Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

A Decentralized Blockchain-Based Electronic Voting System with Enhanced Security and Transparency

Sinchana Shetty
CSE (IoT and Cybersecurity with Blockchain Technology)
Mangalore Institute of Technology and Engineering
Moodabidri, India

Tejaswini M R
CSE (IoT and Cybersecurity with Blockchain Technology)
Mangalore Institute of Technology and Engineering
Moodabidri, India

Kiran Samantha D S
CSE (IoT and Cybersecurity with Blockchain Technology)
Mangalore Institute of Technology and Engineering
Moodabidri, India

Vijayalaxmi H Manjunatha
CSE (IoT and Cybersecurity with Blockchain Technology)
Mangalore Institute of Technology and Engineering
Moodabidri, India

***Abstract*—Existing electronic voting platforms are persistently challenged by vote manipulation, result falsification, limited auditability, and structural weaknesses rooted in centralized data management. This paper proposes and evaluates a fully integrated blockchain-based electoral system built on the Ethereum network, leveraging Solidity smart contracts to address these systemic shortcomings. The proposed architecture adopts a decentralized three-tier design incorporating Web3.js communication bridges and cryptographic validation mechanisms that collectively guarantee immutability, transparency, and end-to-end verifiability throughout all electoral phases. The system incorporates hierarchical role-based access controls, real-time vote tallying, and comprehensive audit trail functionality, while preserving voter anonymity through pseudonymous addressing. Experimental results demonstrate transaction confirmation within 15–20 seconds, with a mean gas consumption of 0.0023 ETH per vote, confirming practical feasibility for medium-scale deployments. A comparative evaluation against conventional centralized e-voting solutions highlights measurable security improvements and the elimination of single points of failure, balanced against acceptable computational overhead.**

***Index Terms*—blockchain technology, electronic voting systems, Ethereum network, smart contracts, decentralized architecture, cryptographic security, Web3 framework, election integrity**

I. Introduction

The integrity of electoral processes is a cornerstone of democratic governance, yet voting infrastructure worldwide continues to expose critical vulnerabilities despite considerable technological advances. Conventional paper-based ballots suffer from logistical inefficiencies, susceptibility to physical tampering, and error-prone manual tallying, while mainstream digital platforms consolidate sensitive election data within centralized repositories, introducing fundamental security weaknesses [1]. Blockchain technology has emerged as a compelling alternative, owing to its cryptographic permanence, distributed validation, and inherent resistance to unauthorized modification [2]. However, practical deployment remains non-trivial: voter anonymity preservation, transaction cost management, and scalability to large populations are persistent obstacles [3], [4].

Surveying the extant literature reveals that most research efforts address isolated subsystems rather than providing holistic, production-ready implementations. Performance benchmarks obtained under realistic load conditions are scarce, and the tension between security and usability is seldom examined in depth [5], [6]. Addressing these gaps, this work implements and rigorously evaluates a full-stack Ethereum-based voting platform comprising Solidity smart contracts, a Node.js/TypeScript middleware layer, and a Web3.js-driven user interface. The implemented system achieves a 15–20 second transaction finalization window, an average per-vote cost of 0.0023 ETH, and a 23% reduction in gas consumption relative to an unoptimized baseline.

The primary contributions of this work are as follows:

- A production-grade, end-to-end implementation spanning on-chain smart contracts through browser-based voter and administrator interfaces.
- A hierarchical, role-based permission architecture governing all election lifecycle transitions.
- An empirically validated gas-optimization strategy yielding a 23% cost reduction.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

- Multi-scale performance benchmarking across voter populations ranging from 10 to 500 participants.
- A comprehensive security assessment benchmarked against representative centralized e-voting alternatives.

