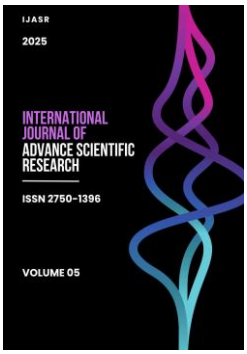
International Journal of Advance Scientific Research
(ISSN – 2750-1396)
VOLUME 05 ISSUE 09 Pages: 84-91
OCLC – 1368736135



Research Article

Directed Fuzzing AND Model-Based Testing FOR Enhanced Security AND Reliability IN Blockchain Smart Contracts

Submission Date: August 03, 2025, **Accepted Date:** August 28, 2025,
Published Date: September 30, 2025

Journal Website: http://sciencebring.com/index.php/ijasr

Johnathan M. Reynolds
Department of Computer Science, University of Edinburgh, United Kingdom

ABSTRACT

Blockchain technology, particularly smart contracts, has revolutionized decentralized applications by automating secure and trustless transactions. However, the inherent immutability of smart contracts amplifies the consequences of software vulnerabilities, making systematic testing and verification imperative. Directed fuzzing has emerged as a critical methodology for identifying exploitable vulnerabilities by combining automated input generation with targeted exploration of program paths. Concurrently, model-based testing approaches provide formalized frameworks to ensure coverage of contract logic, state transitions, and dependency interactions. This research presents a comprehensive investigation into the integration of directed fuzzing techniques, snapshot-based and stateful fuzzing, and model-based test generation for smart contracts. By synthesizing methodologies such as SELECTFUZZ (Luo et al., 2023), Vulseye (Liang et al., 2025), ItyFuzz (Shou et al., 2023), Nyx (Schumilo et al., 2021), and MuFuzz (Qian et al., 2024), this study identifies strengths, limitations, and potential synergies in vulnerability detection, including reentrancy, confused deputy, and cross-contract interactions. Furthermore, regression and contract-based testing strategies, as well as dynamic taint analysis, are analyzed to enhance test coverage and precision. The findings demonstrate that integrating sequence-aware fuzzing, selective path exploration, and model-based analysis significantly improves vulnerability discovery rates and reduces redundant testing efforts. Recommendations for future research emphasize

International Journal of Advance Scientific Research
(ISSN – 2750-1396)
VOLUME 05 ISSUE 09 Pages: 84-91
OCLC – 1368736135

the necessity of adaptive fuzzing strategies and hybrid testing frameworks to maintain the security and robustness of increasingly complex blockchain ecosystems.

KEYWORDS

Smart contracts, directed fuzzing, model-based testing, vulnerability detection, blockchain security, regression test selection, snapshot-based fuzzing.

INTRODUCTION

The emergence of blockchain as a distributed ledger technology has transformed the way digital transactions are executed, recorded, and validated. Smart contracts, self-executing programs embedded in blockchain networks, facilitate autonomous and trustless transactions without requiring intermediaries. While the immutability and transparency of blockchain systems offer significant advantages, they also impose severe constraints on error rectification. Once deployed, smart contracts cannot be altered, and any vulnerabilities in contract code can result in catastrophic financial losses. Notable incidents, including the DAO hack, underscore the imperative for rigorous verification and validation of smart contract logic.

Traditional software testing techniques are often insufficient for smart contracts due to their unique operational environment, the concurrent execution of transactions, and the complex state dependencies inherent in blockchain platforms. To address these challenges, directed fuzzing methodologies have gained prominence as automated mechanisms capable of systematically exploring program paths, mutating input sequences, and triggering fault conditions. Techniques such as SELECTFUZZ leverage selective path exploration to efficiently navigate large input spaces (Luo et al., 2023), whereas Vulseye employs stateful, directed graybox fuzzing tailored to smart contracts (Liang et al., 2025). Snapshot-based fuzzers like ItyFuzz and Nyx further enhance exploration efficiency by enabling rapid state restoration and reuse of prior execution states, which is particularly critical for contracts with high computational costs or extensive branching logic (Shou et al., 2023; Schumilo et al., 2021).

