vulnerabilities in Smart Contracts

Alireza Assadzadeh
Department of Computer Engineering, Islamic Azad University, Tabriz Branch, Iran
Alireza.asadzadeh@iaut.ac.ir

Abstract

Blockchain systems face a wide range of security challenges that span from the consensus layer to the application layer. While many studies have examined consensus protocols or smart contract vulnerabilities separately, fewer works have analyzed these issues together or considered the role of human involvement in the detection process. This paper presents a focused study of security challenges in blockchain, progressing from consensus mechanisms to logical vulnerabilities in smart contracts, and examining the potential of Human-in-the-Loop (HITL) approaches. We review the main families of consensus protocols and highlight their characteristic security trade-offs. We then analyze logical vulnerabilities, which remain difficult for purely automated tools to detect reliably due to their context-dependent nature. Next, we propose a simple HITL framework that combines automated analysis with targeted human review. Experiments on the SmartBugs Curated dataset show that the proposed approach improves precision from 0.7125 to 0.8382 and F1-score from 0.8321 to 0.9120, while reducing false positives by 52.2% without any loss in recall. The results suggest that integrating human expertise can meaningfully improve the detection of logical vulnerabilities. The paper also identifies several research gaps and outlines directions for future work.

Keywords: Blockchain security; Consensus protocols; Smart contracts; Logical vulnerabilities; Human-in-the-Loop

1. Introduction

Blockchain technology has become a foundational infrastructure for decentralized applications in finance, supply chain, healthcare, and governance [1], [2]. Its core properties decentralization, immutability, transparency, and cryptographic security have attracted significant attention from both academia and industry [3], [4]. However, the rapid growth of blockchain systems has also revealed a range of complex security challenges that continue to threaten their reliability and trustworthiness [5], [6].

At the core of every blockchain lies the consensus protocol, which enables distributed nodes to agree on the state of the ledger [7]. Traditional mechanisms such as Proof-of-Work (PoW) and Proof-of-Stake (PoS) have been widely studied, yet they still face fundamental trade-offs between security, scalability, energy efficiency, and decentralization [8], [9]. Hybrid consensus designs have been proposed to address some of these limitations, but they often introduce new attack surfaces and operational complexities [10], [11]. As a result, a growing body of research has focused on analyzing the formal security properties, fault tolerance, and practical vulnerabilities of different consensus families [12], [13].

Beyond the consensus layer, smart contracts have become a critical component of modern blockchain platforms. These self-executing programs encode business logic and manage digital assets autonomously [14]. While smart contracts offer clear automation benefits, they are also highly susceptible to security flaws. In particular, logical vulnerabilities those arising from flawed business logic, incorrect assumptions, or incomplete access control have proven especially difficult to detect through conventional automated tools [15], [16]. High-profile exploits have repeatedly shown that even carefully audited contracts can contain subtle logical errors capable of causing substantial financial losses [17], [18].

Existing automated analysis techniques, including static analysis, symbolic execution, fuzzing, and machine learning-based detectors, have made notable progress in identifying well-known vulnerability patterns such as reentrancy and

integer overflows [19], [20]. Nevertheless, these approaches often struggle with context-dependent logical flaws that require a deeper semantic understanding of the contract's intended behavior [21], [22]. Recent empirical studies have further shown that current tools prevent only a limited fraction of real-world attacks, particularly those involving complex business logic [23].

In response to these limitations, researchers have begun exploring Human-in-the-Loop (HITL) approaches that combine automated analysis with human expertise [24], [25]. By integrating developer insight, visual abstraction, and interactive decision-making, HITL frameworks aim to improve the detection of non-trivial logical vulnerabilities while remaining practical to use [26], [27]. Similar ideas have also appeared in broader cybersecurity contexts, where human oversight is considered essential for high-stakes decision-making [28], [29]. Despite these promising directions, the literature on HITL methods specifically tailored to smart contract security remains fragmented and lacks a unified analytical perspective.