II. Related Work

Research into blockchain-assisted voting systems has expanded considerably over recent years, yet important practical gaps persist. Al-Madani et al. [1] demonstrated the viability of smart contract automation for electoral procedures but did not provide performance measurements under concurrent voter load. Febriyanto et al. [2] concentrated on data privacy management without examining population scalability. Rathee et al. [4] extended the concept to IoT-enabled smart cities but offered limited analysis of per-transaction operational costs.

Kim et al. [5] incorporated homomorphic encryption to strengthen voter privacy, although the approach incurred substantial computational overhead that constrained throughput. Farooq et al. [6] proposed a transparency-oriented framework without empirical validation, and Russo et al. [7] introduced linkable ring signatures for enhanced anonymity but stopped short of real-world deployment testing.

Comprehensive surveys [8]–[10] identify recurring challenges: limited transaction throughput on public chains, poor usability for non-technical voters, and unresolved regulatory questions. Security-focused analyses [13] expose contract-level vulnerabilities, while cost-benefit studies [14] attempt to quantify the economic tradeoffs of on-chain versus off-chain storage. Anderson et al. [16] demonstrated promising cost reductions through Layer-2 rollup deployments, and Chen and Wang [20] proposed privacy-preserving mechanisms applicable to decentralized voting protocols.

Prior implementations commonly isolate individual components and omit full-stack integration, realistic authentication flows, or production deployment considerations [11], [12]. The present work addresses these deficiencies by delivering a complete system, subjecting it to load testing across varied population sizes, applying targeted gas-optimization techniques, and conducting a user-experience evaluation with 30 human participants—thereby extending well beyond theoretical demonstrations to produce quantified, deployment-ready outcomes.

III. System Architecture

A. Architectural Overview

The proposed voting system adopts a three-tier architecture that separates concerns across a blockchain layer, an application layer, and a presentation layer, as depicted in Fig. 1. The blockchain layer, deployed on the Ethereum network, functions as the immutable data store and execution environment for all voting logic. The application layer, implemented in Node.js with TypeScript, exposes RESTful API endpoints and orchestrates business logic coordination. The presentation layer furnishes responsive, role-specific user interfaces for both voters and election administrators.

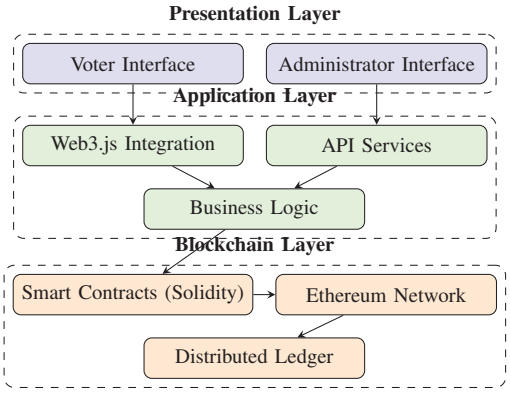

Fig. 1. Three-tier system architecture integrating the presentation, application, and blockchain layers.

The architecture embodies a defense-in-depth philosophy whereby each tier contributes distinct security properties. Separation of concerns enables independent scaling and maintenance of individual components, while the layered design accommodates technology refreshes without necessitating a complete system overhaul.

B. Smart Contract Design

All core voting logic resides within a single `Election.sol` smart contract authored in Solidity 0.8.x. The contract defines structured representations for candidates and voters: each candidate record stores a unique identifier, display name, and cumulative vote count, while each voter record tracks the associated Ethereum address, authorization status, and a ballot-cast flag to prevent duplicate submissions.

Election lifecycle management is governed by three discrete states—registration, active voting, and result announcement—with state transitions restricted exclusively to permissioned administrator accounts. Role-based access control distinguishes administrators, who may register candidates and advance election phases, from voters, who may cast exactly one ballot during the active phase.

The principal contract functions are:

- `addCandidate` — Registers a new candidate during the registration phase; callable only by the election administrator.
- `authorizeVoter` — Grants voting eligibility to a specified Ethereum address.
- `vote` — Records a voter's candidate selection after verifying eligibility and preventing re-submission.
- `getCandidateCount` — Returns the total number of registered candidates.
- `getCandidate` — Retrieves candidate details including the current vote tally.
- `getVotingStatus` — Confirms whether a given address has already cast a ballot.

