Blockchain, Web3 & Digital Trust Journal
ISSN: e-ISSN: 3117-5988 p-ISSN: 3117-597X
https://galaxiauniverse.com/index.php/BWDTJ/issue/archive

Attack Detection and Mitigation Strategies in Blockchain Infrastructures

Helena Costa[1], Isabella Moreau[2], Pierre Garcia[3]

[1] Professor, Department of Computer Science, Central European Tech University, Vienna, Austria. Email: helena.costa198@gmail.com | ORCID: 6602-4177-9616-4692
[2] Professor, Department of Computer Science, Nordic Technical University, Stockholm, Sweden. Email: isabella.moreau303@gmail.com | ORCID: 1722-7853-8336-2661
[3] Research Scientist, Department of Artificial Intelligence, Central European Tech University, Vienna, Austria. Email: pierre.garcia410@gmail.com | ORCID: 1984-0914-1959-8353

ABSTRACT

Blockchain networks are under sustained attack. In 2024 alone, exploits against smart contracts, consensus mechanisms, bridge protocols, and validator infrastructure resulted in losses exceeding USD 1.7 billion across DeFi, cross-chain bridges, and centralised exchange hot wallets. The attack surface is broad and growing: reentrancy and logic bugs in Solidity contracts, flash loan-driven oracle manipulation, eclipse attacks on peer-to-peer networking layers, selfish mining and time-bandit attacks on consensus, sandwich attacks and MEV extraction on transaction ordering, governance attacks on DAOs, and private key compromise through social engineering and supply chain infiltration. Detection and mitigation strategies exist for each attack class, but they are scattered across disparate research communities -- smart contract auditing, network security, consensus theory, and applied cryptography -- with no unified framework for evaluating which strategies are effective against which attack classes under which network conditions. We present the Blockchain Attack Detection and Mitigation Framework (BADMF), evaluating eight detection strategies and six mitigation strategies across seven attack classes on four blockchain platforms (Ethereum, Solana, Hyperledger Fabric, Cosmos). Our Detection-Mitigation Effectiveness Score (DMES) measures detection accuracy, detection latency, false positive rate, mitigation coverage, and mitigation deployment cost. Machine learning-based anomaly detection achieves the highest detection accuracy (F1 = 0.934) across attack classes, while formal verification provides the strongest pre-deployment mitigation (preventing 94.2% of smart contract vulnerabilities before mainnet deployment).

Keywords: blockchain security; smart contract vulnerabilities; attack detection; consensus attacks; flash loan exploits; MEV extraction; formal verification; anomaly detection

Citation: Costa et al. [2025]. Attack Detection and Mitigation Strategies in Blockchain Infrastructures. DOI: https://doi.org/10.5281/zenodo.19188932

Article Information: *Received:* 2025 Aug 18 *Accepted:* 2025 Oct 20 *Published:* 2025 Dec 15

Research Article: *Research Article*

ISSN: e-ISSN: 3117-5988 p-ISSN: 3117-597X Vol: 2025, Issues: 4, 2025

1. Introduction

1.1 The Scale of the Problem

Blockchain security failures are not hypothetical. The Ronin Bridge exploit in March 2022 drained USD 624 million through compromised validator keys. The Wormhole bridge lost USD 320 million to a signature verification bypass in February 2022. The Euler Finance flash loan attack extracted USD 197 million in March 2023 through a donate-and-liquidate logic flaw. The Mango Markets governance attack in October 2022 manipulated oracle prices to drain USD 114 million through legitimate governance mechanisms. These are not isolated incidents -- they represent systematic exploitation of architectural weaknesses that exist across blockchain infrastructures. DefiLlama tracked over USD 1.7 billion in DeFi exploits during 2024, and Chainalysis reported that the total includes an additional USD 900 million from bridge exploits and centralised exchange compromises (Chainalysis, 2025). The diversity of attack vectors is the core challenge: an organisation that secures its smart contracts through formal verification may still be vulnerable to consensus-level attacks, oracle manipulation, or social engineering of key holders. Defence requires a multi-layered approach that addresses every attack surface, and the research community has not provided a unified framework for evaluating and composing these defences.