Although numerous surveys have examined consensus mechanisms [30], [31] or smart contract vulnerability detection [32], [33], few works have systematically connected these two layers while incorporating the emerging role of human involvement. Moreover, existing reviews rarely provide a structured taxonomy that spans from consensus-level threats to logical vulnerabilities and HITL-based mitigation strategies. This gap limits the ability of researchers and practitioners to develop more holistic security solutions.

To address this shortcoming, the present paper provides a focused analysis of security challenges in blockchain systems, progressing from consensus protocols to logical vulnerabilities in smart contracts, and examining the potential of Human-in-the-Loop approaches. The main contributions of this work are as follows:

- We present a structured overview of security issues in major consensus protocols, highlighting their strengths, weaknesses, and open challenges.
- We analyze logical vulnerabilities in smart contracts, emphasizing the limitations of purely automated detection methods.
- We review existing Human-in-the-Loop approaches in blockchain security and identify their current capabilities and gaps.
- We propose and empirically evaluate a simple HITL framework on a curated dataset of vulnerable smart contracts, demonstrating a substantial reduction in false positives while preserving detection coverage.
- We outline a unified perspective that connects these layers and suggest promising directions for future research.

The remainder of this paper is organized as follows. Section 2 provides essential background. Section 3 presents the proposed HITL framework. Section 4 describes the experimental setup and results. Section 5 concludes the paper.

2. Background

This section briefly introduces the main concepts needed to understand the rest of the paper. We cover blockchain architecture, consensus mechanisms, smart contracts, logical vulnerabilities, and the idea of Human-in-the-Loop systems.

2.1 Blockchain Architecture

A blockchain can be viewed as a distributed ledger that records transactions in an append-only chain of blocks. Each block contains a set of transactions, a timestamp, and a cryptographic link to the previous block, which together help ensure data integrity and immutability. The network operates in a peer-to-peer fashion, removing the need for a central authority to validate or store information [7], [12].

Two main categories of blockchain networks are commonly distinguished. Permissionless (public) blockchains allow any participant to join and take part in consensus, while permissioned (private or consortium) blockchains restrict participation to a known set of nodes. This distinction has direct implications for security, performance, and the choice of consensus protocol [10], [36].

2.2 Consensus Mechanisms

Consensus is the process through which distributed nodes agree on the next valid state of the ledger [7]. In permissionless settings, Proof-of-Work (PoW) and Proof-of-Stake (PoS) remain the most widely deployed solutions [8], [9]. PoW relies on computational effort to deter attacks, whereas PoS assigns influence according to the amount of cryptocurrency held by participants.

Permissioned systems often adopt voting-based protocols such as Practical Byzantine Fault Tolerance (PBFT), which can achieve higher throughput and lower latency under the assumption of a partially trusted environment [10]. Hybrid designs that combine elements of different families have also been explored to balance security, scalability, and energy efficiency [9], [11]. Regardless of the specific algorithm, every consensus protocol must address the classic requirements of safety and liveness while tolerating a certain fraction of faulty or malicious nodes [12], [37].

2.3 Smart Contracts

Smart contracts are self-executing programs stored on the blockchain that automatically enforce the terms of an agreement when predefined conditions are met [14]. They enable complex decentralized applications by encoding business logic directly into code. Once deployed, a smart contract is generally immutable, which means that any flaw in its logic becomes permanently part of the system [15], [16].

Because smart contracts frequently control significant financial value, their correctness is critical. Even small mistakes in coding or design can lead to severe consequences, including the irreversible loss of assets [17], [18].

2.4 Logical Vulnerabilities

While many security tools focus on well-known patterns such as reentrancy or integer overflows, a particularly challenging class of problems arises from logical vulnerabilities [15], [16]. These flaws stem from incorrect assumptions about business rules, incomplete access-control policies, or mismatches between the intended behavior and the actual implementation [17], [21]. Unlike syntactic bugs, logical vulnerabilities are highly context-dependent and often require an understanding of the contract's purpose to be identified [22], [23].