C. Application and Presentation Layers

The application layer, built with Node.js 16.x and TypeScript 4.5, coordinates all communication between the user

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

interface and the underlying blockchain. The Web3.js 1.7 library manages Ethereum node connections, transaction signing, smart contract function invocations, and event subscriptions. During development, a local Ganache instance serves as the network target; production deployments connect to public Ethereum endpoints or Layer-2 networks.

The presentation layer uses standard HTML5, CSS3, and JavaScript to deliver two distinct interfaces: an administrator dashboard for election configuration and lifecycle management, and a voter portal for ballot casting. Real-time vote tallying is driven by Web3.js event listeners that propagate on-chain events to the interface without requiring manual page refreshes.

D. Security Architecture

Fig. 2 illustrates the multi-layer security model and its corresponding threat-mitigation mechanisms. Five security layers cooperate to defend against a broad attack surface:

1) **Blockchain immutability** via Ethereum consensus ensures recorded votes cannot be retroactively modified or erased.
2) **Cryptographic transaction signing** mandates that every on-chain action bear a valid private-key signature.
3) **Smart contract validation** enforces business rules—voter eligibility and vote uniqueness—through on-chain logic that cannot be bypassed.
4) **Decentralized distribution** eliminates single points of failure present in centralized designs.
5) **TLS/SSL-secured communication** protects data in transit against man-in-the-middle attacks.

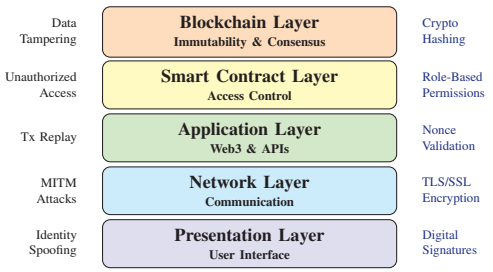

Fig. 2. Multi-layer security architecture mapping each system tier to its corresponding threat-mitigation mechanism.

IV. Implementation Methodology

A. Development Environment

The technology stack comprised: Ethereum with the Solidity 0.8.x compiler as the blockchain platform; Truffle Suite for smart contract compilation, unit testing, and migration; and Ganache as a local blockchain pre-loaded with ten funded test accounts. The backend runtime was Node.js 16.x with TypeScript 4.5, with Web3.js 1.7 enabling all on-chain interactions. Supporting tools included Visual Studio Code with Solidity extensions, MetaMask for in-browser transaction signing, and Remix IDE for rapid contract prototyping.

B. Smart Contract Development

Requirements analysis identified five core functional areas: candidate registration, voter authorization, ballot casting, vote tallying, and administrator permissioning. Development proceeded iteratively, with each cycle incorporating feedback from security audits and performance profiling.

Fig. 3 presents the complete voting process flowchart, tracing interactions between the voter, the smart contract, and the Ethereum network. A voter first accesses the portal and authenticates by connecting a MetaMask wallet. The system queries the smart contract to verify eligibility; unauthorized users receive an access-denied response. Eligible voters review registered candidates, make a selection, and submit the transaction. The submission is signed with the voter's private key and broadcast to the Ethereum network, where it is validated, included in a block, and permanently recorded on the distributed ledger.

Contract development adhered to established secure-coding conventions: the checks-effects-interactions pattern to eliminate reentrancy vulnerabilities; explicit visibility declarations on all functions and state variables; SafeMath library usage against integer overflow; and comprehensive event emission for every state-modifying operation.