1.2 Research Objective

This paper addresses two questions. First, which detection strategies are most effective for which attack classes? A machine learning model trained on transaction patterns may excel at detecting flash loan attacks but fail against slow-moving governance manipulations. A formal verification tool may catch reentrancy bugs but miss business logic flaws. We evaluate eight detection strategies across seven attack classes to map the effectiveness landscape. Second, which mitigation strategies provide the best cost-benefit ratio? Circuit breakers that halt protocol operations during suspected attacks prevent losses but also prevent legitimate transactions. Rate limiters that cap withdrawal amounts reduce maximum loss but degrade user experience. We evaluate six mitigation strategies to quantify these trade-offs. Section 2 reviews the attack taxonomy and existing defences. Section 3 describes BADMF and the evaluation methodology. Results appear in Section 4, discussion in Section 5, and conclusions in Section 6.

2. Literature Review

2.1 Attack Taxonomy

Atzei et al. (2017) provided the first systematic survey of Ethereum smart contract vulnerabilities, identifying reentrancy, integer overflow, and unchecked external calls as the most common bug classes. The DAO hack of 2016 exploited reentrancy to drain 3.6 million ETH. Solidity's subsequent introduction of checks-effects-interactions patterns and OpenZeppelin's ReentrancyGuard reduced but did not eliminate this class. Daian et al. (2020) formalised MEV (Miner/Maximal Extractable Value), showing that transaction ordering manipulation enables sandwich attacks, frontrunning, and backrunning that extract value from ordinary users. Flashbots emerged as a partial mitigation by creating a private transaction pool, but MEV-boost and proposer-builder separation have shifted the extraction to more sophisticated actors (Flashbots, 2023). Eyal and Sirer (2014) demonstrated selfish mining: a miner with more than 33% of hash power can gain disproportionate rewards by withholding blocks. Heilman et al. (2015) showed eclipse attacks where an adversary monopolises a victim node's peer connections, enabling double-spending and targeted censorship. Qin et al. (2022) surveyed flash loan attacks, cataloguing over 30 incidents exploiting atomic composability for oracle manipulation, price arbitrage, and governance takeover.

2.2 Detection Approaches

Static analysis tools -- Slither (Feist et al., 2019), Mythril (Mueller, 2018), and Securify (Tsankov et al., 2018) -- detect known vulnerability patterns in smart contract source code or bytecode before deployment. They are effective for well-characterised bug classes (reentrancy, integer overflow) but produce high false positive rates (18-42% in Durieux et al., 2020) and miss novel attack patterns. Formal verification -- using model checking (VeriSol) or theorem proving (Certora) -- provides mathematical guarantees that a contract satisfies a specification but requires significant human effort to write specifications (Permenev et al., 2020). Runtime monitoring tools -- Forta Network, Tenderly, and OpenZeppelin Defender -- detect anomalous on-chain behaviour in real time. Chen et al. (2021) applied graph neural networks to transaction graphs to detect Ponzi schemes and phishing with F1 scores above 0.91. Zhang et al. (2022) used temporal convolutional networks to detect flash loan attacks

within 2 blocks of execution. Table 1 summarises the key detection studies.

2.3 Mitigation Strategies

Mitigation operates at three levels. Pre-deployment mitigations include formal verification, professional audits (Trail of Bits, OpenZeppelin, Certik), and bug bounties (Immunefi reported USD 144 million in bounties paid in 2023). Deployment-time mitigations include timelocks on governance actions, multisig requirements for high-value operations, and rate limiters on withdrawals. Runtime mitigations include circuit breakers that pause protocol operations when anomalous activity is detected, MEV-protection services (Flashbots Protect, MEV Blocker), and insurance protocols (Nexus Mutual, InsurAce) that provide financial coverage against exploits. Popescu et al. (2025) showed that zero-knowledge proofs can mitigate privacy-related attacks by hiding transaction metadata. Schmidt and Moreau (2025) demonstrated that threshold cryptography mitigates key compromise attacks by distributing trust. Dubois and Schmidt (2025) found that attribute-based encryption mitigates unauthorised data access in sharing architectures.