Empirical evidence suggests that purely automated detectors still struggle with this category of defects, leaving a noticeable gap between tool capabilities and real-world attack patterns [17], [23].

2.5 Human-in-the-Loop Systems

Human-in-the-Loop (HITL) approaches seek to combine the speed and scalability of automated analysis with the contextual judgment of human experts [24], [25]. In the context of blockchain security, HITL methods may involve interactive visualization, expert validation of suspicious patterns, or collaborative decision-making between developers and analysis tools [26], [35]. The underlying premise is that certain complex judgments especially those involving business logic remain difficult for machines alone to perform reliably [33], [34].

Having established these foundational concepts, the following sections present the proposed framework and its empirical evaluation.

3. Proposed Method

In this section, we present a simple yet practical Human-in-the-Loop (HITL) framework designed to improve the detection of logical vulnerabilities in smart contracts. The main idea is to combine the speed of automated analysis tools with the contextual understanding of a human expert. The framework consists of three main stages: automated detection, confidence-based filtering, and human review.

3.1 Overall Framework

The proposed system works as follows. First, an automated analysis tool processes the smart contract and produces a list of potential issues. Each reported issue is assigned a confidence score. Only those issues whose confidence score exceeds a predefined threshold are passed to a human expert for final review. The expert then confirms or rejects each warning. The final set of accepted vulnerabilities is the result of this combined process. Figure 1 illustrates the overall architecture of the proposed framework.

Figure 1. Overview of the proposed Human-in-the-Loop framework

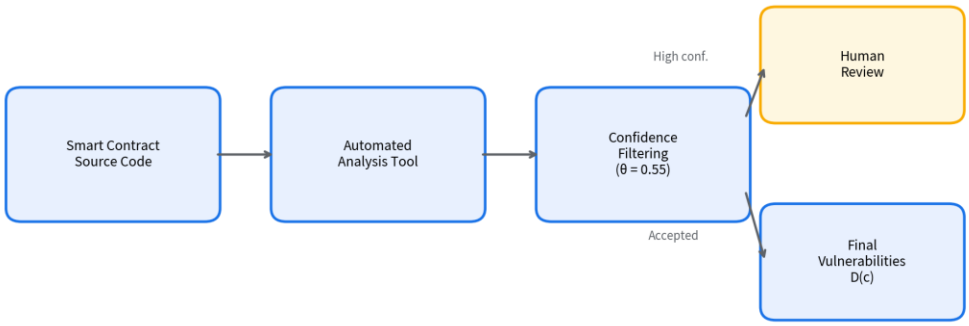


Figure 1. Overview of the proposed Human-in-the-Loop framework

Formally, let M(c) denote the set of potential issues reported by the automated tool for a given contract c. Let θ be the confidence threshold. The subset of issues sent to the human reviewer is defined as:

$$M\theta(c) = \{ v \in M(c) \mid confidence(v) \geq \theta \} \quad (1)$$

The final detection result after human review is expressed as:

$$D(c) = H(M\theta(c)) \quad (2)$$

where H(·) represents the human review function that accepts or rejects each warning [24], [35].

3.2 Automated Detection Stage

In the first stage, we use a static analysis tool (in our experiments, we simulate the behavior of tools such as those commonly used in the literature) to scan the smart contract source code. The tool reports potential vulnerabilities along with an estimated confidence score for each finding. Different categories of vulnerabilities tend to produce different confidence distributions. For example, well-studied patterns such as reentrancy often receive higher confidence scores, while access-control and other logic-related issues usually receive lower or more uncertain scores [15], [17], [21].

3.3 Confidence-Based Filtering

To reduce the cognitive load on the human expert, we apply a confidence threshold θ. In our experiments, we set θ = 0.55. Only warnings with a confidence score equal to or higher than this value are presented to the reviewer. This

filtering step significantly reduces the number of cases that require human attention while still covering the majority of high-risk findings [24], [35].

