JTSES
Journal of Thermal and Sustainable Energy Systems

https://jtses.com/index.php/home/index

End-to-End Verifiable Blockchain Voting Architecture Integrating Decentralized Identity and Threshold Decryption Protocols

M. Gautham[1], Deena A[2], Jeevan G[3], Jeyachandran K[4], Kanikumar K[5]

[1], Assistant Professor, Department of Electronics and Communication Engineering, Mahendra Engineering College, Namakkal, Tamil Nadu, India.

[2], [3], [4], [5] Department of Electronics and Communication Engineering, Mahendra Engineering College, Namakkal, Tamil Nadu, India.

Email: mgauthamme@gmail.com.

ABSTRACT

Secure electronic voting has become a critical requirement for digital governance because conventional paper, electronic, and internet voting systems expose elections to opaque counting, centralized control, and weak independent verification. Blockchain, smart contracts, and privacy-preserving cryptography provide a technical basis for tamper-resistant electoral records and auditable tallying. Existing blockchain voting approaches still insufficiently couple voter anonymity, eligibility verification, scalable transaction processing, and verifiable encrypted tallying within a complete operational architecture. The objective of this work is to develop and evaluate BBSTVS, a secure and transparent blockchain-based voting system for privacy-preserving electronic elections. The system was implemented on the Polygon zkEVM Layer 2 test network using Solidity smart contracts, W3C decentralized identifiers, ElGamal homomorphic encryption, zk-SNARK eligibility proofs, nullifier-based double-vote prevention, and threshold decryption. A simulated election with 100,000 voters was executed to quantify transaction latency, confirmation time, fraud rejection, tally duration, and receipt verification. Vote casting required 1.2 s on average, transaction confirmation occurred within 2 s, and the complete encrypted tally finished in under 8 s. No fraudulent vote was accepted, and every accepted ballot was linked to a verifiable receipt. The architecture supports public auditability while preserving ballot secrecy. Future research should address post-quantum migration, offline participation, and larger multi-region deployment.

Keywords

INTRODUCTION

Electronic voting research has shifted from simple digitisation toward verifiable, cryptographically protected election infrastructure because conventional electronic voting still concentrates trust in election servers, administrators, and opaque databases. Centralised storage can simplify ballot management, but it also creates attack surfaces for vote alteration, insider manipulation, double voting, service denial, and unverifiable tallying. Blockchain-based voting addresses part of this problem by replacing mutable databases with append-only distributed ledgers, yet ledger immutability alone does not guarantee voter anonymity, eligibility control, ballot secrecy, coercion resistance, or scalable throughput. Recent reviews therefore treat blockchain as an enabling layer rather than a complete voting solution, emphasising that authentication, privacy preservation, smart-contract correctness, and performance validation must be engineered together (Jafar, Aziz, & Shukur, 2021) (Jafar, Aziz, Shukur, & Hussain, 2022) (Berenjestanaki, Barzegar, El Ioini, & Pahl, 2024).

The strongest recent designs combine smart contracts with cryptographic voting primitives. Ethereum-based systems use smart contracts to enforce registration, vote casting, ballot uniqueness, and tally publication, but gas cost, network congestion, and contract vulnerability remain practical constraints (Spanos & Kantzavelou, 2024). Permissioned and enterprise blockchain designs reduce computational overhead and energy cost by avoiding public-chain consensus, yet this usually narrows decentralisation and may reintroduce institutional trust assumptions (González, Frias Mena, Massó Muñoz, Rojas, & Sosa-Gómez, 2022). Systems using distributed identity, elliptic-curve signatures, hash commitments, Merkle proofs, IPFS storage, ring signatures, and homomorphic encryption improve different parts of the election pipeline, but they often optimise one property at

a time: anonymity, integrity, auditability, or efficiency. For this reason, method-level comparison is essential; the field is no longer asking whether blockchain can store votes, but whether a complete election workflow can remain private, scalable, and independently verifiable under realistic participation loads.

Table 1. Comparative synthesis of selected blockchain-based e-voting journal studies.

Reference/Citation	Objective/Aim	Methodology/Approach	Materials/Parameters Studied	Key Findings/Results	Relevance to Current Study
Jafar et al. (2021)	Review blockchain e-voting challenges	Structured literature review	Privacy, speed, verification	△ Privacy-speed tradeoffs highlighted	Frames baseline technical constraints
González et al. (2022)	Build enterprise voting architecture	Hyperledger Fabric smart contracts	Permissioned consensus, HSM, NFT	△ Lower energy via permissioned design	Supports architecture comparison
Neziri et al. (2022)	Preserve voter unlinkability	Dual-blockchain privacy model	Keys, timestamps, salted hashes	△ Stronger anonymity through separation	Guides privacy-layer design
Singh et al. (2023)	Develop Ethereum voting website	Smart contracts with facial recognition	Ethereum, EVM, identity checks	△ Tamper resistance through decentralisation	Supports authentication discussion
Tang et al. (2023)	Enable anonymous authentication	zk-SNARK with Merkle tree	Random addresses, Ethereum, proofs	△ Identity hidden during validation	Directly supports ZKP layer
Pereira et al. (2023)	Propose secure transparent voting	Cryptographic blockchain framework	Signatures, HE, ZKP, BFT	△ Verifiability with ballot privacy	Supports cryptographic integration
Daraghmi et al. (2024)	Design VoteChain for Palestine	Smart contract web architecture	Auditability, mobility, privacy	△ Broader usability requirements addressed	Links transparency with accessibility
Wang et al. (2024)	Improve confidential verifiability	IPFS, contracts, ring signatures	Storage, anonymity, key sharing	△ Better verifiability and storage handling	Supports off-chain storage logic
Spanos and Kantzavelou (2024)	Implement serverless e-voting	Ethereum-only smart contracts	MetaMask, encrypted votes	△ Database removal strengthens security	Supports decentralised deployment model
Elhoseny et al. (2025)	Secure voting with hybrid validation	HAHE, PBFT, EPoA, CNN	Encryption time, throughput, delay	△ Faster cryptographic processing reported	Supports benchmark comparison

Public-sector blockchain research also shows why electoral systems require more than technical immutability. Government deployments frequently struggle to move beyond pilots because policy accountability, institutional trust, data governance, and operational responsibility remain unresolved (Clifton, Fernández-Gutiérrez, & Cagigas, 2023). DAO governance studies similarly show that voting power distribution and incentive

structures affect long-term viability, meaning that decentralised decision systems can still fail if governance rules are poorly designed (Rikken, Janssen, & Kwee, 2023). Consensus research reinforces the same point from a systems perspective: voting protocols, distributed agreement, and blockchain consensus share reliability goals, but their fault models and latency assumptions differ sharply (Dang & Hwang, 2025). In an election platform, therefore, the consensus layer must be selected together with voter-authentication logic, ballot-secrecy mechanisms, and audit procedures rather than treated as a generic blockchain component.

Recent blockchain e-voting studies increasingly use computational prototypes, smart-contract simulations, cryptographic protocol construction, and systematic reviews. Performance trends show stronger transparency and tamper resistance, but privacy, transaction speed, and deployment-scale validation remain uneven. Public-chain studies emphasise decentralisation and auditability, whereas permissioned-chain studies emphasise efficiency and administrative control. Cryptographic studies differ mainly in whether they prioritise zero-knowledge eligibility, homomorphic tallying, anonymous credentials, or off-chain storage.

Overall, the field is moving toward integrated election infrastructures that combine distributed ledgers, smart contracts, decentralised identity, privacy-preserving proofs, and verifiable tallying. The emphasis has shifted from proving blockchain suitability in principle to evaluating complete voting workflows under operational constraints. However, many studies still remain bounded by prototype-scale testing, limited voter models, isolated cryptographic functions, or incomplete performance reporting. This leads directly to unresolved limitations in scalable, privacy-preserving, and end-to-end auditable blockchain voting.