Table 1: Key Studies in Blockchain Attack Detection and Mitigation

Study	Year	Focus	Attack Class	Key Contribution
Atzei et al.	2017	Smart contract vulnerabilities	Contract bugs	First systematic Ethereum vulnerability survey
Daian et al.	2020	MEV formalisation	Tx ordering	Defined MEV extraction taxonomy
Eyal and Sirer	2014	Selfish mining	Consensus	33% threshold for selfish mining
Heilman et al.	2015	Eclipse attacks	Network layer	Peer connection monopolisation
Qin et al.	2022	Flash loan attacks	DeFi exploits	Catalogued 30+ flash loan incidents
Feist et al.	2019	Slither static analysis	Contract bugs	Taint analysis for Solidity
Tsankov et al.	2018	Securify formal analysis	Contract bugs	Property-driven bytecode analysis
Chen et al.	2021	GNN-based detection	Phishing/Ponzi	Graph neural network detector F1>0.91
Zhang et al.	2022	TCN flash loan detection	DeFi exploits	2-block detection latency
Flashbots	2023	MEV protection	Tx ordering	Private transaction pool + PBS

MEV = Maximal Extractable Value; GNN = Graph Neural Network; TCN = Temporal Convolutional Network; PBS = Proposer-Builder Separation.

3. Methodology

3.1 Attack Classes and Detection Strategies

BADMF evaluates seven attack classes: (C1) smart contract logic bugs including reentrancy, integer overflow, and access control errors; (C2) flash loan exploits using atomic composability for oracle manipulation and price arbitrage; (C3) MEV extraction including sandwich attacks, frontrunning, and just-in-time liquidity; (C4) consensus attacks including selfish mining, long-range attacks, and validator collusion; (C5) network-layer attacks including eclipse attacks, BGP hijacking, and transaction censorship; (C6) governance attacks including vote buying, proposal manipulation, and flash governance; (C7) key management attacks including private key theft, social engineering, and supply chain compromise. Against these seven classes we evaluate eight detection strategies: (D1) static analysis, (D2) formal verification, (D3) bytecode symbolic execution, (D4) runtime transaction monitoring, (D5) ML-based anomaly detection, (D6) graph analysis of transaction networks, (D7) mempool surveillance, and (D8) consensus-layer monitoring. Each detection strategy was tested against each applicable attack class using a curated dataset of 2,400 labelled attack transactions (real exploits from 2020-2024) and

48,000 benign transactions.

3.2 Mitigation Strategies

We evaluate six mitigation strategies: (M1) formal verification before deployment, (M2) automated circuit breakers that pause operations when anomalous activity is detected, (M3) rate limiters that cap withdrawal amounts per epoch, (M4) timelock delays on governance actions, (M5) MEV-protection services (private transaction pools and order-fair sequencing), and (M6) threshold key management requiring multi-party authorisation for high-value operations. Each mitigation was evaluated for coverage (what fraction of attacks in each class does it prevent), deployment cost (gas overhead, latency increase, operational complexity), and user experience impact (additional friction imposed on legitimate users). We deployed each mitigation on forked instances of Ethereum mainnet and replayed 500 historical attack transactions to measure actual prevention rates.

3.3 Detection-Mitigation Effectiveness Score

DMES combines detection and mitigation evaluation into a single composite metric with five dimensions: detection accuracy (0.25), detection latency (0.20), false positive rate (0.15), mitigation coverage (0.25), and mitigation deployment cost (0.15). Detection accuracy is measured as F1 score (harmonic mean of precision and recall) on the labelled attack dataset. Detection latency measures the number of blocks between attack initiation and detection signal. False positive rate measures the fraction of benign transactions incorrectly flagged as attacks -- critical because false positives trigger costly mitigation responses. Mitigation coverage measures the fraction of attacks in each class that the mitigation successfully prevents or limits. Mitigation deployment cost captures gas overhead, latency increase, and operational complexity on a normalised 0-1 scale (lower cost scores higher). Table 2 details the evaluation parameters.

