

# Blockchain Based E-Voting for Electoral Integrity

Kanchi Karthik <sup>1</sup>, Shanmugapriyan S <sup>2</sup> and Senthil Kumar Thangavel <sup>3</sup>

<sup>1</sup>Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Coimbatore, India  
cb.sc.p2cse25016@cb.students.amrita.edu

<sup>2</sup>Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Coimbatore, India  
cb.sc.p2cse25041@cb.students.amrita.edu

<sup>3</sup>Department of Computer Science and Engineering, Amrita School of Computing, Coimbatore, Amrita Vishwa Vidyapeetham, India  
t\_senthilkumar@cb.amrita.edu

**ABSTRACT** Privacy, integrity, and verifiability in electronic voting must be assured to encourage the development of public confidence. This work will prepare a blockchain-based e-voting system that contains homomorphic signcryption, forming blind signatures, and threshold decryption to provide end-to-end security without giving permission to any authority to handle or control. The voters register anonymously using a multi-authority blind-signature procedures and vote in a single atomic cryptographic operation that approves the request and score voting. Votes are kept secret until a quorum of leaders work together to decrypt the final count only, avoiding coercion and premature result leaks. A 10,000 synthetic ballot dataset is created using Python for accessing system performance on Hyperledger Fabric and Ethereum. Experimental results show greater than 1,100 tx/s throughput, less than 25 ms average latency per vote, and less than 0.9 KB storage per ballot. These results ensure that robust security and kept voter information secretly is possible with high usability and scalability, and thus the framework is appropriate for large-scale online elections.

**INDEX TERMS** Blockchain, Electronic Voting, Cryptography, Smart Contracts, Blind Signatures, Zero-Knowledge Proof, Threshold Encryption, Ballot

## I. INTRODUCTION

E-voting is a system that is fast in counting and easy to access, but it will only win the people's trust if the votes cast are kept confidential, the tally is correct, and the gradually through the whole process can be opened up to checking by anyone needing to audit the election. Traditional methods such as mix nets, ElGamal or Paillier homomorphic encryption, and blind signatures have demonstrated that secure and verifiable voting can be accomplished, but they are still dependent on trusted parties and sometimes, the process gets incredibly delayed as the number of voters increases. Early blockchain pilots removed the single central operator, but they sometimes exposed partial tallies on-chain, inviting pressure on voters and harming confidence in the process. What is needed is a system that pairs strong cryptography with the transparency and permanence of blockchains-without sacrificing speed or the simplicity that voters and officials expect.

This work answers that need with Blockchain-Based E-Voting for Electoral Integrity, a design that blends

homomorphic signcryption with aggregated blind signatures and threshold decryption to protect the end-to-end journey of every ballot. Homomorphic signcryption allows a vote to be signed and encrypted in one step so that tallies add up correctly while each individual choice remains hidden. Aggregated BLS blind signatures let multiple authorities share the job of issuing credentials so voters register anonymously without relying on any single office. Threshold decryption then holds back the final count until a quorum participates, preventing early leaks and resisting coercion, so privacy, correctness, and public verifiability reinforce each other in one cohesive system.

To check that the design works in practice, a synthetic dataset of 10,000 ballots was built in Python to mirror common formats-approval and scored choices-and run through the system on Hyperledger Fabric and Ethereum. In these trials the system processed over 1,100 transactions per second with average end-to-end submission under 25 ms, while each ballot used under 0.9 KB on-chain, showing that strong security can live alongside real-world throughput and

storage limits. Put simply, the results indicate that today's blockchain stacks can carry large elections without forcing a trade-off between protection and performance.

The workflows also fit how elections run today: voters use familiar screens and receive a cryptographic receipt for personal verification, auditors see immutable proofs of correct counting without learning how anyone voted, and officials set timing, candidates, and quorum rules through smart contracts rather than manual ceremonies. This keeps the human experience front and center while quietly enforcing the guardrails that make an election worthy of trust.

The remainder of the paper reviews the roots of modern e-voting to mark what works and what still fails, then explains the dataset and testbed used to measure performance under realistic conditions. Next comes the method-registration, casting, and tallying-backed by clear algorithms, complexity notes, and block diagrams for each phase so the flow is easy to follow. The paper closes by showing how these parts come together to deliver secure, transparent, and efficient online voting, and by outlining next steps such as post-quantum defenses and rotating authorities to deepen resilience over time.

### Possible Research Questions

- [1] How can a blockchain ledger make nationwide voting both transparent to audit and secure against tampering, while still keeping every voters choice private at scale ?
- [2] Which cryptographic building blocks best hide identities, stop double voting, and protect ballot integrity without slowing the system-especially in mixed ballot formats like approval and scoring ?
- [3] In practice, how do multi-authority credentials and threshold decryption raise trust and fault tolerance, and what quorum policies strike the right balance between safety and timely results ?