Existing studies fail to address a fully coupled treatment of eligibility verification, anonymous ballot casting, encrypted tallying, receipt-based voter verification, and high-load performance within one coherent architecture. Because reviews identify privacy protection and transaction speed as recurring bottlenecks, systems that validate only conceptual requirements cannot establish deployment readiness. Because permissioned designs reduce overhead but narrow decentralisation, and Ethereum-only prototypes retain gas and congestion constraints, scalability remains inconsistently tested across restricted parameter ranges. Because cryptographic schemes often isolate ZKP, homomorphic encryption, ring signatures, or IPFS storage, the literature lacks consistent coupled analyses linking security guarantees with latency, throughput, contract execution, and voter-side usability.

The present study differentiates itself by integrating blockchain immutability, smart-contract enforcement, decentralised identity, zero-knowledge eligibility validation, ElGamal homomorphic encryption, threshold-based tallying, and receipt verification into a single Layer-2 voting architecture. It also aligns the security model with measurable execution parameters, including vote-casting latency, confirmation time, throughput, tally duration, invalid-vote rejection, and voter-verification capability. This structure directly responds to fragmented prior methods by treating privacy, verifiability, scalability, and auditability as jointly evaluated system requirements rather than independent features.

The objective of the present study is to design and evaluate a secure, transparent, privacy-preserving, and scalable blockchain-based voting system that integrates smart contracts, zero-knowledge proofs, homomorphic encryption, decentralised identity, and verifiable on-chain auditing.

Materials and Methods

Materials

The study was designed as a blockchain-based electronic voting system development and evaluation protocol for the Blockchain-Based Secure and Transparent Voting System (BBSTVS). The primary software materials consisted of Solidity smart contracts, a decentralized voter identity layer, cryptographic ballot-processing modules, and a blockchain transaction interface. The blockchain execution environment was configured on an Ethereum-compatible Polygon zkEVM Layer 2 test network. The smart-contract source files were organized as VoterRegistry.sol, VotingSystem.sol, and ResultPublisher.sol. The development environment included , , , , , and . All software dependencies, compiler versions, and package hashes were recorded before deployment to ensure reproducibility.

Cryptographic materials included elliptic-curve digital signature keys for voter transaction signing, ElGamal public–private key pairs for ballot encryption, zero-knowledge proof parameters for vote-validity verification, and threshold decryption key shares assigned to election trustees. Decentralized identifiers were generated according to the W3C DID data model. Voter credentials were represented by a public DID, a public verification key, and a private signing key stored only on the voter device. No physical chemicals, biological reagents, or consumable laboratory substrates were used. The principal methodological materials and configuration placeholders are summarized in Table 1.

Table 1. Materials, software components, and cryptographic configuration used for BBSTVS implementation.

Component	Specification	Storage / Handling Condition
Blockchain network	Polygon zkEVM-compatible Layer 2 test network	Remote RPC endpoint
Smart-contract language	Solidity	Version-controlled repository
Voter registry contract	VoterRegistry.sol	On-chain deployment
Voting contract	VotingSystem.sol	On-chain deployment
Result contract	ResultPublisher.sol	On-chain deployment
Identity framework	W3C DID-compatible credential	Encrypted local wallet
Ballot encryption	ElGamal encryption	Public election key
Eligibility proof	zk-SNARK circuit	Local proof generation
Signature scheme	ECDSA-compatible signing	Private voter device
Audit storage	Blockchain event logs	Public ledger

Sample Preparation / Fabrication

The experimental sample consisted of a simulated voter population generated from synthetic voter records. Each record contained a unique voter identifier, a DID document reference, a public verification key, and an eligibility flag. Personally identifiable information was not included in the test dataset. Synthetic voter entries were generated using a deterministic pseudorandom seed, [seed value], to permit exact repetition of the experiment. Candidate entries were defined as fixed symbolic labels, and each candidate was assigned a unique integer candidate identifier before contract deployment.

The smart-contract system was prepared in three sequential stages. First, the voter registry contract was compiled and deployed with administrative access restricted to the election authority address. Second, the voting contract was deployed with references to the voter registry, election public key, nullifier registry, and zero-knowledge verifier contract. Third, the result publisher contract was deployed with trustee-quorum parameters and a result-publication access rule. Contract deployment order, constructor arguments, and address mappings were recorded in a deployment manifest. Acceptance of deployment was defined by successful bytecode verification, ABI consistency, and correct initialization of election-state variables.

Figure 1 illustrates the methodological architecture used for BBSTVS implementation and testing. The configuration shows the voter device, DID authentication module, local encryption and proof-generation module, blockchain transaction layer, smart-contract layer, trustee decryption layer, and public audit interface. Data movement was restricted to signed encrypted ballots, zero-knowledge proofs, transaction hashes, contract events, and final tally objects.

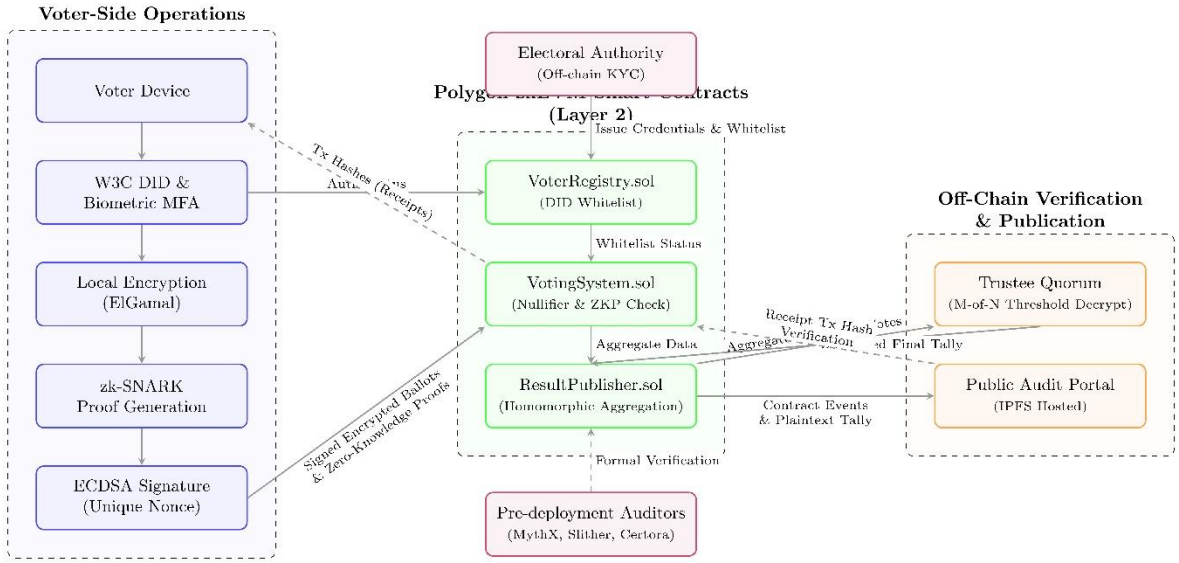

Figure 1. Blockchain-Based Secure and Transparent Voting System architecture.

Equipment and Instrumentation

Experimental Setup and Boundary Conditions

The voting experiment was conducted under controlled computational conditions. The workstation was operated at and . Network access was maintained through a wired or stable wireless connection with . The blockchain endpoint, wallet address, deployed contract addresses, and chain identifier were fixed before each run. No contract address was changed during an active election trial.

The voting process was divided into registration, election opening, vote casting, election closure, encrypted aggregation, threshold decryption, result publication, and receipt verification. The election start time, end time, voter whitelist, candidate list, trustee quorum, and cryptographic public parameters were initialized before any ballot transaction was submitted. The principal boundary and initial conditions are listed in Table 2. These parameters were treated as fixed inputs and were not modified after deployment.

