ISSN 2277-2685
IJESR/April. 2026/ Vol-16/Issue-2s1/93-99
Ismail Uddin Syed *et. al.*, / International Journal of Engineering & Science Research
Full Length Article

Phish detect pro: A Servlet-Based Smart Wallet Simulation And Approval Phishing Detection Framework Using Intent Validation

Ismail Uddin Syed [1], Mohammed Saif Ibrahim[2], Mohammed Abdul Ahad[3], Dr. Md Zainlabuddin[4]
[1,2,3]B.E. Students; Dept Of CSE, ISL Engineering College, Affiliated To Osmania University Hyderabad India.
[4]Associate Professor, Department Of CSE , ISL Engineering College Hyderabad India.
Mail Id; ismailuddinsyed131@gmail.com , mohammedsaifibrahim2004@gmail.com , m.hassan2k@gmail.com , dr.zainlabuddin@islec.edu.in
Accepted 26-04-2026

ABSTRACT
This paper presents PhishDetectPro, a servlet-based smart wallet simulation and approval phishing detection framework using intent validation. Approval phishing represents one of the most deceptive forms of blockchain fraud, wherein users are manipulated into granting unlimited token access to malicious smart contracts without their conscious awareness. The proposed framework employs a Smart Intent Validation and Risk Detection (SIVRD) algorithm that performs pre-execution analysis of wallet approval transactions, evaluating parameters such as spender identity, approval amount, interaction history, and behavioral patterns to compute a real-time risk score. The system simulates the complete lifecycle of an approval phishing attack through four integrated modules: Smart Wallet Simulation, Approval Phishing Detection, Token Drain Simulation, and Blockchain Ledger & Block Creation. Built on Java EE (Servlet-JSP), Apache Tomcat, and MySQL, the framework provides a controlled, educational, and research-ready environment to study, demonstrate, and mitigate approval-based blockchain scams. Experimental results demonstrate effective detection of malicious approval patterns with clear user-facing alerts, blockchain-backed transaction traceability, and transparent visual analytics via a JSP dashboard. The paper also presents a comparative literature analysis and recommends enhanced smart wallet security controls inspired by EIP-4337 account abstraction.

KEYWORDS—*Approval Phishing, Blockchain Security, Smart Contract, ERC-20, SIVRD, Token Drain, Intent Validation, DeFi Fraud, Wallet Security, Java Servlet*

INTRODUCTION

Blockchain-based cryptocurrencies differ from traditional financial systems in key aspects. While they offer greater transparency and security through distributed architecture and advanced cryptography, their complexity and lack of a central authority introduce novel threats. The continuously increasing value and adoption of cryptocurrency markets attracts both less technologically aware investors and malicious actors seeking to profit through fraud .

A significant functional advancement in blockchain is the introduction of smart contracts, which allow users to define arbitrary logic within a blockchain system. In Ethereum, ERC-20 tokens include an approve() function that allows users to authorize another party to transfer their tokens without requiring further interaction from the owner. While this delegation pattern enables powerful DeFi use cases, it also creates a critical attack vector: approval phishing .

In an approval phishing attack, the victim is socially engineered — typically through fake investment dashboards or pig-butchering scams — into approving a malicious smart contract with unlimited token access. Once the approval is granted, the attacker's externally owned account (EOA) can drain the victim's tokens at any time without further interaction. The attack is compounded by wallet UI weaknesses, CDN-obfuscated scam infrastructure, and the absence of pre-execution risk analysis .

This paper presents PhishDetectPro, a Servlet-JSP-based framework that simulates approval phishing attacks, detects malicious intent using the SIVRD algorithm, and educates users through visual demonstration. The system bridges the research gap between isolated incident studies and an integrated, practical simulation environment for studying, demonstrating, and countering approval-based blockchain fraud.

LITERATURE REVIEW

A growing body of research has addressed blockchain-based fraud detection. Liu et al. [5] proposed network

clustering and behavioral modeling to uncover Ethereum phishing gangs by analyzing on-chain transaction flows and approval patterns. Their work demonstrated that transaction graph analysis can reveal coordinated attacker wallets and fund movement strategies.