3.4 Human Review Stage

In the final stage, a human expert examines the filtered warnings. For each reported issue, the expert decides whether it is a true positive or a false positive. The expert may also provide additional context or mark the issue for further investigation. This step is particularly valuable for logical vulnerabilities, which often depend on the intended business rules of the contract and are difficult for fully automated tools to judge correctly [16], [22], [23].

3.5 Evaluation Metrics

To measure the benefit of human involvement, we use standard classification metrics. Let TP, FP, and FN denote the number of true positives, false positives, and false negatives, respectively. Precision, Recall, and F1-score are calculated as follows:

$$Precision = TP / (TP + FP) \quad (3)$$

$$Recall = TP / (TP + FN) \quad (4)$$

$$F_1 = 2 \cdot (Precision \cdot Recall) / (Precision + Recall) \quad (5)$$

We also measure the reduction in false positives achieved by the human review stage:

$$\Delta FP = FPauto - FPHITL \quad (6)$$

$$Reduction\ rate = \Delta FP / FPauto \quad (7)$$

These metrics allow us to quantify both the accuracy improvement and the practical cost of adding human oversight [24], [35].

4. Experimental Setup and Results

In this section, we describe the experimental setup used to evaluate the proposed Human-in-the-Loop framework and present the obtained results.

4.1 Dataset

We conducted our experiments on the SmartBugs Curated dataset, which contains 143 annotated Solidity smart contracts. These contracts are organized according to the DASP taxonomy and cover a range of vulnerability types, including reentrancy, access control, arithmetic issues, unchecked low-level calls, and others [17], [23]. Table 1 shows the distribution of contracts across the main vulnerability categories.

Table 1. Distribution of contracts in the SmartBugs Curated dataset

Category	Number of Contracts
Unchecked Low Level Calls	52
Reentrancy	31
Access Control	18
Arithmetic	15
Bad Randomness	8
Denial of Service	6
Time Manipulation	5
Front Running	4
Other	3
Short Addresses	1

Total	**143**

This dataset is particularly suitable for evaluating logical and access-control related vulnerabilities, which are known to be challenging for fully automated tools.

4.2 Experimental Design

We simulated the behavior of a static analysis tool based on performance patterns commonly reported in the literature [17], [21], [23]. Each reported warning was assigned a confidence score. Only warnings with a confidence score greater than or equal to 0.55 were forwarded to the human review stage. The human reviewer was assumed to correctly identify and reject a portion of the false positives, consistent with previous studies on human oversight in security analysis [24], [35].

We compared two settings:

- Automated only: All warnings produced by the tool were accepted without human intervention.
- HITL: Only high-confidence warnings were reviewed by a human expert, and false positives identified during review were removed.

4.3 Results

Table 2 summarizes the performance of the automated detector and the HITL approach.

Table 2. Performance comparison between automated detection and HITL

Metric	Automated Only	HITL	Improvement
True Positives (TP)	57	57	–
False Positives (FP)	23	11	–12 (52.2%)
True Negatives (TN)	63	63	–
Precision	0.7125	**0.8382**	+0.1257
Recall	1.0000	1.0000	–
F1-Score	0.8321	**0.9120**	+0.0799

As shown in Table 2, the introduction of human review led to a clear improvement in precision (from 0.7125 to 0.8382) and F1-score (from 0.8321 to 0.9120). The number of false positives was reduced by 52.2% (from 23 to 11), while recall remained unchanged at 1.0. This indicates that the human expert successfully filtered out a substantial portion of incorrect warnings without missing any true vulnerabilities that had been flagged by the automated tool.

4.4 Discussion of Results

The results support the main hypothesis of this work: combining automated analysis with targeted human review can significantly improve the reliability of vulnerability detection, especially for issues that involve business logic or access control [16], [22], [24]. The confidence-based filtering step played an important role in keeping the human workload manageable, as only a subset of warnings required manual inspection.