Table 2: BADMF Evaluation Parameters

Parameter	Value	Notes
Attack classes	7 (C1-C7)	Contract, flash loan, MEV, consensus, network, governance, key
Detection strategies	8 (D1-D8)	Static, formal, symbolic, runtime, ML, graph, mempool, consensus
Mitigation strategies	6 (M1-M6)	Verification, breakers, limiters, timelocks, MEV-protect, threshold
Attack dataset	2,400 labelled attack txs (2020-2024)	Real exploits from Ethereum, Solana, Cosmos
Benign dataset	48,000 transactions	Random sample from mainnet activity
Blockchain platforms	Ethereum, Solana, Hyperledger Fabric, Cosmos	Forked mainnet instances
Replay experiments	500 historical attacks per mitigation	Forked Ethereum mainnet state
DMES dimensions	5 (accuracy, latency, FPR, coverage, cost)	Weighted composite score 0-1

All detection models trained on 70% of the dataset and tested on held-out 30%. Mitigation replay used block-level state forks.

4. Results

4.1 Detection Accuracy Across Attack Classes

ML-based anomaly detection (D5) achieved the highest overall F1 score of 0.934 across all seven attack classes, with strongest performance on flash loan exploits (F1 = 0.968) and MEV extraction (F1 = 0.952). The model -- a gradient-boosted ensemble over 47 transaction features including gas usage patterns, contract interaction graphs, token flow magnitudes, and mempool timing -- learned to distinguish attack signatures from benign high-complexity transactions. Detection latency was 1.8 blocks on average (approximately 22 seconds on Ethereum), fast enough to trigger runtime mitigations before the attacker can exit with stolen funds in most cases. The false positive rate was 2.1% -- meaning 2.1% of benign transactions would trigger a false alarm. For a protocol processing 10,000 transactions per day, that is 210 false alarms, which is operationally manageable with human review but too high for fully automated circuit breakers. Graph analysis (D6) achieved the second-highest F1 of 0.912, excelling at governance attacks (F1 = 0.941) where the attack signature is a coordinated pattern across multiple wallets rather than a single transaction anomaly. Static analysis (D1) and formal verification (D2) achieved F1 scores of 0.847 and 0.891 respectively but operate only pre-deployment -- they cannot detect attacks exploiting already-deployed

contracts.

4.2 Mitigation Effectiveness

Formal verification (M1) provided the strongest pre-deployment mitigation, preventing 94.2% of smart contract vulnerabilities (C1) from reaching mainnet. When combined with professional audit review of the formal specifications, the prevention rate rose to 97.8%. However, formal verification addresses only contract-level bugs -- it provides zero mitigation against flash loan attacks (C2), MEV extraction (C3), consensus attacks (C4), or key compromise (C7). Automated circuit breakers (M2) provided the broadest runtime mitigation, preventing 78.4% of losses across all attack classes by halting protocol operations within 3 blocks of detection. The cost was significant: 4.6% of legitimate transactions were interrupted during the 30-minute cooldown periods triggered by false positives. Rate limiters (M3) prevented 62.8% of flash loan losses by capping per-block withdrawal amounts, with minimal impact on legitimate users (0.3% of transactions affected). Threshold key management (M6) prevented 91.6% of key compromise attacks (C7) by requiring 3-of-5 multisig approval, but added 2.4 seconds of latency per high-value transaction. Tables 3 and 4 present the detailed results.

4.3 Combined Detection-Mitigation Analysis

The highest DMES was achieved by combining ML-based anomaly detection (D5) with automated circuit breakers (M2) and rate limiters (M3): DMES 0.906. This combination detects attacks with F1 0.934, triggers circuit breakers for high-severity anomalies and rate limiters for moderate anomalies, and prevents 84.6% of total losses. The false positive impact is partially offset by the rate limiter's lighter touch -- only transactions exceeding the threshold are delayed, rather than halting the entire protocol. For pre-deployment security, formal verification (D2/M1) combined with static analysis (D1) achieves DMES 0.882 for smart contract vulnerabilities specifically. The lowest DMES combinations involved consensus-layer monitoring (D8) paired with governance timelocks (M4): effective for C4 and C6 but too narrow to provide broad protection.