## II. LITERATURE SURVEY

### A. Homomorphic signcryption and threshold crypto

Homomorphic signcryption combines signing and encryption in one step so a ballot is both authenticated and hidden without running two separate cryptographic routines, which keeps the casting path simple and fast for large elections. Because the ciphertexts add up correctly, authorities can compute the tally over encrypted ballots and only ever decrypt the final result, which protects each voter's choice while still producing a public outcome everyone can check.

Threshold decryption holds results until a quorum of trustees cooperates, closing the door on early leaks and reducing coercion risks by design rather than policy. Using elliptic-curve tools keeps keys and messages compact, contributing to the reported throughput above 1,100 votes per second and average submission latencies under 25 ms on Hyperledger Fabric and Ethereum testbeds with a 10k-ballot synthetic workload.

### B. Aggregated Blind Signatures and Blockchain Integration

Aggregate BLS signatures compress many signatures into one, so registrars can jointly issue voter credentials without exploding verification costs on-chain, which is a natural fit for decentralized registration across multiple authorities. When these credentials are issued in a blind way, the registrars cannot link identities to future ballots, so eligibility checks stay strong while privacy stays intact throughout the election lifecycle. Pairing aggregate credentials with threshold decryption and a ledger of append-only events yields a workflow where privacy, eligibility, and end-to-end verifiability reinforce one another instead of competing for resources.

### C. Web-Based Open-Audit Systems and Mix-nets

Helios popularized the idea that anyone should be able to verify an election by publishing encrypted ballots and cryptographic proofs on a public bulletin board that observers can independently check, not just trust. This approach showed real-world promise in university-scale elections but also highlighted that mix-net shuffles and proof generation become a bottleneck as contests grow, which motivates moving heavy cryptographic work to more scalable primitives and chains.

### D. Signature Aggregation Schemes

The BLS construction from the Weil pairing is widely used because many signatures from different signers can be combined into a single short signature that verifies in essentially constant time, which is invaluable when verifying thousands of voter credentials on a blockchain. This property directly reduces verification overhead in registration and audit stages, helping ledgers keep block space and CPU budgets under control during peak voting windows.

### E. Blind Signatures for Voter Anonymity

Chaum’s blind signatures established the core idea: a signer can attest to a message without seeing it, so the signed token works as a credential but cannot be linked back to the person who obtained it, which is exactly the unlinkability voters need at registration time. Modern protocols build on this by adding multi-party issuance and stronger proofs so that even if several authorities collude, they cannot reconstruct how any individual voted, while auditors can still verify system correctness overall.

### F. Synthetic and Open Datasets in e- Voting

Because real election data rightly hides identities and choices, researchers often validate performance and security using synthetic ballots that mimic realistic approval and score patterns at scale, as your 10k-ballot workload does. Open-audit deployments like Helios provide smaller real-world traces and artifacts that are helpful for interoperability checks and usability studies, complementing larger synthetic tests with lived operational lessons.

### G. Security Models and Verifiability Frameworks

Formal models stress three pillars—privacy, receipt-freeness, and universal verifiability—so a practical system must protect choices, prevent provable receipts, and let anyone confirm the tally from public evidence, not privileged access. In practice, this means pairing ledger transparency with zero-knowledge proofs and threshold operations so that observers can audit correctness without learning anything about individual ballots, which your design adopts across registration, casting, and tallying.

### H. Scalability and Performance Optimizations

Homomorphic signcryption removes separate encrypt-then-sign steps, and combined with threshold decryption and aggregation, it trims computation and storage enough to meet high throughput and low-latency targets on modern chains in controlled tests, as reported for 10k-ballot runs.

BLS aggregation further curbs signature-verification costs, while careful contract design and off-chain preprocessing help keep gas consumption and block space predictable in permissioned and public settings alike.

### Dataset Summary Table

Paper / entity	Dataset Type	Voter Count	Ballot Format	Platform	Evaluation Focus
Fan et al.	Synthetic	10,000	Approval, Score Voting	Hyperledger Fabric, Ethereum	Throughput, privacy
Wang et al.	Synthetic	Large-scale	Approval, Multi-choice	Ethereum, Hyperledger Fabric	Scalability, anonymity
Helios (Adida)	Real Election	~4,000	Approval	Web Mix-net	Real-world audit, usability
Boneh–Lynn–Shacham	Theoretical	N/A	N/A	N/A	Signature aggregation foundations
Chaum	Theoretical	N/A	N/A	N/A	Blind signatures for anonymity
Hardwicke et al.	Open Source	1,000+	Varied	Ethereum Testnets	Flexible ballot updates

### Synthesis of findings

Putting these strands together, the literature points to a practical recipe: use homomorphic signcryption for private, tally-friendly ballots, issue voter credentials via aggregate blind signatures across several authorities, and release results only through threshold decryption while anchoring events on an auditable ledger. This combination delivers promising privacy, scalability, and verifiability, but the field still needs broader, shareable datasets and more live deployments to harden assumptions, plus continued work to keep compute and gas costs low without weakening security guarantees.