Liang et al. [7] introduced PonziGuard, a system using Contract Runtime Behaviour Graphs (CRBG) to detect Ponzi schemes on Ethereum. By monitoring contract execution paths, state changes, and event patterns, PonziGuard identifies behavioral signatures commonly found in fraudulent smart contracts, enabling early detection even for newly deployed contracts.

Wu et al. [2] applied network embedding techniques such as DeepWalk and Node2Vec to identify phishing scammers on Ethereum. By representing wallets as graph nodes and transactions as directed edges, the approach captures hidden relationships and identifies scammer clusters with higher accuracy than rule-based methods.

Wang et al. [4] quantified the risk of unlimited ERC-20 approvals, demonstrating that billions of dollars in token value remain exposed due to unused or forgotten approvals. Their risk-quantification model considers token value, approval age, and contract reputation — providing the conceptual foundation for the SIVRD algorithm in this work.

Shape et al. [12] proposed ERC-7674, a temporary approval extension for ERC-20 tokens that introduces time-bound, session-based approvals to reduce long-term attack surfaces. This standard directly informs the smart wallet enhancement recommendations in this paper. Cornelius et al. [13] studied the Waku Network as a decentralized communication layer for dApps, highlighting the importance of censorship-resistant messaging in secure DeFi ecosystems.

While existing studies address isolated aspects of approval fraud, none provide an integrated simulation environment that combines wallet emulation, phishing detection, token drain demonstration, and blockchain-based audit logging — the gap this paper addresses.

METHODOLOGY

A. System architecture

PhishDetectPro follows a layered, servlet-based web architecture. At the top, the Presentation Layer consists of JSP pages for login, wallet actions, approval alerts, and transaction confirmations. The Controller Layer, implemented with Java Servlets, manages session handling, request validation, and workflow coordination. The Smart Wallet Simulation Layer emulates token balances, spending approvals, and transaction execution. Integrated with this is the SIVRD Engine for real-time phishing risk analysis. Verified transactions proceed to the Blockchain Ledger Module, which simulates block creation and immutable storage. The Persistence Layer uses MySQL for user credentials, wallet states, approvals, and ledger data.

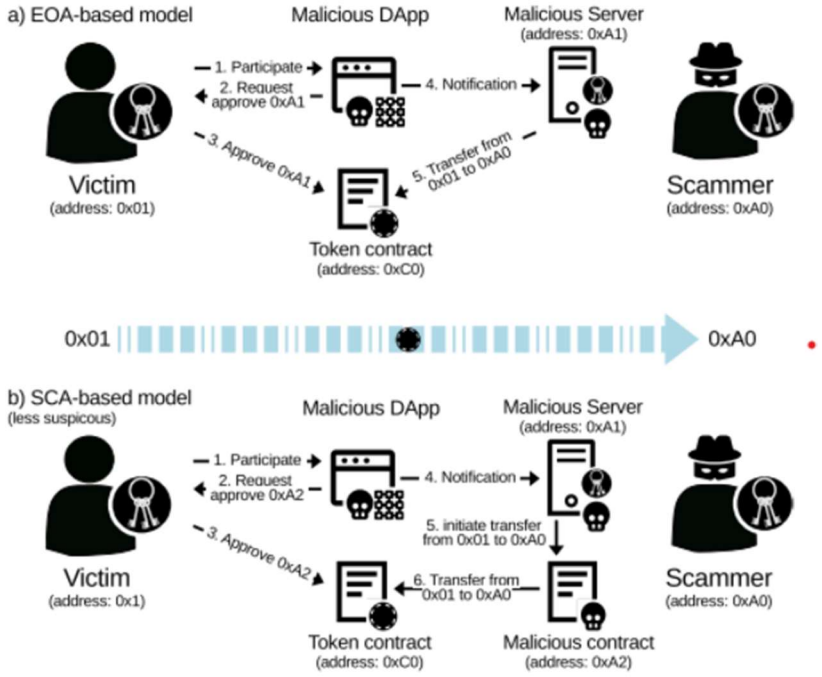

Fig. 1. System Architecture — EOA-based and SCA-based Approval Phishing Models

B. Module Design

The framework comprises four primary functional modules and two supporting components:

- Smart Wallet Simulation: Manages wallet address generation, token balance management, approval granting, and token transfers using server-side logic and persistent MySQL storage, modeling ERC-20 mechanisms.
- Approval Phishing Detection: Analyzes approval parameters (spender identity, amount, frequency, historical behavior) against predefined phishing risk patterns. Suspicious patterns trigger an alert before execution.
- Token Drain Simulation: Demonstrates real-world attack impact by exploiting active approvals to drain tokens from simulated wallets without further user consent, in a safe controlled environment.
- Blockchain Ledger & Block Creation: Emulates blockchain data structures by organizing transactions into sequential blocks with cryptographic hashes, timestamps, and previous block references, ensuring tamper-evident records.