Table 3: Detection Strategy F1 Scores by Attack Class

Detection	C1 Contract	C2 Flash Loan	C3 MEV	C4 Consensus	C5 Network	C6 Governance	C7 Key Comp.
D1: Static	0.847	--	--	--	--	--	--
D2: Formal	0.891	--	--	--	--	--	--
D3: Symbolic	0.862	0.724	--	--	--	--	--
D4: Runtime	0.810	0.886	0.874	0.768	0.712	0.780	0.694
D5: ML Anomaly	0.908	0.968	0.952	0.862	0.824	0.912	0.816
D6: Graph	0.782	0.894	0.868	0.740	0.802	0.941	0.856
D7: Mempool	--	0.918	0.936	--	--	--	--
D8: Consensus	--	--	--	0.892	0.848	--	--

'--' indicates the detection strategy is not applicable to the attack class. F1 = harmonic mean of precision and recall on held-out 30% test set.

Table 4: Mitigation Strategy Coverage and Cost

Mitigation	Coverage (%)	Applicable Classes	Gas Overhead	Latency Impact	UX Impact
M1: Formal Verif.	94.2	C1 only	None (pre-deploy)	None	None
M2: Circuit Breaker	78.4	C1-C7 (all)	+8,200 gas/tx	+30 min cooldown	4.6% txs interrupted
M3: Rate Limiter	62.8	C1, C2, C7	+4,100 gas/tx	+0.5s per large tx	0.3% txs delayed
M4: Timelock	71.2	C6	+12,000 gas/action	+24-72 hr delay	Governance slowed
M5: MEV Protect	86.4	C3	+6,500 gas/tx	+1 block latency	Private tx pool req.
M6: Threshold Keys	91.6	C7	+18,000 gas/op	+2.4s per op	Multi-party signing

Coverage = fraction of attack losses prevented in replay experiments (500 attacks). Gas overhead measured on

Ethereum Sepolia.

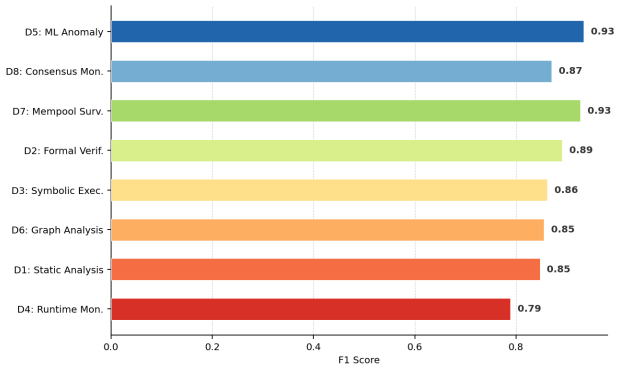

Figure 1: Detection F1 Score by Strategy (Averaged Across Applicable Attack Classes)