It is worth noting that these results are based on a controlled simulation that reflects typical tool behavior reported in prior empirical studies [17], [23]. While a full manual review of all contracts would provide even stronger evidence, the current findings already demonstrate the practical value of the proposed HITL approach.

5. Conclusion

Blockchain systems continue to face significant security challenges that span from the consensus layer to the application layer. In this paper, we examined these challenges with a particular focus on logical vulnerabilities in smart contracts and the potential of Human-in-the-Loop approaches.

At the consensus layer, different families of protocols offer distinct trade-offs between security, performance, and decentralization. Proof-of-Work provides strong guarantees under high computational assumptions but faces energy and scalability concerns. Proof-of-Stake reduces energy use while introducing new economic and historical attack risks. Hybrid designs attempt to combine advantages, yet they also create additional complexity and new attack surfaces [8], [9], [11].

At the smart-contract layer, logical vulnerabilities remain especially difficult to detect. These flaws often arise from mismatches between intended business rules and actual code behavior. While automated tools have improved the detection of well-known patterns, they still show clear limitations when dealing with context-dependent logical errors [15], [16], [17], [21].

To address this gap, we proposed a simple Human-in-the-Loop framework that combines automated analysis with targeted human review. Experiments on the SmartBugs Curated dataset showed that the proposed approach increased precision from 0.7125 to 0.8382 and F1-score from 0.8321 to 0.9120, while reducing false positives by 52.2% without any loss in recall. These results suggest that integrating human expertise can meaningfully improve the reliability of logical vulnerability detection [24], [35].

Several research directions remain open. Future work could focus on developing more advanced confidence estimation methods, creating larger and more diverse datasets specifically labeled for logical vulnerabilities, and designing interactive visualization tools that further reduce the cognitive load on human reviewers. Cross-layer analysis that connects consensus-level risks with contract-level vulnerabilities also deserves more attention.

In summary, securing blockchain systems requires attention to both technical mechanisms and human factors. By combining automated detection with structured human oversight, it becomes possible to build more reliable and trustworthy decentralized applications.

The source code and dataset used in this study are publicly available at:
https://github.com/alireza9999/blockchain-hitl-framework

References

[1] Liu, Q., Wu, M., Chen, J., Chen, T., Chen, X., Jiang, R., ... & Cheng, Y. (2025). Exploring the potential of chatgpt in detecting logical vulnerabilities in smart contracts. *Blockchain: Research and Applications*, 100294.

[2] Qi, M., Li, J., Zhu, T., Zhang, L., & Sun, Z. (2026). Detecting logic vulnerabilities across the contract and device layers of blockchain-enabled IoT with multi-agent heterogeneous graph attention. *arXiv preprint arXiv:2609.18344*.

[3] Meilinda, V., Ming, L. W., Muhtarom, M., Zanubiya, J., Kusuma, M. R., & Yaputra, R. (2026). Artificial intelligence and iot integration for intelligent decision-making systems. *Smart Human-centered Emerging Research in Machine Intelligence*, *2*(1), 1-13.

[4] Cao, R., Fan, S., Sui, Z., Fang, L., Yang, Z., Jiao, Y., & Liu, Z. (2026). DeFiScreener: Efficient DeFi Attack Pre-screening in Smart Contracts via Historical Case Matching. *arXiv preprint arXiv:2607.22184*.

[5] Wang, B., Tong, Y., Ji, S., Dong, H., Luo, X., & Zhang, P. (2026). A review of learning-based smart contract vulnerability detection: A perspective on code representation. *ACM Transactions on Software Engineering and Methodology*, *35*(6), 1-46.

[6] Ruaro, N., Gritti, F., McLaughlin, R., Grishchenko, I., Kruegel, C., & Vigna, G. (2024, March). Not your Type! Detecting Storage Collision Vulnerabilities in Ethereum Smart Contracts. In *NDSS*.