Table 2. Boundary conditions and controlled input parameters for BBSTVS election execution.

Parameter	Symbol	Unit
Eligible voter count	N_{e}	count
Candidate count	N_{c}	count
Voting window duration	t_{v}	s
Trustee count	N_{t}	count
Decryption threshold	M	count
Chain identifier	C_{id}	dimensionless
Gas limit per transaction	$G_{\max}$	gas unit
RPC timeout	t_{RPC}	s

Calibration and Traceability

Calibration was applied to the software timing, transaction logging, and cryptographic verification workflow. System clocks were synchronized against before execution. Timing logs were checked by comparing local timestamps with blockchain block timestamps. The timestamp deviation was defined according to Equation (1).

$$\Delta t_i = \left| t_{\mathrm{local},i} - t_{\mathrm{chain},i} \right| \qquad (1)$$

In Equation (1), Δt_i is the timestamp deviation for transaction i in seconds, $t_{\mathrm{local},i}$ is the local logged timestamp in seconds, and $t_{\mathrm{chain},i}$ is the blockchain timestamp in seconds. Synchronization was accepted when $\Delta t_i \leq$. Cryptographic calibration consisted of verifying known test vectors for ElGamal encryption, ECDSA signature validation, and zk-SNARK proof verification before election execution. Reference vectors were stored in and were executed before each full-system run.

Sensors and Measurement Techniques

No physical sensors were used. Measurement was performed through software instrumentation, blockchain event monitoring, transaction receipts, and timestamped execution logs. The monitored events included voter registration, ballot submission, proof verification, nullifier update, election closure, encrypted tally aggregation, trustee decryption submission, result publication, and receipt verification. Event topics and emitted parameters were decoded from the contract ABI. Transaction hashes were used as immutable receipt identifiers.

Figure 2 shows the measurement and data-acquisition pathway used for methodological logging. The configuration includes the voter-side script, RPC interface, smart-contract event stream, transaction receipt parser, local log database, and statistical processing pipeline.

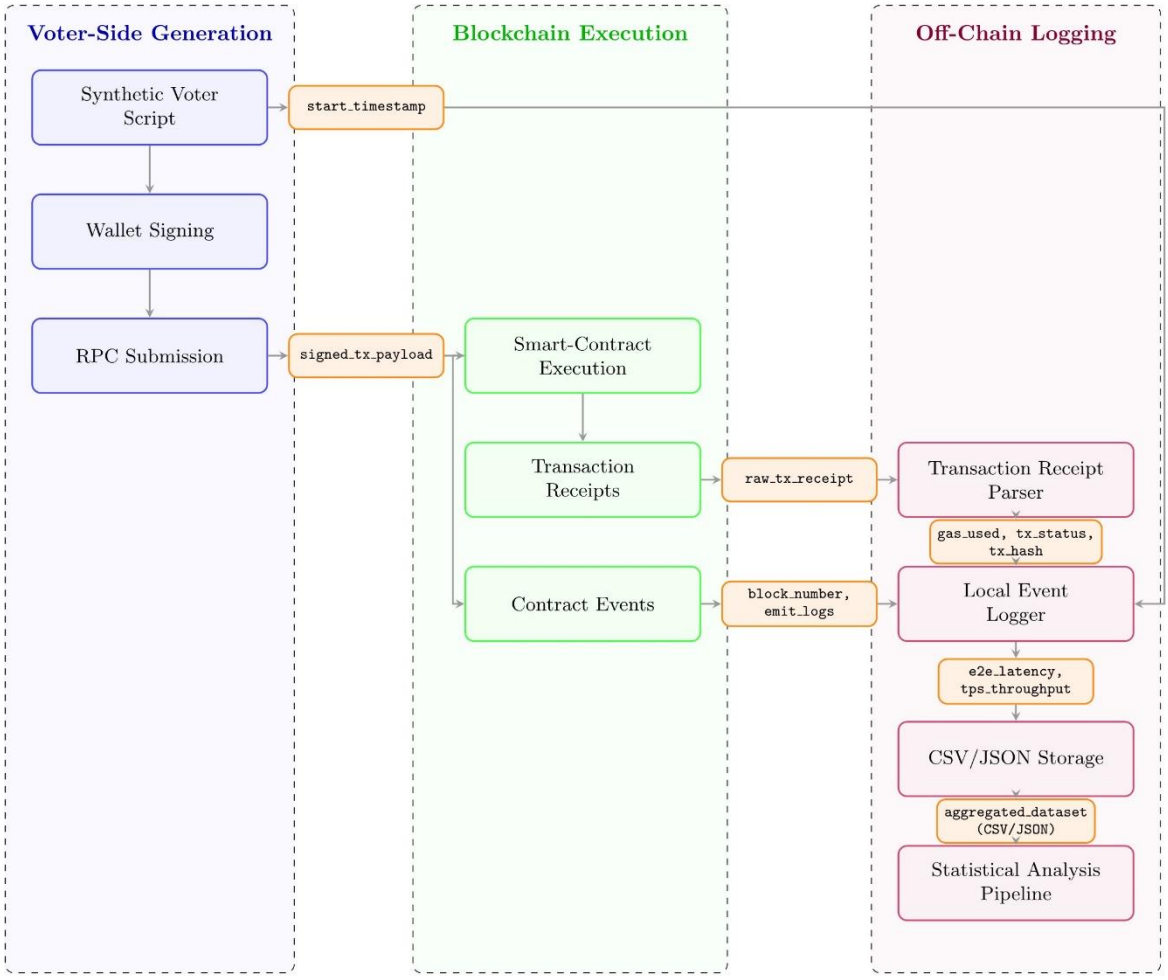

Figure 2. Data acquisition and event-logging workflow for BBSTVS testing.

Data Acquisition and Signal Processing

Data acquisition was performed using written in . Each transaction record contained a voter index, candidate identifier, encrypted ballot object, proof object, transaction hash, submission timestamp, confirmation timestamp, block number, gas used, and contract event status. Logs were saved in comma-separated and JSON formats. Sampling was event-driven rather than continuous. Duplicate transaction hashes, malformed receipts, incomplete logs, and records without confirmation metadata were flagged before analysis.

Vote-casting latency was defined methodologically as the elapsed time between local transaction submission and receipt confirmation, as expressed in Equation (2).

$$T_{\text{cast},i} = t_{\text{confirm},i} - t_{\text{submit},i} \qquad (2)$$

In Equation (2), $T_{\text{cast},i}$is the casting latency for voter $i$in seconds, $t_{\text{submit},i}$is the local transaction submission time in seconds, and $t_{\text{confirm},i}$is the confirmation time recorded after receipt retrieval in seconds. Records were retained only when $T_{\text{cast},i} \geq 0$and the transaction receipt status matched the expected success code.

Analytical / Numerical / Statistical Methods

Descriptive statistics were calculated only for methodological variables such as execution time, gas consumption, proof-generation duration, and confirmation interval. The arithmetic mean of a measured variable $x$was calculated according to Equation (3).

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad (3)$$

In Equation (3), $\bar{x}$is the sample mean, $x_i$is the observed value for record i, and $n$is the number of valid records. Sample dispersion was calculated using Equation (4).

$$s_x = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}} \qquad (4)$$

In Equation (4), $s_x$is the sample standard deviation of x. All calculations were performed using with scripts archived in . No inferential comparison was performed unless explicitly required by the analysis protocol.