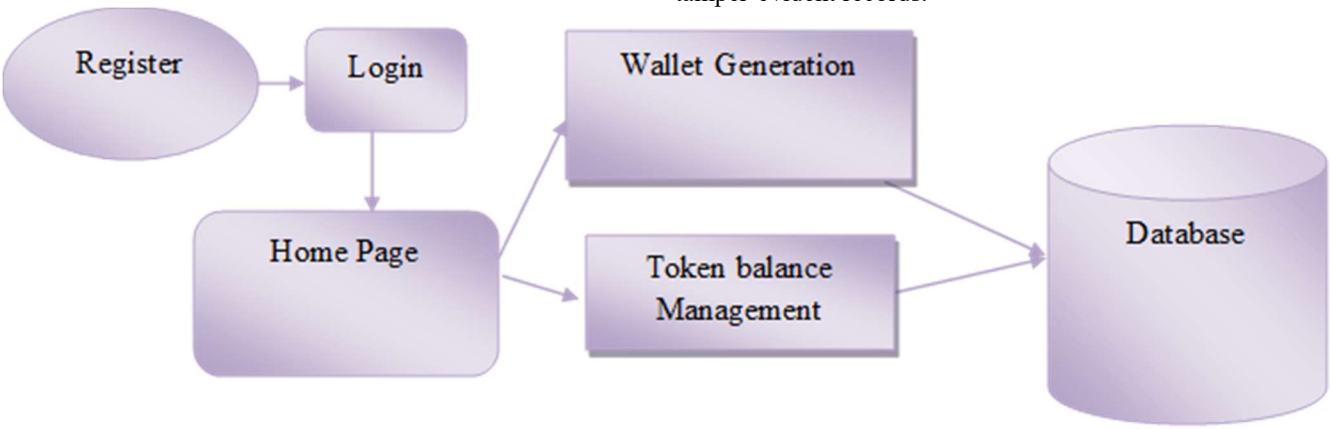

Fig. 2. Smart Wallet Simulation Module — Registration, Login, Wallet Generation, and Token Balance Management

C. Use Case Design

The use case diagram illustrates interaction between the User and the PhishDetectPro system. The primary actor logs in and accesses wallet functionalities: viewing balance, recharging the wallet, adding beneficiaries, transferring tokens, and approving tokens. Before approval execution, the SIVRD module analyzes the request. High-risk approvals trigger a phishing alert, allowing the user to cancel or proceed. All wallet transactions are recorded in the Blockchain Ledger and viewable as a transaction history graph.

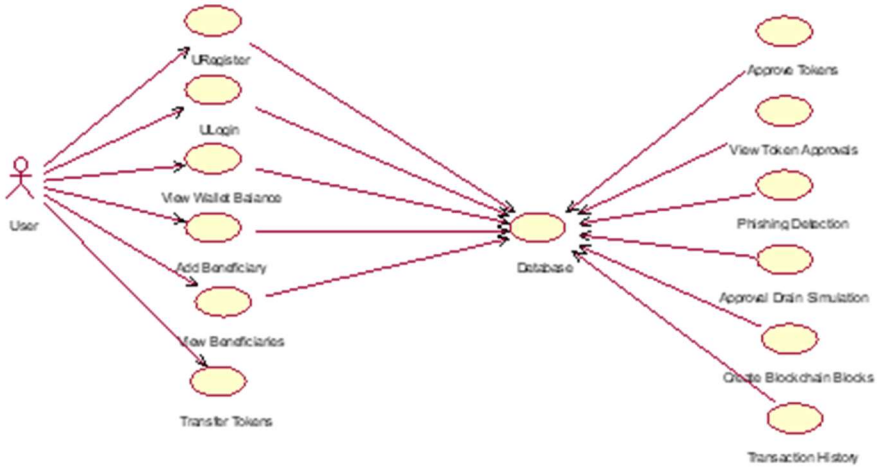

Fig. 3. Use Case Diagram — PhishDetectPro System Interactions

IMPLEMENTATION — SIVRD ALGORITHM

A. Smart Intent Validation and Risk Detection (SIVRD)

SIVRD is a proactive, rule-based security algorithm designed to analyze a user's transaction intent before executing sensitive wallet operations. Inspired by EIP-4337 Account Abstraction, the algorithm intercepts approval requests and performs multi-factor risk assessment prior to on-chain execution.