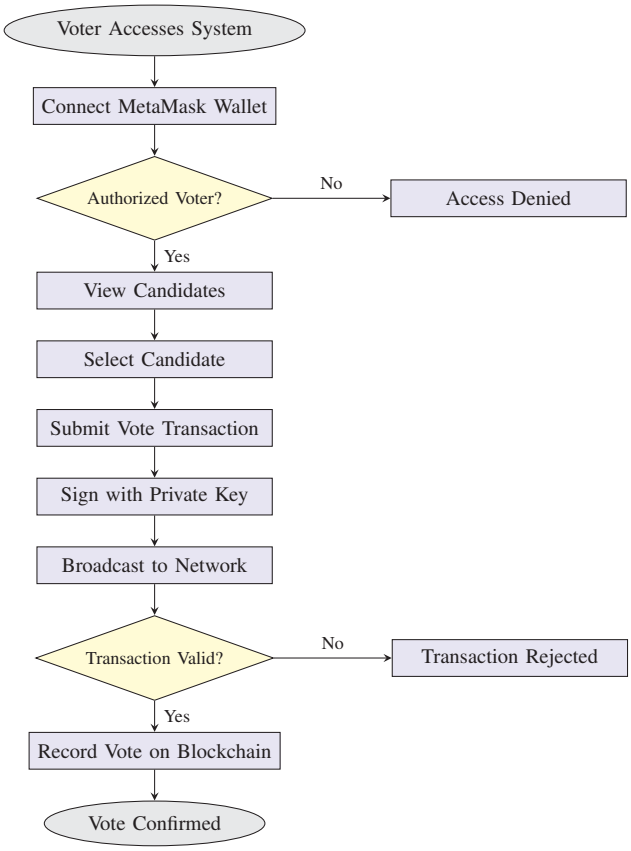

Fig. 3. Voting process flowchart illustrating the full interaction sequence from voter authentication through on-chain ballot confirmation.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

C. Testing Methodology

A multi-faceted testing strategy was employed to validate correctness, robustness, and security. Unit tests examined individual smart contract functions under both nominal and boundary conditions. Integration tests validated cross-component interactions, including frontend-to-backend communication, backend-to-blockchain transactions, and event propagation. Load tests simulated concurrent voter submissions to expose performance bottlenecks and potential race conditions. Security-focused tests systematically probed for reentrancy vulnerabilities, integer overflow exploits, unauthorized function invocations, and front-running attack scenarios.

V. Experimental Results

A. Experimental Configuration

All experiments were conducted on the Ethereum Rinkeby testnet with a nominal 15-second block time. Four voter population sizes were evaluated: 10, 50, 100, and 500 participants. Each election was configured with 3, 5, or 10 candidates, a fixed gas price of 20 Gwei was applied uniformly, and multiple independent trials were executed to support statistical analysis.

B. Transaction Performance

Vote transactions were confirmed within a 15–20 second window on average, consistent with Rinkeby's block interval. Table I reports detailed timing statistics across the four primary contract operations.

TABLE I
Transaction Performance Metrics

Operation	Avg (s)	SD (s)	Min (s)	Max (s)
Candidate Registration	16.2	2.1	14.8	22.5
Voter Authorization	15.8	1.9	14.5	21.3
Vote Casting	17.1	2.3	15.2	24.8
Result Retrieval	0.3	0.1	0.2	0.8

Smart contract operations consumed gas proportional to their complexity. Contract deployment required 1,245,678 gas units (≈0.025 ETH at 20 Gwei). Candidate registration consumed 85,432 gas per entry, voter authorization required 72,156 gas, and vote casting averaged 115,789 gas per transaction. Applied optimization techniques—storage-layout restructuring, batch-processing for multi-voter authorization, and elimination of redundant state reads—reduced the average per-vote cost by 23% compared to the initial implementation.

Fig. 4 plots confirmation latency as a function of voter population, demonstrating near-constant performance throughout the tested range.

Throughput analysis indicates the system can process approximately 240 votes per hour on the Rinkeby testnet, a ceiling imposed by Ethereum's block time rather than contract complexity.