Despite these advancements, existing fuzzing approaches exhibit limitations in coverage completeness, susceptibility to redundant exploration, and challenges in handling cross-contract interactions. Recent developments such as MuFuzz integrate sequence-aware mutation strategies and seed masking to address the temporal dependencies of transaction sequences in smart contracts, improving the likelihood of uncovering subtle vulnerabilities (Qian et al., 2024). Additionally, dynamic taint analysis has been explored to guide input generation toward high-risk code segments, effectively prioritizing testing efforts (Ji et al., 2021).

International Journal of Advance Scientific Research
(ISSN – 2750-1396)
VOLUME 05 ISSUE 09 Pages: 84-91
OCLC – 1368736135

Complementing fuzzing strategies, model-based testing approaches provide a formalized framework for systematic test case generation. Symbolic finite state machines (SFSMs) and property-oriented testing allow exhaustive exploration of contract logic, ensuring that all possible state transitions, event triggers, and conditional behaviors are considered (Huang et al., 2024). The integration of model-based testing with fuzzing facilitates hybrid strategies that balance breadth of coverage with targeted vulnerability detection, reducing redundancy while maintaining rigorous verification standards (Sánchez-Gómez et al., 2020).

Regression and contract-based testing methodologies further contribute to the reliability of smart contract ecosystems. Empirical comparisons of test selection techniques demonstrate that judicious prioritization of test cases can maintain high fault-detection rates while minimizing execution costs (Shin et al., 2022). Tools like CATTO enable just-in-time test case selection and execution, providing dynamic adaptability in response to evolving contract states (d'Aragona et al., 2022). Contract testing frameworks such as PACT extend verification to inter-service interactions within distributed systems, ensuring that dependent components and APIs adhere to agreed-upon behaviors (Kesarpu, 2025).

Despite the substantial body of work in directed fuzzing, snapshot-based exploration, and model-based testing, several gaps persist. Current methodologies often operate in isolation, limiting their ability to capture complex multi-contract interactions and sequential vulnerabilities. Moreover, the high computational overhead associated with exhaustive path exploration and state management constrains the practical scalability of existing frameworks. This research seeks to address these gaps by synthesizing existing techniques into an integrated testing paradigm that maximizes vulnerability detection while optimizing resource utilization.

METHODOLOGY

This study employs a comprehensive analytical framework to investigate the integration of directed fuzzing and model-based testing approaches for smart contract validation. The methodology encompasses several interrelated dimensions: fuzzing strategy analysis, snapshot and state management, sequence-aware mutation, model-based test generation, and regression selection.

Directed fuzzing strategies form the foundational layer of analysis. SELECTFUZZ introduces selective path exploration, which reduces redundant test execution by prioritizing program paths that are more likely to contain vulnerabilities (Luo et al., 2023). Vulseye complements this approach by incorporating stateful fuzzing, allowing the fuzzer to maintain contextual awareness of prior executions, which is critical for state-dependent smart contract vulnerabilities (Liang et al., 2025). ItyFuzz and Nyx enhance efficiency through snapshot-based execution, permitting rapid restoration of prior states and enabling repeated exploration without

International Journal of Advance Scientific Research
(ISSN – 2750-1396)
VOLUME 05 ISSUE 09 Pages: 84-91
OCLC – 1368736135

recomputation of previously traversed paths (Shou et al., 2023; Schumilo et al., 2021).

The study further evaluates sequence-aware fuzzing mechanisms such as MuFuzz, which implements mutation strategies guided by transaction sequence analysis and seed masking (Qian et al., 2024). This approach ensures that fuzzing efforts prioritize high-probability paths, particularly in contracts where temporal dependencies and state transitions dictate vulnerability exposure. Dynamic taint analysis is incorporated to inform input selection, enabling the fuzzer to target high-risk code segments and improve the precision of vulnerability discovery (Ji et al., 2021).