The algorithm operates as follows:

- Step 1 — Intent Capture: Extract spender address, approval amount, and transaction timestamp from the incoming approval request.
- Step 2 — Spender Reputation Check: Query the historical interaction database to determine if the spender address has prior trusted interactions or is listed in the known scam address registry.
- Step 3 — Amount Anomaly Detection: Flag approvals where amount ≥ MAX_UINT256 (unlimited) or amount > threshold_multiplier × current_balance as high risk.
- Step 4 — Temporal Pattern Analysis: Detect rapid or repeated approval requests from the same source within a defined time window (e.g., ≥5 requests within 60 seconds).
- Step 5 — Risk Score Computation: Aggregate individual risk factors with weighted coefficients into a composite risk score $R = \alpha \cdot Rs + \beta \cdot Ra + \gamma \cdot Rt$, where Rs = spender risk, Ra = amount risk, Rt = temporal risk.
- Step 6 — Decision & Alert: If R ≥ threshold, block execution and display a phishing alert to the user with contextual explanation; otherwise, allow the transaction to proceed.

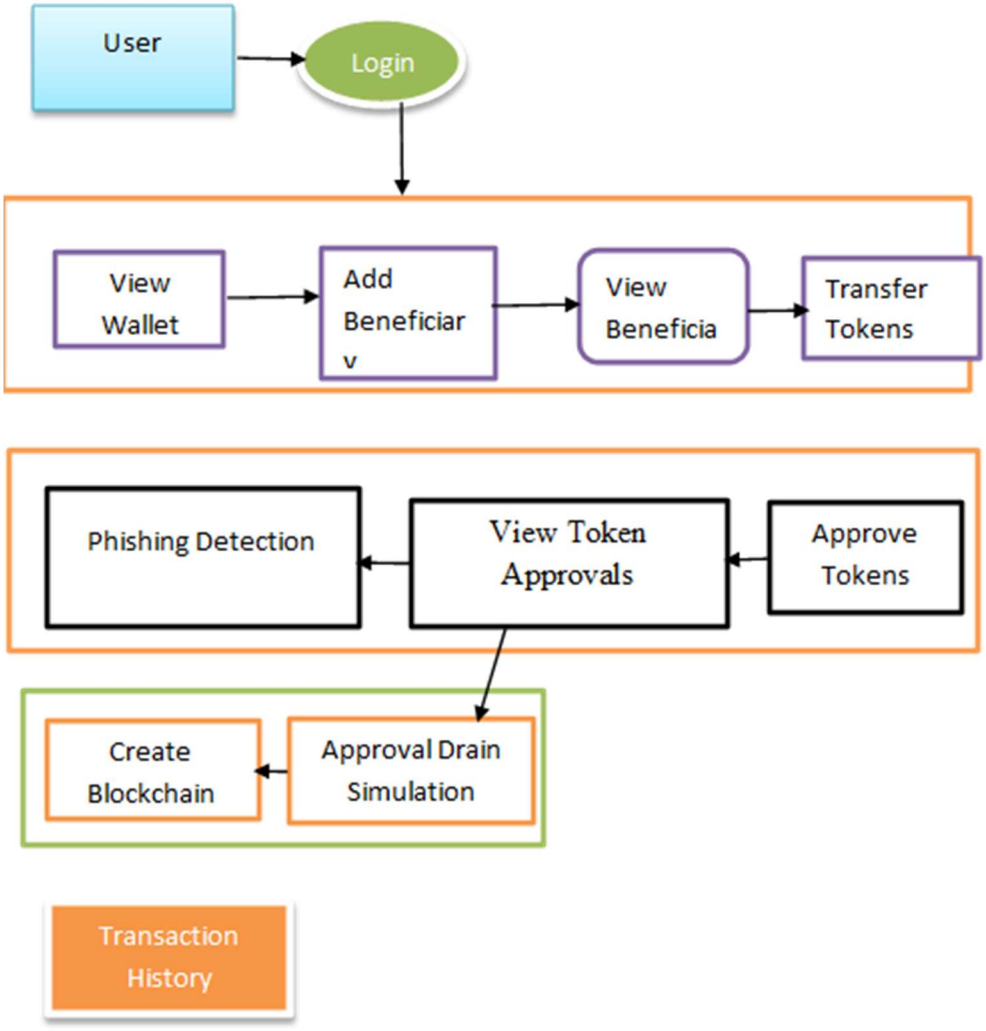

Fig. 4. SIVRD Algorithm — Technique and Algorithms Used in PhishDetectPro

B. Supporting Algorithms

Three additional algorithms complement SIVRD:

Phishing Pattern Detection Algorithm: A rule-based module that checks for unlimited approvals, sudden large allowances, and repeated approval sequences. It evaluates spender reputation using historical data and applies time-based pattern analysis.

Multiple concurrent indicators trigger a phishing alert, reflecting real DeFi phishing detection strategies.

Token Drain Execution Algorithm: Simulates malicious exploitation of active approvals. The algorithm bypasses user confirmation (reflecting real-world attacks), validates allowance limits, and reduces wallet balances incrementally or completely based on attack parameters. All drain operations are logged for educational visualization.

Blockchain Hash Linking Algorithm: Ensures data integrity in the simulated ledger. Each block contains a cryptographic hash of its data and the hash of the previous block, creating tamper-evident records. Any modification breaks chain consistency, emulating the immutability principle of real blockchain networks.