[7] Jain, A. K., Gupta, N., & Gupta, B. B. (2025). A survey on scalable consensus algorithms for blockchain technology. *Cyber Security and Applications*, *3*, 100065.

[8] Abellán Álvarez, I., Gramlich, V., & Sedlmeir, J. (2024, April). Unsealing the secrets of blockchain consensus: A systematic comparison of the formal security of proof-of-work and proof-of-stake. In *Proceedings of the 39th ACM/SIGAPP symposium on applied computing* (pp. 278-287).

[9] Al-awamy, A. A., Al-shaibany, N., Sikora, A., & Welte, D. (2025). Hybrid consensus mechanisms in blockchain: a comprehensive review. *International Journal of Intelligent Systems*, *2025*(1), 5821997.

[10] Nasir, N. M., Hassan, S., & Zaini, K. M. (2024). Securing permissioned blockchain-based systems: An analysis on the significance of consensus mechanisms. *IEEE Access*, *12*, 138211-138238.

[11] Rao, O. (2026). Human-in-the-Loop Blockchain Governance for Project Change Control: Balancing Smart-Contract Automation and Managerial Discretion. *The Regional Tribune*, *5*(5), 82-97.

[12] Assadzadeh, A. (2026). Blockchain Security: A Reproducible Machine Learning Framework for Smart Contract Vulnerability Detection.

[13] Tsabary, I., Manuskin, A., Bar-Zur, R., & Eyal, I. (2024, March). LedgerHedger: Gas Reservation for Smart Contract Security. In *International Conference on Financial Cryptography and Data Security* (pp. 248-270). Cham: Springer Nature Switzerland.

[14] Zhang, B., Liu, H., Tian, Q., Chen, S., Wang, Z., & Qi, Q. (2026, July). Towards Trustworthy Smart Contract Synthesis: A Multi-Agent Framework with Lean-Based Verification. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 39548-39582).

[15] Li, D., Wong, W. E., Wang, X., Pan, S., & Koh, L. S. (2024). Smart contract vulnerability detection based on static analysis and multi-objective search. *arXiv preprint arXiv:2410.00282*.

[16] Chahoki, A. Z., & Roveri, M. (2025). Static analysis for detecting transaction conflicts in ethereum smart contracts. *arXiv preprint arXiv:2507.04357*.

[17] Chaliasos, S., Charalambous, M. A., Zhou, L., Galanopoulou, R., Gervais, A., Mitropoulos, D., & Livshits, B. (2024, February). Smart contract and defi security tools: Do they meet the needs of practitioners?. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering* (pp. 1-13).

[18] Dong, J., Qin, Z., Fang, Z., Chen, X., Huang, Z., Guo, H., ... & Chen, S. (2023, September). Visual analytics for phishing scam identification in blockchain transactions with multiple model comparison. In *Proceedings of the 16th International Symposium on Visual Information Communication and Interaction* (pp. 1-9).

[19] Cao, Y., Xia, M., Shigyo, K., Cheng, F., Yu, Q., Yang, X., ... & Qu, H. (2023, September). NFTeller: Dual-centric visual analytics for assessing market performance of NFT collectibles. In *Proceedings of the 16th International Symposium on Visual Information Communication and Interaction* (pp. 1-8).

[20] Liao, Z., Nan, Y., Liang, H., Hao, S., Zhai, J., Wu, J., & Zheng, Z. (2024). Smartaxe: Detecting cross-chain vulnerabilities in bridge smart contracts via fine-grained static analysis. *Proceedings of the ACM on Software Engineering*, *1*(FSE), 249-270.

[21] Liu, D., Zhang, J., Wang, Y., Shen, H., Zhang, Z., & Ye, T. (2025). Blockchain smart contract security: Threats and mitigation strategies in a lifecycle perspective. *ACM Computing Surveys*, *58*(4), 1-34.

[22] Al. Ghanmi, H., Ahmadjee, S., Bahsoon, R., & Adeyemo, H. B. (2026). Exploring Human-Centric Dimensions in Blockchain Smart Contracts. *ACM Computing Surveys*, *58*(7), 1-43.