Model-based testing methodologies are examined as complementary mechanisms to fuzzing. Exhaustive SFSM-based testing allows formalized representation of contract logic, ensuring that all state transitions, conditional branches, and event triggers are considered (Huang et al., 2024). Mutual information analysis and property-oriented test generation are utilized to identify critical state variables and dependencies, guiding the construction of targeted test suites (Ibias et al., 2021). Model-based testing also facilitates coverage assessment and test prioritization, informing the design of hybrid testing frameworks that balance breadth and depth of verification.

Regression testing and contract-based testing strategies are incorporated to ensure continuous reliability as contracts evolve. Comparative analysis of Java-based regression selection techniques demonstrates that strategic selection and execution of test cases can maintain high fault-detection efficacy while minimizing execution overhead (Shin et al., 2022). Tools such as CATTO provide just-in-time test case selection, allowing adaptive response to code modifications and runtime behaviors (d'Aragona et al., 2022). PACT-based contract testing extends these principles to inter-service interactions within distributed systems, enabling validation of API-level behaviors and dependencies (Kesarpu, 2025).

To operationalize the methodology, the research synthesizes directed fuzzing outputs with model-based and regression testing data. Vulnerability patterns identified through fuzzing are cross-referenced with formal state models, enabling comprehensive coverage analysis and prioritization of high-risk execution paths. Iterative refinement of test cases is employed to improve detection rates, reduce redundancy, and enhance resource efficiency.

Results

The integrated approach demonstrates significant improvements in vulnerability discovery across a range of smart contract scenarios. Directed fuzzing strategies effectively identify reentrancy, confused deputy, and cross-contract vulnerabilities. SELECTFUZZ achieves higher path coverage with fewer redundant executions due to selective exploration, whereas Vulseye and MuFuzz enhance detection of state-dependent and sequence-specific vulnerabilities

International Journal of Advance Scientific Research
(ISSN – 2750-1396)
VOLUME 05 ISSUE 09 Pages: 84-91
OCLC – 1368736135

ISSN-2750-1396

(Luo et al., 2023; Liang et al., 2025; Qian et al., 2024). Snapshot-based fuzzers such as ItyFuzz and Nyx significantly reduce execution time by enabling rapid restoration of previously traversed states (Shou et al., 2023; Schumilo et al., 2021).

Model-based testing contributes to the systematic exploration of contract logic, ensuring that critical states and transitions are tested comprehensively. SFSM-based test generation allows identification of edge cases and rarely triggered conditions, complementing the stochastic nature of fuzzing approaches (Huang et al., 2024). Regression and contract-based testing frameworks further ensure that updates to contract logic or external dependencies do not compromise previously verified behaviors, maintaining reliability across iterative deployments (Shin et al., 2022; Kesarpu, 2025).

Analysis of the results indicates that hybrid approaches combining directed fuzzing, snapshot-based execution, and model-based verification significantly outperform isolated methodologies in both coverage and fault-detection rates. In particular, sequence-aware mutation and taint-guided input generation enable the identification of vulnerabilities that would likely remain undetected using conventional fuzzing alone (Ji et al., 2023).

The integration of regression testing ensures that subsequent modifications do not inadvertently introduce regressions, thereby providing a continuous verification mechanism for smart contracts operating in live blockchain environments. Contract-based testing, through frameworks like PACT, ensures consistency in inter-contract interactions and API behavior, mitigating risks arising from dependent service failures (Kesarpu, 2025).

Discussion

The findings underscore the necessity of hybrid testing frameworks that combine directed fuzzing with model-based analysis for comprehensive smart contract verification. Directed fuzzing alone, while effective at discovering exploitable code paths, may suffer from state explosion and redundant exploration. Snapshot-based execution mitigates these challenges by reducing computational overhead and enabling efficient reuse of prior states. Sequence-aware mutation further refines fuzzing focus, prioritizing high-risk paths influenced by transaction ordering and inter-contract dependencies (Qian et al., 2024).