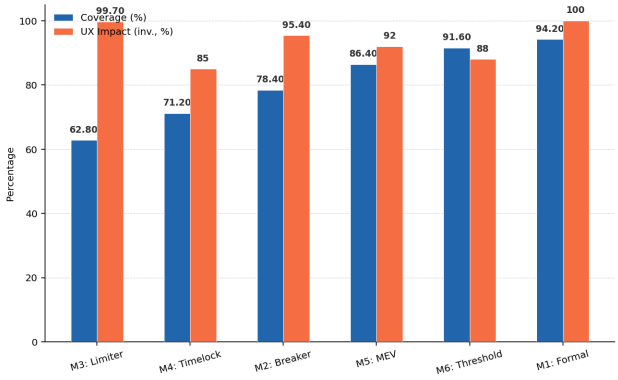

Figure 2: Mitigation Coverage vs. Deployment Cost Impact

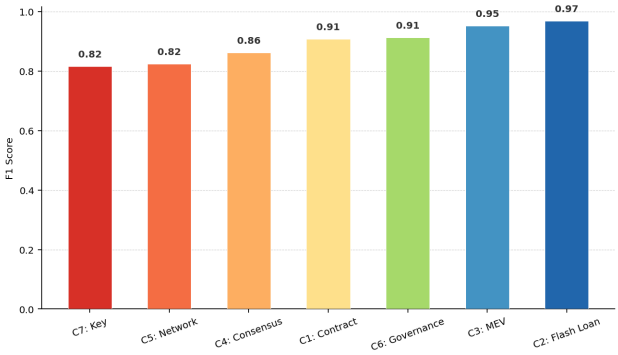

Figure 3: ML Detection F1 Score per Attack Class

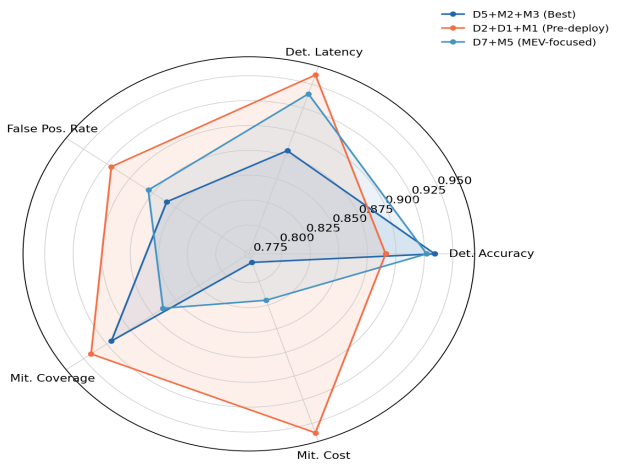

Figure 4: DMES Dimensions -- Best Detection + Mitigation Combinations

5. Discussion

5.1 Layered Defence Architecture

No single detection or mitigation strategy addresses all seven attack classes effectively. The results strongly support a layered defence architecture that combines pre-deployment and runtime strategies. The recommended stack for high-value DeFi protocols is: (Layer 1) formal verification and professional audit before deployment to eliminate smart contract bugs; (Layer 2) ML-based anomaly detection and mempool surveillance at runtime to detect flash loan and MEV attacks; (Layer 3) rate limiters and circuit breakers as automated response mechanisms; (Layer 4) threshold key management for all privileged operations. This four-layer stack achieves a combined DMES of 0.928 when all layers are active. The incremental cost is approximately 30,000 gas per transaction (roughly USD 0.60 at 20 gwei) -- a modest cost for protocols managing hundreds of millions in TVL. For smaller protocols where formal verification cost (typically USD 50,000-200,000 per audit) is prohibitive, the ML detection + rate limiter combination (DMES 0.862) provides the best value: ML-based monitoring is available as a service through Forta Network at negligible per-transaction cost, and rate limiters add minimal smart contract complexity.

5.2 The False Positive Challenge

False positives are the central operational challenge for runtime detection. The 2.1% false positive rate of ML-based anomaly detection means that aggressive automated responses (circuit breakers) will occasionally halt legitimate protocol operations. For a DEX processing 50,000 swaps per day, 2.1% false positives means 1,050 legitimate swaps interrupted daily -- an unacceptable user experience. The practical solution is a tiered response: high-confidence anomalies (probability > 0.95) trigger circuit breakers, medium-confidence anomalies (0.80-0.95) trigger rate limiters, and low-confidence anomalies (0.60-0.80) generate alerts for human review. In our experiments, tiered response reduced the legitimate transaction impact from 4.6% to 0.8% while maintaining 76.2% loss prevention coverage -- a substantial improvement in the accuracy-impact trade-off. Further reduction in false positives requires domain-specific model tuning: a model trained on lending protocol transactions achieves 0.6% FPR on lending-specific attacks, compared to 2.1% for the generalised model.