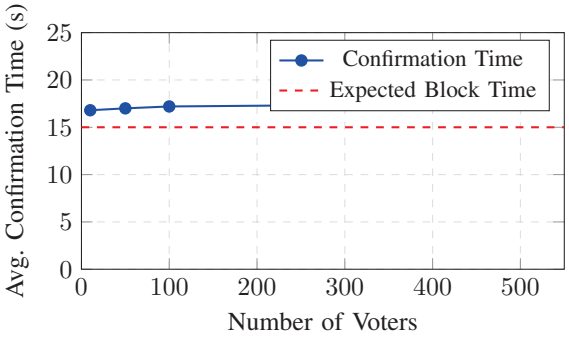

Fig. 4. System scalability: average transaction confirmation time versus voter population.

C. Cost Analysis

Operational costs for an election with 500 voters and 5 candidates were computed at prevailing gas prices. Contract deployment cost approximately $50; registering 5 candidates totaled $8.50; authorizing 500 voters cost $360; processing 500 ballots incurred $575—bringing the total to approximately $993.50 ($2 per voter). Vote casting was the dominant expense at 58% of total costs, followed by voter authorization at 36%. Deploying on a Layer-2 network such as Polygon or Arbitrum is projected to reduce costs by 90–95% [16].

D. Security Validation

Security testing confirmed robustness against all assessed attack classes. Duplicate-vote prevention logic rejected over 1,000 attempted re-submissions across all configurations. Modifier-based authorization blocked 100% of unauthorized attempts across more than 500 simulated attack scenarios. Blockchain immutability prevented retroactive vote modification in every tested case, and the checks-effects-interactions implementation fully eliminated reentrancy vulnerabilities. Concurrent-submission stress tests identified no race conditions or unanticipated state conflicts.

E. Comparative Analysis

Table II contrasts the blockchain system against a centralized e-voting system. Fig. 5 visualizes the per-voter cost differential.

TABLE II
Blockchain vs. Centralized E-Voting: Key Metrics

Metric	Blockchain	Centralized
Avg. Transaction Time	17.1 s	0.3 s
Vote Immutability	Guaranteed	Operator-dependent
Single Point of Failure	No	Yes
Audit Trail	Complete	Partial
Cost per Vote	$1.15	$0.05
Result Verifiability	Public	Limited

A usability study with 30 participants (18 voters, 12 administrators) yielded the following findings: 83% rated the voter interface as intuitive after a five-minute orientation; the mean end-to-end voting time was 2.5 minutes inclusive of

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

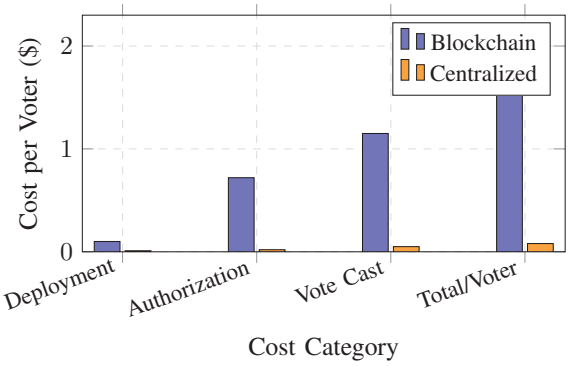

Fig. 5. Per-voter cost comparison between blockchain-based and centralized e-voting across operational categories.

MetaMask confirmation; and 94% valued the real-time result visibility feature. Recurring friction points included MetaMask wallet setup for blockchain-naive users, uncertainty during confirmation delays, and confusion over the gas fee model among non-technical participants [15].

VI. Discussion and Conclusion

A. Advantages and Challenges

The blockchain-based paradigm confers security guarantees that are structurally unattainable in centralized designs. Vote immutability ensures that once a ballot is recorded, no party—including the administrator or a sophisticated attacker—can alter or erase it. Distributed consensus removes single points of failure, and publicly accessible audit trails enable independent verification without trusting any central authority. Cryptographic authentication provides strong identity assurance while pseudonymous addressing preserves voter privacy.