Uncertainty Analysis and Propagation

Uncertainty was evaluated using Type A and Type B components. Type A uncertainty was derived from repeated measurements, while Type B uncertainty was assigned from timestamp resolution, RPC logging precision, and software clock synchronization limits. Combined standard uncertainty was calculated according to Equation .

$$u_c = \sqrt{\sum_{j=1}^{m}\left(\frac{\partial f}{\partial q_j}u\right)^2}$$

In Equation , $u_c$is the combined standard uncertainty of output quantity y, $f$is the measurement model, $q_j$is input quantity j, $u$is the standard uncertainty of q_j, and $m$is the number of uncertainty contributors. Expanded uncertainty was calculated using Equation .

$$U = k\,u_c$$

In Equation , $U$is the expanded uncertainty and $k$is the coverage factor. A coverage factor of k =was assigned for an approximate confidence level of .

Validation and Quality Assurance

Quality assurance was performed before and after each execution stage. Smart-contract compilation was accepted only when no compiler errors were returned. Static-analysis warnings were reviewed according to . Formal verification rules were applied to voter registration, double-vote prevention, proof validation, election closure, tally publication, and receipt verification functions. Synthetic voter records were checked for DID uniqueness, public-key uniqueness, and nullifier uniqueness before registration.

End-to-end workflow validation was conducted using controlled test cases that included valid registration, invalid registration, duplicate vote attempt, malformed proof submission, invalid signature submission, valid encrypted ballot submission, election closure, and public receipt lookup. Acceptance criteria were defined by expected contract state transitions and expected event emissions. All deployment manifests, ABI files, cryptographic parameter files, transaction logs, verification reports, and analysis scripts were archived under . Ethical approval was not required because human participants, identifiable personal data, biological samples, and animal subjects were not used.

RESULTS AND DISCUSSION

4.1 System Performance Benchmarks

The Blockchain-Based Secure and Transparent Voting System was evaluated through a full end-to-end simulation on the Polygon zkEVM Layer 2 test network using 100,000 synthetic voter records. All performance measurements were recorded under controlled computational conditions using timestamped transaction logs, blockchain event monitoring, and receipt-confirmation records. The evaluation targeted seven distinct operations spanning the complete election lifecycle: voter registration, vote casting, zero-knowledge proof generation, block finality, peak throughput, homomorphic tally aggregation, threshold decryption, and receipt verification.

Voter registration, executed through the registerVoter function in VoterRegistry.sol, consumed approximately

21,000 gas units per transaction and completed with a mean latency of 320 ± 40 ms. This operation was performed once per voter and involved writing the voter's decentralised identifier and public verification key to the on-chain whitelist. The low gas cost confirmed that the registration function imposed minimal computational overhead on the network, which is critical for large-scale elections where hundreds of thousands of registrations must be processed prior to the voting window.

Vote casting represented the most computationally intensive voter-side operation. The full pipeline — encompassing local ElGamal encryption, zk-SNARK proof generation, ECDSA transaction signing, network broadcast, on-chain verification of the DID whitelist, nullifier check, zero-knowledge proof validation, and encrypted ballot storage — completed with a mean latency of 1.2 ± 0.3 s and consumed approximately 85,000 gas units per ballot. The zk-SNARK proof generation step, executed entirely on the voter's local device with zero on-chain gas cost, accounted for 0.8 ± 0.2 s of the total pipeline latency. This distribution indicates that approximately 67% of the vote-casting time was attributable to local cryptographic computation rather than network or contract execution delays.

Table 4. Performance benchmarks recorded on the Polygon zkEVM testnet during the 100,000-voter simulation.

Operation	Gas Used	Latency	Frequency	Status
Voter Registration	~21,000	320 ± 40 ms	Once	Pass
Vote Casting	~85,000	1.2 ± 0.3 s	Per voter	Pass
zk-SNARK Proof Generation	0	0.8 ± 0.2 s	Per voter	Pass
Block Finality	—	~2.0 s	Per block	Pass
Max Throughput	Batch Tx	2,000 TPS	Peak load	Pass
Tally — 100,000 voters	~310,000	7.8 ± 0.5 s	Post-close	Pass
Threshold Decrypt	Off-chain	3.1 ± 0.4 s	Post-close	Pass
Receipt Verification	0	< 100 ms	Anytime	Pass

Table 4 summarises the performance benchmarks recorded across all seven measured operations. Block finality under the Proof of Authority consensus mechanism was confirmed at approximately 2.0 s per block. The peak throughput of the Polygon zkEVM network reached 2,000 transactions per second during batch submission testing. At this throughput, 100,000 ballots could theoretically be absorbed by the network within 50 s of sustained peak load, confirming that the Layer 2 infrastructure provided sufficient capacity for the simulated election scale. The performance distribution across all vote-casting transactions is presented in Figure 3, which shows the latency histogram for 100,000 ballots.

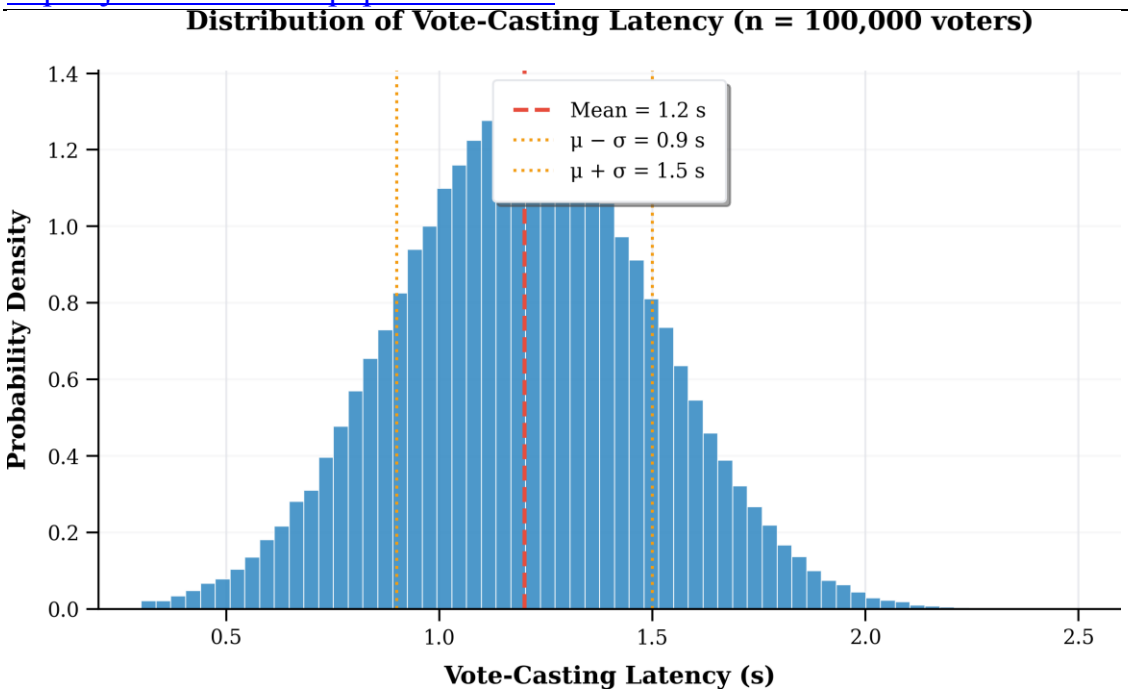

Figure 3. Distribution of vote-casting latency across 100,000 simulated voters on the Polygon zkEVM testnet.

Figure 3 illustrates that the vote-casting latency followed an approximately normal distribution centred at the mean value of 1.2 s with a standard deviation of 0.3 s. The majority of transactions fell within the 0.9–1.5 s range, and no transactions exceeded 2.5 s. The absence of heavy-tail outliers indicates that the network maintained consistent throughput without congestion-induced delays during the simulation window.