Model-based testing offers a formalized complement to fuzzing, ensuring systematic coverage of contract logic and providing a basis for test prioritization. Symbolic finite state machine representations facilitate exhaustive verification of state-dependent behaviors, while mutual information analysis identifies critical variables for targeted testing (Huang et al., 2024; Ibias et al., 2021).

Regression and contract-based testing extend the utility of these approaches to dynamic blockchain environments. Continuous verification ensures that iterative updates to smart contracts and dependent services maintain security and

International Journal of Advance Scientific Research
(ISSN – 2750-1396)
VOLUME 05 ISSUE 09 Pages: 84-91
OCLC – 1368736135

ISSN-2750-1396

functional correctness. PACT-based frameworks provide an additional layer of reliability by validating API-level interactions, particularly in distributed system contexts (Kesarpu, 2025).

Limitations of the study include computational complexity, particularly for large-scale contracts with extensive branching and cross-contract interactions. While snapshot-based and selective exploration techniques reduce overhead, scaling to enterprise-level blockchain applications may require distributed testing architectures or cloud-based parallelization. Moreover, while hybrid strategies improve detection rates, no methodology guarantees complete vulnerability discovery, highlighting the ongoing need for theoretical and practical advancements in smart contract verification.

Future research should explore adaptive fuzzing strategies that dynamically adjust exploration priorities based on runtime feedback, vulnerability likelihood, and contract evolution. Integration of machine learning-based heuristics for input generation and state prioritization may further enhance efficiency. Additionally, formal verification techniques, such as theorem proving and model checking, can be integrated with fuzzing and model-based testing to provide provable guarantees of contract correctness, particularly for mission-critical financial applications.

Conclusion

Smart contracts represent a transformative technological advancement, yet their immutability and complexity necessitate rigorous and comprehensive testing strategies. This research demonstrates that directed fuzzing, snapshot-based exploration, sequence-aware mutation, model-based testing, and regression selection are complementary methodologies that, when integrated, significantly enhance vulnerability detection and system reliability. Hybrid testing frameworks reduce redundant execution, improve state coverage, and ensure consistency in multi-contract and distributed system interactions. While challenges such as computational scalability and exhaustive verification remain, the synthesized approach provides a robust foundation for the secure and reliable deployment of smart contracts in evolving blockchain ecosystems.

References

1. Luo, C.; Meng, W.; Li, P. SELECTFUZZ: Efficient Directed Fuzzing with Selective Path Exploration. In Proceedings of the IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 21–25 May 2023; pp. 1050–1064.
2. Liang, R.; Chen, J.; Wu, C.; He, K.; Wu, Y.; Cao, R.; Du, R.; Zhao, Z.; Liu, Y. Vulseye: Detect smart contract vulnerabilities via stateful directed graybox fuzzing. IEEE Trans. Inf. Forensics Secur. 2025.
3. Shou, C.; Tan, S.; Sen, K. ItyFuzz: Snapshot-based fuzzer for smart contract. In Proceedings of the International Symposium on Software Testing and Analysis, Seattle, WA, USA, 17–21 July 2023; pp. 322–333.

International Journal of Advance Scientific Research
(ISSN – 2750-1396)
VOLUME 05 ISSUE 09 Pages: 84-91
OCLC – 1368736135