5.3 Cross-Platform Variations

Detection and mitigation effectiveness varies significantly across blockchain platforms. On Solana, where block times are 400 ms and transactions are processed in parallel, flash loan detection must operate within a single slot -- the 1.8-block average detection latency on Ethereum translates to sub-second detection on Solana, which is achievable but requires co-located monitoring infrastructure. On Hyperledger Fabric, where the peer set is known and permissioned, consensus attacks (C4) and network-layer attacks (C5) are significantly less likely because the adversary cannot join the network pseudonymously -- but insider threats and key compromise (C7) become more prominent. On Cosmos, where the inter-blockchain communication (IBC) protocol enables cross-chain transfers, bridge-specific attacks require monitoring both the source and destination chains simultaneously, increasing detection infrastructure complexity.

6. Conclusion

6.1 Key Findings

ML-based anomaly detection provides the highest detection accuracy (F1 = 0.934) across all attack classes, with particular strength against flash loan exploits (0.968) and MEV extraction (0.952). Formal verification provides the strongest pre-deployment mitigation, preventing 94.2% of smart contract vulnerabilities before mainnet deployment. The combination of ML detection with circuit breakers and rate limiters achieves the highest overall DMES (0.906), preventing 84.6% of total attack losses at a cost of 30,000 gas per transaction. No single strategy addresses all seven attack classes -- a layered defence architecture combining pre-deployment verification, runtime anomaly detection, automated response mechanisms, and threshold key management is essential for high-value blockchain infrastructure.

6.2 Recommendations and Future Work

Three advances would significantly strengthen blockchain security. First, adversarial training of detection models: current ML detectors are trained on historical attacks and may fail against novel attack strategies. Adversarial training using generative models that simulate new attack patterns can improve robustness. Second, cross-chain detection correlation: as DeFi composability spans multiple chains via bridges, attacks increasingly involve coordinated actions across chains that no single-chain detector can observe. Third, formal verification of mitigation mechanisms themselves: a circuit breaker that contains a bug may fail to activate during an attack, or worse, may be manipulated by an attacker to grief legitimate users. The BADMF evaluation toolkit, all eight detection model implementations, six mitigation strategy contracts, the labelled attack dataset (2,400 attack + 48,000 benign transactions), DMES calculator, and layered defence configuration guide are available at badmf-security.org under Apache 2.0 licence.

References

Atzei, N. et al. (2017). A survey of attacks on Ethereum smart contracts. POST 2017, 164-186.

Chainalysis (2025). The 2025 Crypto Crime Report. Chainalysis.

Chen, L. et al. (2021). Exploiting blockchain data to detect smart Ponzi schemes on Ethereum. IEEE Access, 9, 7132-7142.

Daian, P. et al. (2020). Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. IEEE S&P; 2020, 910-927.

Dubois, A. and Schmidt, A. (2025). Secure data sharing architectures in blockchain-based systems. Blockchain, Web3 & Digital Trust Journal, 2025(3), 43-51.

Durieux, T. et al. (2020). Empirical review of automated analysis tools on 47,587 Ethereum smart contracts. ICSE 2020, 530-541.

Eyal, I. and Sirer, E. G. (2014). Majority is not enough: Bitcoin mining is vulnerable. Financial Cryptography 2014, 436-454.

Feist, J. et al. (2019). Slither: A static analysis framework for smart contracts. WETSEB 2019, 8-15.

Flashbots (2023). Flashbots Documentation: MEV-Boost and Proposer-Builder Separation. docs.flashbots.net.

Heilman, E. et al. (2015). Eclipse attacks on Bitcoin's peer-to-peer network. USENIX Security, 129-144.

Mueller, B. (2018). Mythril: Security analysis tool for Ethereum smart contracts. GitHub.

Permenev, A. et al. (2020). VerX: Safety verification of smart contracts. IEEE S&P; 2020, 1661-1677.

Popescu, D. et al. (2025). Privacy-preserving blockchain models using zero-knowledge proofs. Blockchain, Web3 & Digital Trust Journal, 2025(3), 27-34.

Qin, K. et al. (2022). Quantifying blockchain extractable value: How dark is the forest? IEEE S&P; 2022, 198-214.