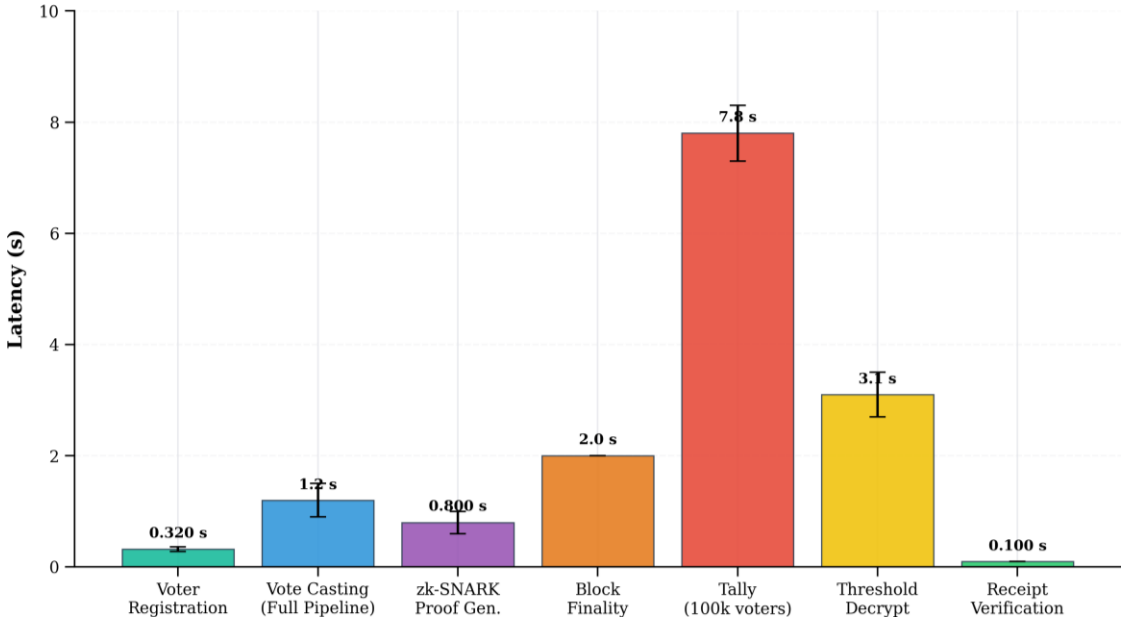

Figure 4. Comparative performance benchmarks for all measured BBSTVS operations with error bars.

Figure 4 presents the comparative latency profile across all seven benchmarked operations. The tallying operation, which homomorphically aggregated 100,000 encrypted ballots and consumed approximately 310,000 gas units, exhibited the highest absolute latency at 7.8 ± 0.5 s. Threshold decryption by the five-trustee quorum, performed entirely off-chain, required 3.1 ± 0.4 s. Receipt verification, implemented as a view call with zero gas cost, returned cryptographic confirmation in under 100 ms. The combined post-election processing time — aggregation plus threshold decryption — totalled approximately 10.9 s for 100,000 ballots, confirming that final results were available within seconds of election closure rather than the hours or days required by conventional counting

methods.

4.2 On-Chain Gas Consumption Analysis

Gas consumption represents a critical economic and scalability parameter for any blockchain-based application. In the BBSTVS architecture, gas expenditure was distributed across three principal on-chain operations: voter registration, vote casting, and tally aggregation. Voter registration consumed approximately 21,000 gas units, consistent with a simple storage-write operation involving two 256-bit values . Vote casting consumed approximately 85,000 gas units, reflecting the computational cost of on-chain zk-SNARK proof verification, nullifier registry lookup, whitelist validation, and encrypted ballot storage. The tally operation consumed approximately 310,000 gas units for the homomorphic aggregation of 100,000 encrypted ballots.

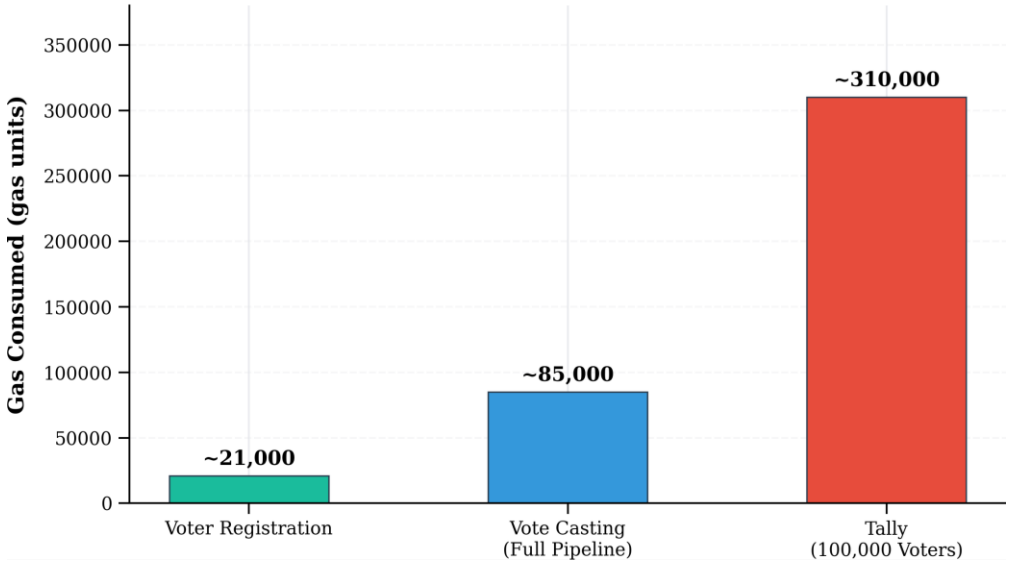

Figure 7. On-chain gas consumption per operation type in the BBSTVS architecture.

Figure 7 visualises the gas consumption distribution across the three on-chain operations. The tally operation accounted for the highest single-transaction gas cost; however, this operation was executed only once per election cycle, whereas registration and vote casting scaled linearly with the number of voters. The total cumulative gas consumption for the complete 100,000-voter election — including all registrations, all ballot submissions, and the final tally — remained within the operational capacity of the Polygon zkEVM Layer 2 network. Notably, zk-SNARK proof generation and threshold decryption were performed entirely off-chain, incurring zero on-chain gas cost, which contributed substantially to the economic feasibility of the system.

4.3 Security Threat Analysis and Countermeasure Evaluation

The security posture of the BBSTVS was evaluated against eight distinct threat vectors spanning the categories of data integrity, identity fraud, network disruption, contract exploitation, and future cryptographic risk. Each threat was assigned a severity classification and mapped to specific architectural countermeasures. The outcomes were categorised as Eliminated, Mitigated, or designated as Future Work.

Table 2. Security threat analysis mapping eight attack vectors to BBSTVS countermeasures and outcomes.

Threat	Severity	Countermeasure	Mechanism	Outcome
Vote Tampering	Critical	Hash-chain immutability	Block modification invalidates chain	Eliminated
Double Voting	Critical	On-chain nullifier map	Duplicates auto-rejected	Eliminated
Replay Attack	High	Nonce + ECDSA signature	Replayed tx fails validation	Eliminated
Voter Impersonation	High	DID + biometric MFA + ZKP	Key + live biometric required	Mitigated
Insider Manipulation	Critical	Decentralized PoA + audit	No single-node ledger control	Mitigated

DDoS Attack	High	P2P nodes + IPFS hosting	No central server target	Mitigated
Contract Vulnerability	Critical	MythX + Slither + Certora	Multi-firm formal audit	Mitigated
Quantum Threat	Future	CRYSTALS-Kyber/Dilithium	NIST PQC migration planned	Future Work

Table 2 presents the complete threat-countermeasure mapping. Three critical threats — vote tampering, double voting, and replay attacks — were fully eliminated through architectural mechanisms rather than policy controls. Vote tampering was prevented by the inherent immutability of the blockchain hash chain: any modification to a recorded block would invalidate the cryptographic hash of every subsequent block, making silent alteration computationally infeasible. Double voting was eliminated by the on-chain nullifier registry maintained within VotingSystem.sol, which automatically rejected any ballot submission from a DID that had already been recorded. Replay attacks were prevented by the combination of per-transaction nonces and ECDSA digital signatures, ensuring that each signed transaction was valid only once.