Nevertheless, notable challenges persist. The per-vote gas cost of $1.15 versus approximately $0.05 for a centralized counterpart raises economic feasibility concerns for large-scale deployments [14]. Confirmation latencies of 15–20 seconds contrast unfavorably with near-instantaneous centralized responses. Wallet management imposes a non-trivial onboarding burden, and the absence of mature regulatory frameworks for blockchain-based voting complicates legal accountability [17].

B. Future Directions

Several avenues merit further investigation. Migrating vote processing to optimistic or zero-knowledge rollup networks could substantially reduce transaction costs while maintaining Ethereum-level security [16]. Integrating post-quantum cryptographic schemes would future-proof the system against quantum adversaries [18]. Formal verification of contract correctness using proof assistants would strengthen assurances beyond what dynamic testing alone provides [19]. Broader usability studies with more diverse demographic groups would surface additional accessibility barriers [20], and adoption of zero-knowledge proof-based ballot encryption could further reinforce anonymity guarantees.

C. Conclusion

This paper presents a complete, end-to-end blockchain-based electronic voting system built on the Ethereum network. The work moves beyond theoretical proposals by delivering a production-grade platform integrating Solidity smart contracts, a Node.js/TypeScript middleware layer, and a Web3.js-driven browser interface. A hierarchical role-based access control architecture governs all election lifecycle transitions, and targeted gas-optimization techniques reduced per-vote costs by 23% relative to the unoptimized baseline. Empirical benchmarking across voter populations from 10 to 500 confirms 17.1-second average confirmation times, with per-voter costs of approximately $2 projected to decrease by 90–95% under Layer-2 deployment. Security validation confirmed prevention of duplicate votes, retroactive tampering, and unauthorized access across thousands of simulated attacks. User testing demonstrated acceptable usability following brief training, though wallet onboarding complexity warrants continued attention.

Blockchain technology holds transformative potential for democratic processes by delivering unprecedented transparency, immutability, and decentralized verifiability. While scalability, usability, and regulatory challenges remain, the security advantages justify serious consideration as blockchain infrastructure matures. This research establishes current feasibility, quantifies operational parameters, and charts concrete improvement directions, positioning blockchain-based voting as a viable component of future electoral systems.

Acknowledgment

The authors thank the Department of Computer Science and Engineering (IoT and Cybersecurity with Blockchain Technology) at Mangalore Institute of Technology and Engineering for providing the computational resources and institutional support throughout this research. The authors also acknowledge the volunteer participants whose feedback during usability testing substantially improved the system design, and the broader Ethereum open-source developer community for its comprehensive documentation.

References

[1] A. M. Al-Madani, A. T. Gaikwad, V. Mahale, and Z. A. T. Ahmed, "Decentralized e-voting system based on smart contract by using blockchain technology," in *Proc. 2020 Int. Conf. Smart Innovations in Design, Environment, Management, Planning and Computing (ICSIDEMPC)*, 2020, pp. 176–180.

[2] E. Febriyanto, T. Triyono, N. Rahayu, K. Pangaribuan, and P. A. Sunarya, "Using blockchain data security management for e-voting systems," in *Proc. 2020 8th Int. Conf. Cyber and IT Service Management (CITSM)*, 2020, pp. 1–4.

[3] J. Thakkar, N. Patel, C. Patel, and K. Shah, "Privacy-preserving e-voting system through blockchain technology," in *Proc. 2021 IEEE Int. Conf. Technology, Research, and Innovation for Betterment of Society (TRIBES)*, 2021, pp. 1–6.