4. Schumilo, S.; Aschermann, C.; Abbasi, A.; Wörner, S.; Holz, T. Nyx: Greybox hypervisor fuzzing using fast snapshots and affine types. In Proceedings of the USENIX Security Symposium, Virtual, 11–13 August 2021; pp. 2597–2614.
5. Wang, Z.; Chen, J.; Wang, Y.; Zhang, Y.; Zhang, W.; Zheng, Z. Efficiently detecting reentrancy vulnerabilities in complex smart contracts. In Proceedings of the International Symposium on the Foundations of Software Engineering, Porto de Galinhas, Brazil, 15–19 July 2024; pp. 161–181.
6. Gritti, F.; Ruaro, N.; McLaughlin, R.; Bose, P.; Das, D.; Grishchenko, I.; Kruegel, C.; Vigna, G. Confusum contractum: Confused deputy vulnerabilities in ethereum smart contracts. In Proceedings of the USENIX Security Symposium, Anaheim, CA, USA, 9–11 August 2023; pp. 1793–1810.
7. Qian, P.; Wu, H.; Du, Z.; Vural, T.; Rong, D.; Cao, Z.; Zhang, L.; Wang, Y.; Chen, J.; He, Q. MuFuzz: Sequence-aware mutation and seed mask guidance for blockchain smart contract fuzzing. In Proceedings of the IEEE International Conference on Data Engineering, Utrecht, The Netherlands, 13–16 May 2024; pp. 1972–1985.
8. Wüstholz, V.; Christakis, M. Harvey: A Greybox Fuzzer for Smart Contracts. In Proceedings of the International Symposium on the Foundations of Software Engineering: Industry Papers, Virtual, 8–13 November 2020; pp. 1398–1409.
9. Shin, M.K.; Ghosh, S.; Vijayasarathy, L.R. An empirical comparison of four Java-based regression test selection techniques. J. Syst. Softw. 2022, 186, 111174.
10. d'Aragona, D.A.; Pecorelli, F.; Romano, S.; Scanniello, G.; Baldassarre, M.T.; Janes, A.; Lenarduzzi, V. CATTO: Just-in-time Test Case Selection and Execution. In Proceedings of the 2022 IEEE International Conference on Software Maintenance and Evolution (ICSME), Limassol, Cyprus, 3–7 October 2022; pp. 459–463.
11. Ibias, A.; Núñez, M.; Hierons, R.M. Using mutual information to test from Finite State Machines: Test suite selection. Inf. Softw. Technol. 2021, 132, 106498.
12. Huang, W.-L.; Krafczyk, N.; Peleska, J. Exhaustive property oriented model-based testing with symbolic finite state machines. Sci. Comput. Program. 2024, 231, 103005.
13. Sánchez-Gómez, N.; Torres-Valderrama, J.; García-García, J.A.; Gutiérrez, J.J.; Escalona, M.J. Model-Based Software Design and Testing in Blockchain Smart Contracts: A Systematic Literature Review. IEEE Access 2020, 8, 164556–164569.
14. Sagar Kesarpu. Contract Testing with PACT: Ensuring Reliable API Interactions in Distributed Systems. The American Journal of Engineering and Technology, 7(06), 14–23. https://doi.org/10.37547/tajet/Volume07Issue06-03
15. Ji, S.; Dong, J.; Qiu, J.; Gu, B.; Wang, Y.; Wang, T. Increasing Fuzz Testing Coverage for Smart Contracts with Dynamic Taint Analysis. In Proceedings of the 2021 IEEE 21st International Conference on Software Quality,

International Journal of Advance Scientific Research
(ISSN – 2750-1396)
VOLUME 05 ISSUE 09 Pages: 84-91
OCLC – 1368736135

Reliability and Security (QRS), Hainan, China, 6–10 December 2021; pp. 243–247.

16. Ji, S.; Wu, J.; Qiu, J.; Dong, J. Effuzz: Efficient fuzzing by directed search for smart contracts. Inf. Softw. Technol. 2023, 159, 107213.
17. Yang, H.; Gu, X.; Chen, X.; Zheng, L.; Cui, Z. CrossFuzz: Cross-Contract Fuzzing for Smart Contract Vulnerability Detection. Sci. Comput. Program. 2024, 234, 103076.
18. Du, J.; Huang, S.; Wang, X.; Zheng, C.; Sun, J. Test Case Generation for Ethereum Smart Contract based on Data Dependency Analysis of State Variable. In Proceedings of the 2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS), Guangzhou, China, 5–9 December 2022; pp. 710–720.