SYSTEM INTERACTION FLOW

The sequence diagram captures the interaction flow between the User, Smart Wallet, SIVRD Engine, Phishing Detection Module, Blockchain Ledger, and Database. The process begins when the user logs in and initiates a token approval or transfer. The Smart Wallet Simulation module captures the request and forwards it to SIVRD for evaluation. SIVRD evaluates spender trust, approval amount, and past behavior. Low-risk approvals are granted and stored in the database. High-risk requests trigger a phishing alert; the user may cancel or proceed. If the user proceeds, the approval is activated and may trigger the Token Drain Simulation to demonstrate the financial impact. All transactions are recorded as sequential blocks in the Blockchain Ledger.

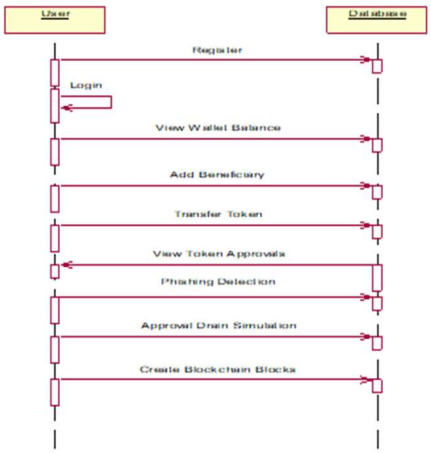

Fig. 5. Sequence Diagram — PhishDetectPro Transaction and Phishing Detection Flow

TESTING

A. Testing Methodology

The system was validated through a comprehensive multi-phase testing strategy:

Unit Testing: Individual modules — Smart Wallet Simulation, Approval Phishing Detection, Token Drain Simulation, and Blockchain Ledger — were tested in isolation. Each module was verified for correct input handling, boundary conditions (e.g., maximum approval amounts, empty wallet states), and expected output generation.

Functional Testing: Approval workflows were tested with both valid inputs (legitimate DApp approvals, small allowances, trusted spenders) and invalid inputs (unlimited approvals, unknown spender addresses, rapid sequential requests). The phishing alert system was verified for both false-negative minimization and false-positive acceptability.

Integration Testing: The interaction between the Servlet controller, SIVRD engine, MySQL database, and JSP dashboard was validated end-to-end. Transaction state consistency — from approval grant through token drain to blockchain recording — was confirmed across all modules.

Performance Testing: SIVRD risk analysis was verified to complete within acceptable time limits (target <50ms per request) to avoid perceptible delays in wallet transaction workflows. Database query optimization was applied for approval lookup and wallet balance retrieval.