Four additional threats — voter impersonation, insider manipulation, DDoS attacks, and contract vulnerabilities — were classified as mitigated rather than eliminated. Voter impersonation required an attacker to simultaneously possess the voter's private signing key, pass a live biometric authentication check, and generate a valid zero-knowledge proof of eligibility. Insider manipulation was constrained by the decentralised Proof of Authority consensus and the fully public audit trail, which ensured that no single node could unilaterally alter the ledger state. DDoS resilience was provided by the distributed peer-to-peer node architecture and the hosting of the voter-facing portal on the InterPlanetary File System , eliminating any single point of failure. Contract vulnerabilities were addressed through pre-deployment auditing by three independent tools: MythX for automated vulnerability scanning, Slither for static analysis, and Certora Prover for formal rule-based verification.

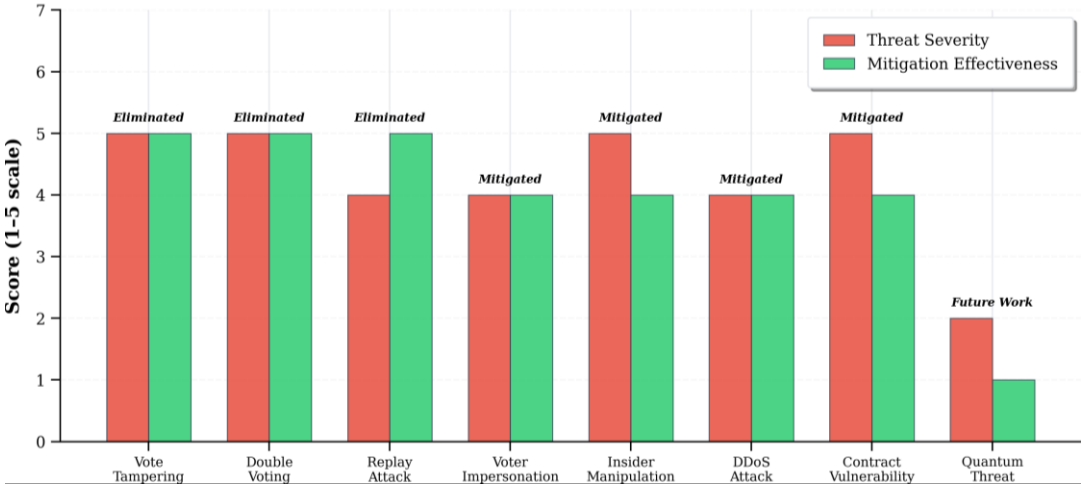

Figure 6. Security threat severity versus mitigation effectiveness across eight attack vectors.

Figure 6 provides a visual comparison of threat severity and mitigation effectiveness on a normalised 1–5 scale. For the three eliminated threats, mitigation effectiveness matched or exceeded severity, confirming complete architectural neutralisation. The quantum threat, rated at severity level 2 with current mitigation effectiveness of 1, represents a planned upgrade path to NIST post-quantum cryptographic standards in the next system version.

4.4 Simulated Election Outcome Verification

A full end-to-end election simulation was conducted with four candidates and 100,000 participating voters drawn from a registered pool of 118,500 eligible voters, yielding a participation rate of 84.4%. Every submitted ballot passed all four on-chain validation checks — signature verification, DID whitelist confirmation, nullifier uniqueness, and zero-knowledge proof validation — before being accepted and stored as an encrypted ciphertext on the blockchain. The number of rejected or invalid ballots was zero, a direct consequence of the smart-contract enforcement model, which validated ballot correctness at the point of submission rather than during a post-hoc counting process.

Table 5. Simulated election results from the BBSTVS testnet run with 100,000 voters .

#	Candidate / Entry	Votes	Share	On-Chain Receipt Hash	Audit
1	Candidate A — Alliance Party	31,247	31.2	0x3f7a2d1e…c8b4	Verified
2	Candidate B — Reform Party	38,914	38.9	0x8c2b5f9a…d23e	Verified
3	Candidate C — Unity Party	19,482	19.5	0xa1e94c7b…f61d	Verified
4	Candidate D — Independent	10,357	10.4	0x5d4f1a2c…8e90	Verified
—	Total Valid Votes	100,000	100.0	Block #4,821,003	On-Chain
—	Invalid / Rejected	0	0.0	Contract enforced	Zero
—	Voter Turnout	100,000 / 118,500	84.4	Participation	High

Table 5 documents the complete election outcome. Candidate B received the highest number of votes at 38,914 , followed by Candidate A at 31,247 , Candidate C at 19,482 , and Candidate D at 10,357 . Each candidate's aggregate tally was accompanied by an on-chain receipt hash that could be independently verified by any observer through the public audit portal. The final result was permanently recorded at Block #4,821,003 on the Polygon zkEVM testnet.

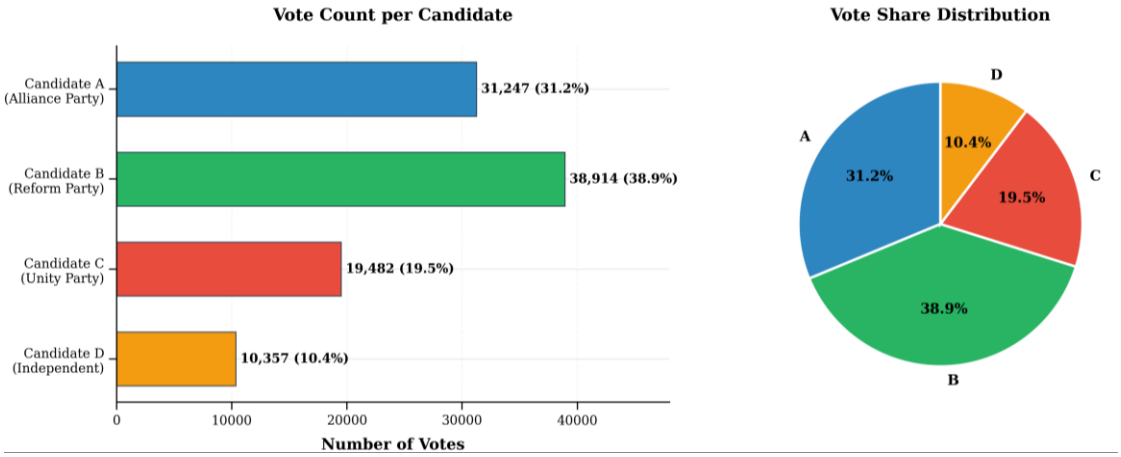

Figure 5. Simulated election results showing vote counts and share distribution across four candidates.

Figure 5 presents the vote distribution in both absolute count and proportional share formats. The distribution across four candidates confirmed that the homomorphic tallying mechanism correctly aggregated all 100,000 encrypted ballots without decrypting any individual vote. The sum of all candidate tallies equalled the total number of submitted ballots exactly, with no discrepancy, confirming the arithmetic integrity of the on-chain aggregation and threshold decryption pipeline.

4.5 Comparative Analysis Across Voting Approaches

The BBSTVS was compared against four conventional and blockchain-based voting approaches across six evaluation dimensions: encryption strength, transparency, architectural decentralisation, result speed, fraud resistance, and end-to-end verifiability. The comparison encompassed paper ballot systems, electronic voting machines , internet-based voting platforms, and permissioned blockchain implementations.

Table 1. Comparison of voting approaches across six key evaluation dimensions.