[23] Yadav, A. K., Singh, K., Amin, A. H., Almutairi, L., Alsenani, T. R., & Ahmadian, A. (2023). A comparative study on consensus mechanism with security threats and future scopes: Blockchain. *Computer Communications*, *201*, 102-115.

[24] Thornton, J. A., & Whitfield, E. (2026). AI-Assisted Human-in-the-Loop Decision-Making for Smart Contract Risk Management in Blockchain-Based Business Systems. *International Journal of Sustainable Applied Science and Engineering*, *3*(1), 95-114.

[25] Al Ghanmi, H. (2024). *Explainability requirement in blockchain smart contracts: a human-centred approach* (Doctoral dissertation, University of Birmingham).

[26] Guru, A., Mohanta, B. K., Mohapatra, H., Al-Turjman, F., Altrjman, C., & Yadav, A. (2023). A survey on consensus protocols and attacks on blockchain technology. *Applied sciences*, *13*(4), 2604.

[27] Yakubu, M. M., Hassan, M. F. B., Danyaro, K. U., Junejo, A. Z., Siraj, M., Yahaya, S., ... & Abdulsalam, K. (2024). A Systematic Literature Review on Blockchain Consensus Mechanisms' Security: Applications and Open Challenges. *Comput. Syst. Sci. Eng.*, *48*(6), 1437-1481.

[28] Sipos, M., & Szénási, S. (2025, November). Blockchain Transaction Graph Analysis Using Image Processing Techniques. In *2025 IEEE 25th International Symposium on Computational Intelligence and Informatics (CINTI)* (pp. 000529-000534). IEEE.

[29] Tehrani, D. N., & Tovanich, N. (2026, October). MoneroVis: Visual Analytics for Exploration and Traceability in a Privacy-Preserving Blockchain. In *8th Conference on Blockchain Research & Applications for Innovative Networks and Service (BRAINS)*.

[30] Vinceslas, L., Dogan, S., Sundareshwar, S., & Kondoz, A. M. (2023). Abstracting data in distributed ledger systems for higher level analytics and visualizations. *Future Internet*, *15*(1), 33.

[31] Kushwaha, S. S., Joshi, S., & Gupta, A. K. (2023). An efficient approach to secure smart contract of Ethereum blockchain using hybrid security analysis approach. *Journal of Discrete Mathematical Sciences and Cryptography*, *26*(5), 1499-1517.

[32] Assadzadeh, A. (2026). Rug Pulls in the IoT and DePIN Ecosystem: Patterns, Red Flags, and Risk Mitigation. Preprints. https://doi.org/10.20944/preprints202611.0259.v1

[33] Cheung, K. T., & Ramamoorti, S. (2026). Accountability in smart contracts and agentic AI: How to address the responsibility gap and retain meaningful human control?. *AI and Ethics*, *6*(6), 585.

[34] Jeyakumar, S., Hóu, Z., Charles, E. Y. A., Palaniswami, M., & Muthukkumarasamy, V. (2023). Visualizing blockchain transaction behavioural pattern: A graph-based approach.

[35] Fattahdizaji, A., Pishdar, M., & Shukur, Z. (2026). SmartGraphical: A Human-in-the-Loop Framework for Detecting Smart Contract Logical Vulnerabilities via Pattern-Driven Static Analysis and Visual Abstraction. *arXiv preprint arXiv:2603.08580*.

[36] Pishdar, M., & Manzoor, J. (2026). Why No Consensus on Consensus? A Deep Dive Into Blockchain Consensus Protocols. *IET Blockchain*, *6*(1), e70050.

[37] Li, N., Qi, M., Xu, Z., Zhu, X., Zhou, W., Wen, S., & Xiang, Y. (2025). Blockchain cross-chain bridge security: Challenges, solutions, and future outlook. *Distributed Ledger Technologies: Research and Practice*, *4*(1), 1-34.