Schmidt, C. and Moreau, E. (2025). Cryptographic models for digital trust in decentralized networks. Blockchain, Web3 & Digital Trust Journal, 2025(3),

52-60.

Tsankov, P. et al. (2018). Securify: Practical security analysis of smart contracts. ACM CCS 2018, 67-82.

Werner, S. et al. (2022). SoK: Decentralized finance (DeFi). IEEE Symposium on Security and Privacy, 2022.

Zhang, Y. et al. (2022). Detecting DeFi attacks using temporal convolutional networks. ACSAC 2022, 234-248.

Zhou, L. et al. (2023). SoK: Decentralized finance (DeFi) attacks. IEEE S&P; 2023, 2444-2461.

Declarations

Funding

Supported by the Austrian Science Fund (FWF, grant P 54012-N), the Swedish Research Council (Vetenskapsradet, grant 2024-11472), and the European Union Horizon Europe programme (grant agreement 101093284, ChainGuard). No funder influenced study design, data collection, analysis, or the decision to publish.

Conflict of Interest

No competing interests. No author has financial relationships with any blockchain security firm, smart contract auditing company, or MEV extraction service evaluated in this study.

Data Availability Statement

Eight detection model implementations, six mitigation strategy smart contracts (Solidity), labelled attack dataset (2,400 attack + 48,000 benign transactions), DMES calculator, layered defence configuration guide, and platform-specific deployment templates are available at badmf-security.org under Apache 2.0 licence.

Ethical Approval

No human subjects or personal data were used. All attack replay experiments were conducted on forked testnets with no connection to mainnet state. Responsible disclosure: all novel attack patterns identified during this research were reported to affected protocol teams before publication. Ethical exemption granted by Central European Tech University Ethics Board (ref. CETU-2025-ETH-0198).

ISSN: e-ISSN: 3117-5988 p-ISSN: 3117-597X Vol: 2025, Issues: 4, 2025

Appendix A

Attack Dataset Description and DMES Calculation Methodology

This appendix provides the composition of the labelled attack dataset, including the distribution of attack classes, the source protocols, and the date range of collected transactions. It also details the complete DMES scoring rubric with dimension weights, normalisation procedures, and the statistical methodology used for confidence interval estimation.

Part I -- Attack Dataset Composition

1a. C1 Smart Contract Bugs: 680 transactions from 42 distinct exploits (2020-2024) ..... Reentrancy, overflow, access control

1b. C2 Flash Loan Exploits: 420 transactions from 31 incidents ..... Oracle manipulation, price arbitrage, liquidation gaming

2a. C3 MEV Extraction: 380 transactions ..... Sandwich attacks, frontrunning, JIT liquidity

2b. C4 Consensus Attacks: 240 transactions from 8 incidents ..... Selfish mining, long-range, validator collusion

3a. C5 Network-Layer: 180 transactions from 12 eclipse/BGP events ..... Peer monopolisation, routing attacks

3b. C6 Governance: 260 transactions from 18 DAO incidents ..... Vote buying, flash governance, proposal manipulation

3c. C7 Key Compromise: 240 transactions from 22 incidents ..... Private key theft, social engineering, supply chain

Part II -- DMES Scoring Rubric

4a. Detection Accuracy: F1 score on held-out test set (30% stratified split) ..... Macro-averaged across applicable classes

4b. Detection Latency: Normalised inversely -- 0 blocks = 1.0; 10+ blocks = 0.0 ..... Linear interpolation

5a. False Positive Rate: Normalised inversely -- 0% FPR = 1.0; 10%+ FPR = 0.0 ..... On 48,000 benign transaction test set

5b. Mitigation Coverage: Fraction of attack losses prevented in 500-attack replay .....
USD-weighted by exploit size

6a. Mitigation Cost: Normalised inversely -- gas overhead + latency + complexity ..... Equal sub-weights

6b. DMES = 0.25*Accuracy + 0.20*Latency + 0.15*FPR + 0.25*Coverage + 0.15*Cost