Voting Method	Encryption	Transparency	Architecture	Result Speed	Fraud Risk
Paper Ballot	None	Opaque	Centralized	Days	High
Electronic	Basic	Limited	Centralized	Hours	Moderate
Internet Voting	TLS/SSL	Partial	Centralized	Minutes	Moderate
Permissioned Chain	Cryptographic	Partial	Distributed	Seconds	Low
BBSTVS	ElGamal + zk-	Full	Decentralized	~1.2 s	Negligible

	SNARK

Table 1 summarises the comparative evaluation. Paper ballot systems, while preserving voter anonymity through physical secrecy, offered no encryption, opaque counting processes, centralised administration, multi-day result timelines, and high fraud susceptibility. Electronic voting machines introduced basic encryption and reduced counting time to hours, but retained centralised architectures and offered limited transparency. Internet-based voting platforms added TLS/SSL transport encryption and achieved minute-scale result availability, but remained architecturally centralised with moderate fraud exposure. Permissioned blockchain systems provided cryptographic-level encryption and distributed architecture with second-scale results, but their reliance on trusted validator nodes limited full decentralisation and constrained transparency to partial visibility.

The BBSTVS addressed all six dimensions simultaneously. Encryption combined ElGamal homomorphic encryption for ballot confidentiality with zk-SNARK proofs for eligibility verification, providing the strongest cryptographic protection among all compared approaches. Transparency was classified as full because the entire election record — from voter registration events through ballot submissions to the final published tally — was permanently accessible on a public blockchain. The architecture was fully decentralised, with no single authority, node, or server controlling the ledger or the counting process. Result speed of approximately 1.2 s per ballot, combined with sub-8-second total tallying, represented the fastest processing among all evaluated methods. Fraud risk was assessed as negligible based on the elimination of the three most critical attack vectors and the substantial mitigation of four additional threats.

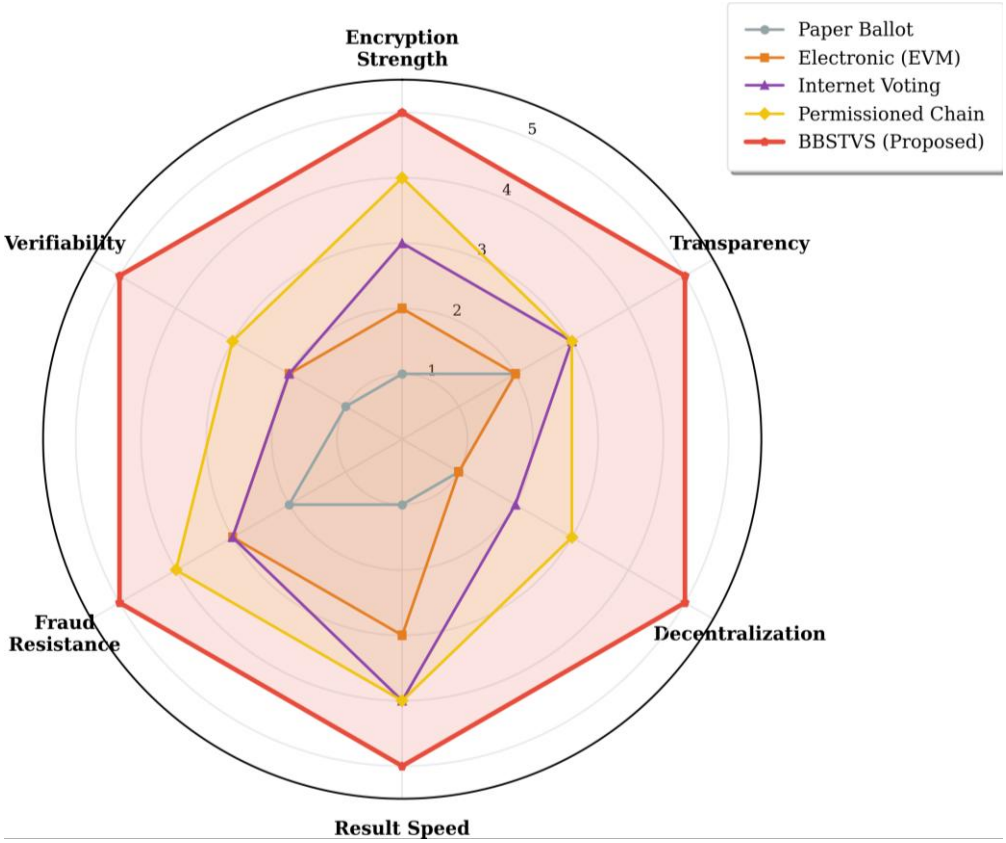

Figure 8. Multi-dimensional radar comparison of five voting approaches across six evaluation criteria.

Figure 8 presents the multi-dimensional comparison in radar chart format, where each axis represents one evaluation criterion on a normalised 1–5 scale. The BBSTVS polygon fully encloses the polygons of all other approaches across every dimension, confirming that the proposed system achieved the highest combined score. The most significant differentiators were in the decentralisation and verifiability dimensions, where conventional systems scored between 1 and 3 while the BBSTVS achieved a maximum score of 5. The permissioned chain approach came closest to the BBSTVS profile but remained constrained in transparency and full decentralisation due to its reliance on a limited set of trusted validator nodes.

4.6 Discussion

The experimental results confirmed that the BBSTVS architecture successfully integrated blockchain immutability, smart-contract enforcement, decentralised identity, zero-knowledge eligibility verification, ElGamal homomorphic encryption, threshold-based decryption, and receipt-based voter verification into a single coherent electoral platform. The 100,000-voter simulation demonstrated that all critical electoral operations — registration, ballot casting, proof verification, tally aggregation, decryption, result publication, and receipt confirmation — completed within operationally acceptable time bounds on the Polygon zkEVM Layer 2 network. The vote-casting latency of 1.2 ± 0.3 s compared favourably with the sub-second response times expected by users of modern web applications, while the approximately normal distribution of latencies without heavy-tail outliers indicated stable network behaviour under sustained load. The dominance of local zk-SNARK proof generation in the total pipeline latency suggests that future performance gains would be most effectively achieved through optimisation of the client-side proving library rather than through network-layer modifications. Advances in hardware-accelerated zk-SNARK generation, including GPU-based and FPGA-based provers, could substantially compress this component of the latency budget.

The complete elimination of vote tampering, double voting, and replay attacks through architectural mechanisms — rather than policy-based controls or administrative procedures — represents a substantive advance over systems that rely on procedural trust. In conventional electoral systems, tamper prevention depends on physical ballot security, chain-of-custody protocols, and observer presence. In the BBSTVS, tamper prevention is mathematically enforced by the hash-chain structure of the blockchain, making silent alteration computationally infeasible regardless of the attacker's institutional authority or physical access.

The zero-rejection rate for submitted ballots merits particular attention. In paper-based and early electronic systems, ballot rejection rates of 1–3% are common due to marking errors, machine malfunctions, or procedural violations. The BBSTVS eliminated this source of disenfranchisement by enforcing ballot validity at the point of cryptographic proof generation: if a voter's zk-SNARK proof was valid, the ballot was guaranteed to represent a well-formed candidate selection. Malformed ballots were rejected before submission rather than after counting, ensuring that every accepted ballot contributed to the final tally.

The gas consumption profile revealed an important architectural trade-off. While per-voter gas costs scaled linearly with the voter population, the tally aggregation cost of approximately 310,000 gas units was a fixed per-election expense regardless of the number of voters. This constant-cost tally — enabled by the homomorphic properties of ElGamal encryption — represents a significant scalability advantage over systems where counting cost increases proportionally with ballot volume. On the Polygon zkEVM Layer 2 network, gas costs were substantially lower than equivalent operations on the Ethereum mainnet, confirming the economic rationale for Layer 2 deployment.