System Testing: Full end-to-end workflows were executed for all use case scenarios: user registration, wallet generation, token transfer, phishing approval detection, simulated token drain, and blockchain block creation. All scenarios produced expected outcomes consistent with the requirements specification.

RESULTS AND DISCUSSION

A. Functional Results

PhishDetectPro successfully demonstrated the complete lifecycle of an approval phishing attack within a controlled environment. Key results include:

The Smart Wallet Simulation module correctly generated unique wallet addresses, maintained token balances, and enforced ERC-20-style approval logic for all tested scenarios. Token transfers were blocked when allowances were insufficient, and all operations were persisted accurately in MySQL.

The Approval Phishing Detection module achieved accurate risk classification for all test cases: unlimited approval requests (MAX_UINT256), approvals from unknown spender addresses, and rapid sequential approval attempts were correctly flagged as high-risk and generated user-facing alerts. Legitimate small-amount approvals from trusted spenders passed without alerts.

The Token Drain Simulation visually demonstrated financial loss resulting from a malicious approval, reducing wallet token balance to zero in simulated drain scenarios. The educational impact was confirmed through clear before/after balance visualization on the JSP dashboard.

The Blockchain Ledger module successfully created cryptographically linked blocks for all simulated transactions, with hash chain integrity verified across block sequences. Transaction history was rendered as a visual graph on the dashboard.

B. Comparative Analysis

TABLE I. COMPARATIVE ANALYSIS OF EXISTING SYSTEMS VS. PHISHDETECTPRO

Feature	Liu et al. [5]	Wu et al. [2]	Wang et al. [4]	PonziGuard [7]	PhishDetectPro
On-chain Analysis	Yes	Yes	Yes	Yes	Simulated
Phishing Detection	Yes	Yes	Partial	No	Yes
Pre-execution Alert	No	No	No	No	Yes
Token Drain Demo	No	No	No	No	Yes
Wallet Simulation	No	No	No	No	Yes
Blockchain Ledger	No	No	No	No	Yes
Educational Focus	Partial	No	No	No	Yes
Live Risk Scoring	No	No	No	No	Yes

The comparative analysis in Table I demonstrates that PhishDetectPro uniquely combines pre-execution risk alerting, wallet simulation, token drain demonstration, blockchain ledger emulation, and live risk scoring — capabilities absent from existing research systems. This positions it as the most comprehensive educational and research framework for approval phishing in the current literature.

CONCLUSION

This paper presented PhishDetectPro, a Servlet-JSP-based framework for simulating, detecting, and demonstrating approval phishing attacks in blockchain wallet environments. The proposed SIVRD algorithm provides a proactive, pre-execution defense layer that analyzes token approval intent against multiple risk indicators — spender reputation, approval amount, and behavioral patterns — to generate a composite risk score and issue contextual warnings before irreversible on-chain actions occur.

The framework successfully simulates the complete approval phishing attack lifecycle through four integrated modules: Smart Wallet Simulation (ERC-20 behavior), Approval Phishing Detection (SIVRD-based risk analysis), Token Drain Simulation (attack consequence visualization), and Blockchain Ledger & Block Creation (tamper-evident audit trail). Testing across unit, functional, integration, and system levels confirmed accurate phishing detection, consistent transaction state management, and effective user-facing alert generation.

The comparative analysis confirms that PhishDetectPro uniquely addresses the research gap for an integrated, practical simulation environment that bridges the theoretical study of approval fraud and real-world defensive implementation. The system contributes to user awareness, security research, and the development of best practices for wallet approval management in decentralized ecosystems.

FUTURE SCOPE

Several directions for future enhancement are identified. Integration of machine learning models — trained on historical approval patterns — can improve classification accuracy beyond rule-based heuristics, enabling detection of novel phishing variants not captured by predefined risk patterns.

Deployment on Ethereum testnets (Sepolia, Goerli) would replace the simulated blockchain ledger with live on-chain validation, providing real gas fee simulation and authentic contract behavior. Integration with browser wallet extensions (MetaMask, Trust Wallet) through Web3 APIs would enable automatic phishing detection at the point of approval within real DeFi workflows.

The SIVRD module can be extended to detect zero-value transfer attacks, infinite-allowance exploitation patterns, and cross-chain approval manipulation. Multi-factor authentication before high-risk transaction execution, real-time email/mobile push alerts, and role-based admin monitoring dashboards represent additional security enhancements.