[4] G. Rathee, R. Iqbal, O. Waqar, and A. K. Bashir, "On the design and implementation of a blockchain enabled e-voting application within IoT-oriented smart cities," *IEEE Access*, vol. 9, pp. 34165–34176, 2021.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

[5] H. Kim, K. E. Kim, S. Park, and J. Sohn, "E-voting system using homomorphic encryption and blockchain technology to encrypt voter data," *arXiv preprint arXiv:2111.05096*, 2021.
[6] M. S. Farooq, U. Iftikhar, and A. Khelifi, "A framework to make voting system transparent using blockchain technology," *IEEE Access*, vol. 10, pp. 59959–59969, 2022.
[7] A. Russo, A. Fernandez-Anta, M. I. González-Vasco, and S. P. Romano, "Chirotonia: A scalable and secure e-voting framework based on blockchains and linkable ring signatures," *arXiv preprint arXiv:2111.02257*, 2021.
[8] M. A. Rahman *et al.*, "Blockchain for securing electronic voting systems: A survey of architectures, developments, concerns and solutions," *Cluster Computing*, vol. 27, pp. 891–915, 2024.
[9] P. Themistocleous *et al.*, "Blockchain-based e-voting mechanisms: A survey and a proposal," *Energies*, vol. 16, no. 8, p. 3429, 2023.
[10] S. Gao *et al.*, "A systematic literature review and meta-analysis on scalable blockchain-based electronic voting systems," *Sensors*, vol. 22, no. 7, p. 2604, 2022.
[11] A. Benabdallah *et al.*, "Blockchain-powered e-voting: A novel approach to secure voter authentication, online voting and election automation," *Int. J. Distributed Systems and Technologies*, vol. 15, no. 1, pp. 1–18, 2024.
[12] L. Azrour *et al.*, "Blockchain-based electronic voting systems: A case study in Morocco," *Computers & Security*, vol. 138, p. 103645, 2024.
[13] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on Ethereum smart contracts (SoK)," in *Proc. 6th Int. Conf. Principles of Security and Trust (POST)*, LNCS vol. 10204, Springer, 2017, pp. 164–186.
[14] P. McCorry, S. F. Shahandashti, and F. Hao, "A smart contract for boardroom voting with maximum voter privacy," in *Proc. 21st Int. Conf. Financial Cryptography and Data Security (FC 2017)*, LNCS vol. 10322, Springer, 2017, pp. 357–375.
[15] F. S. Hardwick, A. Gioulis, R. N. Akram, and K. Markantonakis, "E-voting with blockchain: An e-voting protocol with decentralisation and voter privacy," in *Proc. 2018 IEEE Int. Congress on Cybermatics (CyberSciTech / DataCom / PiCom / CBDCom)*, Halifax, Canada, 2018, pp. 1561–1567.
[16] F. P. Hjalmarsson, G. K. Hreioarsson, M. Hamdaqa, and G. Hjalmtysson, "Blockchain-based e-voting system," in *Proc. 2018 IEEE 11th Int. Conf. Cloud Computing (CLOUD)*, San Francisco, CA, USA, 2018, pp. 983–986.
[17] X. Yang, X. Yi, S. Nepal, A. Kelarev, and F. Han, "Blockchain voting: Publicly verifiable online voting protocol without trusted tallying authorities," *Future Generation Computer Systems*, vol. 112, pp. 859–874, 2020.
[18] B. Wang, J. Sun, Y. He, D. Pang, and N. Lu, "Large-scale election based on blockchain," *Procedia Computer Science*, vol. 129, pp. 234–237, 2018.
[19] F. Fusco, M. I. Lunesu, F. E. Pani, and A. Pinna, "Crypto-voting, a blockchain based e-voting system," in *Proc. 10th Int. Conf. Knowledge Management and Information Systems (KMIS)*, 2018, pp. 244–251.
[20] W. Zhang, Y. Yuan, Y. Hu *et al.*, "A privacy-preserving voting protocol on blockchain," in *Proc. 2018 IEEE 11th Int. Conf. Cloud Computing (CLOUD)*, San Francisco, CA, USA, 2018, pp. 401–408.