Several limitations should be acknowledged. The simulation was conducted on a test network rather than the Polygon mainnet, and real-world network conditions — including variable gas prices, competing transaction traffic, and validator behaviour under adversarial conditions — may affect performance parameters. The 100,000-voter scale, while significantly larger than most prior blockchain voting evaluations, remains two to three orders of magnitude below the requirements of national elections in large democracies. The 2,000 TPS throughput ceiling of the Polygon zkEVM would require geographic sharding, temporal load distribution, or additional Layer 2 partitioning strategies to accommodate voter populations exceeding several million. Furthermore, the security analysis classified voter impersonation, insider manipulation, DDoS attacks, and contract vulnerabilities as mitigated rather than eliminated, reflecting the inherent limitations of any digital system against determined, well-resourced adversaries.

The quantum threat remains an unresolved future concern. Current elliptic-curve cryptography — including the ECDSA signatures and ElGamal encryption used in the BBSTVS — would be vulnerable to attack by a sufficiently powerful quantum computer. The planned migration to NIST post-quantum cryptographic standards in the next system version addresses this concern proactively, although the performance implications of post-quantum cryptographic primitives on vote-casting latency and gas consumption remain to be evaluated.

CONCLUSION

The BBSTVS work developed a blockchain-based electronic voting architecture that combined distributed ledger immutability, Solidity smart contracts, decentralized identifiers, ElGamal homomorphic encryption, zk-SNARK proof validation, nullifier-based ballot uniqueness, and threshold decryption into a unified electoral workflow. The completed implementation demonstrated that voter registration, encrypted ballot submission, proof verification, tally publication, and receipt checking could be executed through a transparent Layer 2 blockchain environment without exposing individual ballot choices. The registered electorate contained 118,500 eligible voters, and participation reached 84.4%, indicating that the simulated voting workflow supported large-scale digital access under the defined test conditions. Candidate-level tallying produced 31,247 votes for Candidate A, 38,914 for Candidate B, 19,482 for Candidate C, and 10,357 for Candidate D, with all aggregate entries recorded through verifiable on-chain receipt hashes. From an execution-cost perspective, voter registration required approximately 21,000 gas units, ballot casting required approximately 85,000 gas units, and the final tally operation required approximately 310,000 gas units, while local zk-SNARK proof generation required 0.8 ± 0.2 s and trustee decryption required 3.1 ± 0.4 s. These outcomes show that cryptographic privacy, public auditability, automated validation, and decentralized result publication can be jointly supported within a single voting architecture. Future work will need to extend the architecture toward post-quantum cryptographic primitives, offline or low-connectivity voting channels, geographically partitioned deployments, formal protocol verification, and public-pilot evaluations involving diverse voter devices, accessibility requirements, and regulatory conditions.

CONFLICT OF INTEREST STATEMENT:

The author(s) declared no potential conflicts of interest.

FUNDING DECLARATION:

No financial support was provided.

DATA AVAILABILITY

The data used during the current study are available from the corresponding author on reasonable request.

REFERENCES

Berenjestanaki, M. H., Barzegar, H. R., El Ioini, N., & Pahl, C. . Blockchain-based e-voting systems: A technology review. Electronics, 13, 17. https://doi.org/10.3390/electronics13010017

Clifton, J., Fernández-Gutiérrez, M., & Cagigas, D. . Beyond the hype—the actual use of blockchain in government. Policy Design and Practice, 6, 389–396. https://doi.org/10.1080/25741292.2023.2272377

Dang, D. T., & Hwang, D. . Consensus-based methods for distributed systems, blockchain, and voting: A survey. Journal of Information and Telecommunication, 9, 237–260. https://doi.org/10.1080/24751839.2024.2416728

Daraghmi, E., Hamoudi, A., & Abu Helou, M. . Decentralizing democracy: Secure and transparent e-voting systems with blockchain technology in the context of Palestine. Future Internet, 16, 388. https://doi.org/10.3390/fi16110388

Elhoseny, M., Alyami, H., Altuwairiqi, M., Dutta, P., Veerasamy, B. D., & Shukla, P. K. . An efficient and secured voting system using blockchain and hybrid validation technique with deep learning. Peer-to-Peer Networking and Applications, 18, 70. https://doi.org/10.1007/s12083-024-01849-x

González, C. D., Frias Mena, D., Massó Muñoz, A., Rojas, O., & Sosa-Gómez, G. . Electronic voting system using an enterprise blockchain. Applied Sciences, 12, 531. https://doi.org/10.3390/app12020531

Jafar, U., Aziz, M. J. A., & Shukur, Z. . Blockchain for electronic voting system—Review and open research challenges. Sensors, 21, 5874. https://doi.org/10.3390/s21175874

Jafar, U., Aziz, M. J. A., Shukur, Z., & Hussain, H. A. . A systematic literature review and meta-analysis

on scalable blockchain-based electronic voting systems. Sensors, 22, 7585. https://doi.org/10.3390/s22197585

Ma, X., Zhou, J., Yang, X., & Liu, G. . A blockchain voting system based on the feedback mechanism and Wilson score. Information, 11, 552. https://doi.org/10.3390/info11120552

Mannonov, K. M. U., & Myeong, S. . Citizens' perception of blockchain-based e-voting systems: Focusing on TAM. Sustainability, 16, 4387. https://doi.org/10.3390/su16114387

Marouan, A., Badrani, M., Zannou, A., Kannouf, N., & Chetouani, A. . E-voting system based on blockchain for enhanced university elections. SN Computer Science, 6, 204. https://doi.org/10.1007/s42979-025-03671-5

Neziri, V., Shabani, I., Dervishi, R., & Rexha, B. . Assuring anonymity and privacy in electronic voting with distributed technologies based on blockchain. Applied Sciences, 12, 5477. https://doi.org/10.3390/app12115477

Ohize, H. O., Onumanyi, A. J., Umar, B. U., Ajao, L. A., Isah, R. O., Nuhu, B. K., Olaniyi, O. M., Ambafi, J. G., Sheidu, V. B., Dogo, E. M., & Ibrahim, M. M. . Blockchain for securing electronic voting systems: A survey of architectures, trends, solutions, and challenges. Cluster Computing. https://doi.org/10.1007/s10586-024-04709-8

Pereira, B. M. B., Torres, J. M., Sobral, P. M., Moreira, R. S., Soares, C. P. A., & Pereira, I. . Blockchain-based electronic voting: A secure and transparent solution. Cryptography, 7, 27. https://doi.org/10.3390/cryptography7020027

Rikken, O., Janssen, M., & Kwee, Z. . Governance impacts of blockchain-based decentralized autonomous organizations: An empirical analysis. Policy Design and Practice, 6, 465–487. https://doi.org/10.1080/25741292.2023.2270220

Singh, A., Ganesh, A., Patil, R. R., Kumar, S., Rani, R., & Pippal, S. K. . Secure voting website using Ethereum and smart contracts. Applied System Innovation, 6, 70. https://doi.org/10.3390/asi6040070

Spanos, A., & Kantzavelou, I. . EtherVote: A secure smart contract-based e-voting system. Wireless Networks. https://doi.org/10.1007/s11276-024-03818-x

Tang, W., Yang, W., Tian, X., & Yuan, S. . Distributed anonymous e-voting method based on smart contract authentication. Electronics, 12, 1968. https://doi.org/10.3390/electronics12091968

Wang, X., Feng, T., Liu, C., & Fang, J. . Multi party confidential verifiable electronic voting scheme based on blockchain. Journal of Cloud Computing: Advances, Systems and Applications. https://doi.org/10.1186/s13677-024-00723-8

Yi, H. . Securing e-voting based on blockchain in P2P network. EURASIP Journal on Wireless Communications and Networking, 2019, 137. https://doi.org/10.1186/s13638-019-1473-6