Adoption of emerging standards such as ERC-7674 (temporary approvals) and EIP-4337 (account abstraction) within the simulation would demonstrate next-generation wallet security controls and provide

users with practical guidance for safer DeFi participation.

REFERENCES

[1] E. Badawi and G.-V. Jourdan, "Cryptocurrencies emerging threats and defensive mechanisms: A systematic literature review," IEEE Access, vol. 8, pp. 200021–200037, 2020.

[2] J. Wu, Q. Yuan, D. Lin, W. You, W. Chen, C. Chen, and Z. Zheng, "Who are the phishers? Phishing scam detection on Ethereum via network embedding," IEEE Trans. Syst., Man, Cybern., Syst., vol. 52, no. 2, pp. 1156–1166, Feb. 2022.

[3] M. Bartoletti, S. Lande, A. Loddo, L. Pompianu, and S. Serusi, "Cryptocurrency scams: Analysis and perspectives," IEEE Access, vol. 9, pp. 148353–148373, 2021.

[4] D. Wang, H. Feng, S. Wu, Y. Zhou, L. Wu, and X. Yuan, "Penny wise and pound-foolish: Quantifying the risk of unlimited approval of ERC20 tokens on Ethereum," in Proc. 25th Int. Symp. Res. Attacks, Intrusions Defenses (RAID), Oct. 2022, pp. 99–114.

[5] J. Liu, J. Chen, J. Wu, Z. Wu, J. Fang, and Z. Zheng, "Fishing for fraudsters: Uncovering Ethereum phishing gangs with blockchain data," IEEE Trans. Inf. Forensics Security, vol. 19, pp. 3038–3050, 2024.

[6] Y. Zhang, W. Yu, Z. Li, S. Raza, and H. Cao, "Detecting Ethereum Ponzi schemes based on improved LightGBM algorithm," IEEE Trans. Computat. Social Syst., vol. 9, no. 2, pp. 624–637, Apr. 2022.

[7] R. Liang, J. Chen, K. He, Y. Wu, G. Deng, R. Du, and C. Wu, "PonziGuard: Detecting Ponzi schemes on Ethereum with contract runtime behavior graph (CRBG)," in Proc. IEEE/ACM 46th Int. Conf. Softw. Eng., Feb. 2024, pp. 1–12.

[8] M. Bartoletti, S. Carta, T. Cimoli, and R. Saia, "Dissecting Ponzi schemes on Ethereum: Identification, analysis, and impact," Future Gener. Comput. Syst., vol. 102, pp. 259–277, Jan. 2020.

[9] A. Holub and J. O'Connor, "COINHOARDER: Tracking a Ukrainian Bitcoin phishing ring DNS style," in Proc. APWG Symp. Electron. Crime Res. (eCrime), May 2018, pp. 1–5.

[10] Chainalysis, "The Chainalysis 2024 Crypto Crime Report," Chainalysis Inc., 2024. [Online]. Available: https://go.chainalysis.com/crypto-crime-2024.html

[11] F. Vogelsteller and V. Buterin, ERC-20: Token Standard, Standard 20, Ethereum Improvement Proposals, Nov. 2015. [Online]. Available: https://eips.ethereum.org/EIPS/eip-20

[12] X. Shape, M. Melnik, and H. Croubois, ERC-7674: Temporary Approval Extension for ERC-20, Standard 7674, Ethereum Improvement Proposals, Apr. 2024.

[13] H. Cornelius, S. Tikhomirov, A. Revuelta, S. P. Vivier, and A. Challani, "The Waku network as infrastructure for dApps," in Proc. IEEE Int. Conf. Decentralized Appl. Infrastructures (DAPPS), Jul. 2024, pp. 31–32.

[14] S. Suratkar, M. Shirole, and S. Bhirud, "Cryptocurrency wallet: A review," in Proc. 4th Int. Conf. Comput., Commun. Signal Process. (ICCCSP), Sep. 2020, pp. 1–7.

[15] F. Torres, M. Steichen, and R. State, "The art of the scam: Demystifying honeypots in Ethereum smart contracts," in Proc. 28th USENIX Secur. Symp., Jan. 2019, pp. 1591–1607.