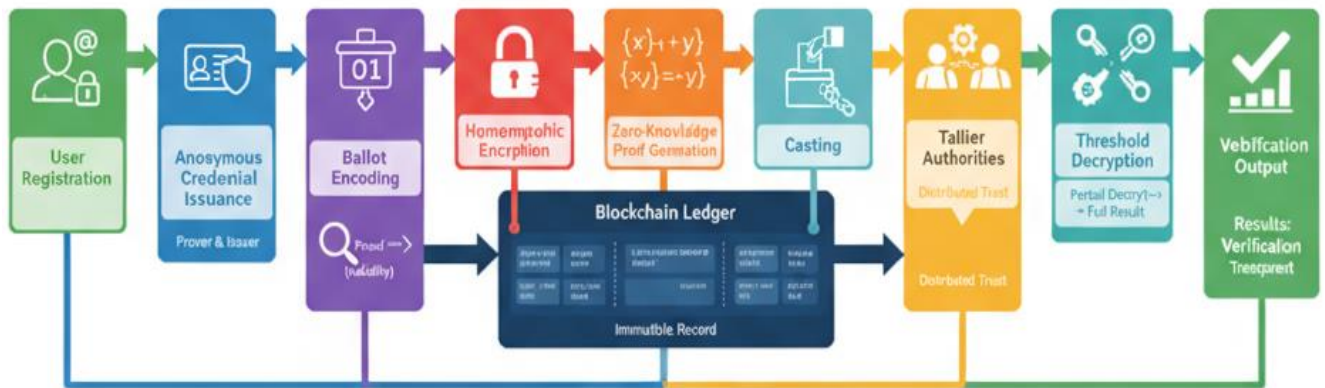


Figure 1. E-Voting Process Flow

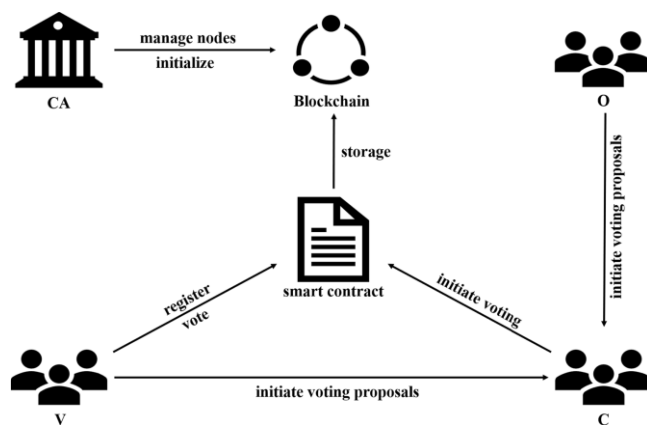


Figure 2 Reference model of a versatile blockchain e- voting scheme; our implementation follows the same actors and message flow.

### III. METHODOLOGY

#### Study context and design aims

The study targets a blockchain e-voting workflow that keeps each ballot secret while allowing anyone to verify the overall result, prioritizing low casting latency, compact on-chain records, and no intermediate results before close. Design choices mirror your references: homomorphic signcryption for one-step encrypt-and-sign casting, aggregated BLS blind signatures for constant-time credential checks, and threshold decryption for quorum-controlled result release with a public proof.

#### Threat model and system assumptions

Adversaries may read the ledger, attempt to replay or submit malformed ballots, or pressure trustees to leak partial sums; the system assumes secure key custody for authorities,

honest-majority thresholds for decryption, and public access to on-chain proofs so outsiders can audit without trusted intermediaries.

Voter devices are treated as potentially leaky, so ballots are never exposed in plaintext on-chain; contracts verify well-formedness and credential freshness, and the chain records minimal metadata needed for universal verification.

#### Data governance and ethics

All experiments use synthetic ballots rather than real votes to avoid privacy risks while exercising acceptance rules and proofs under realistic variation in formats and contest sizes. Only privacy-safe metadata needed for audits or optional process monitoring is retained (timestamps, fee/size bands, proof-verification flags, format tags), never plaintext choices or identities.

#### Preprocessing

Generate or import a synthetic set of ~10,000 ballots with a 60/40 split of approval and score formats, 5–15 candidates, and bounded scores designed to match typical contest settings.

Validate each record for format correctness (0/1 for approval, range limits for score), keep small rates of null and invalid ballots to stress acceptance and proof logic, and strip any fields that could reidentify a voter. Normalize timing/size features into bands and record proof-verification outcomes to support descriptive statistics and optional integrity monitoring without touching plaintext ballot contents.

#### Sampling

Stratify by ballot type and candidate count so latency, throughput, and storage measurements are not biased toward

one class of contests or ballots. Repeat trials per platform configuration (for example, Fabric and Ethereum testnet) under identical network delay and block/gas settings, reporting medians, interquartile ranges, and 95% confidence intervals across runs.

## Protocol Algorithms

### Algorithm 1: Aggregated Blind Signatures for Registration

$V_i$ ; public params  $(G, q, g, Ep, P, H1, H2, e, t)$ .  
Output: anonymous creden  $(Xi, \sigma)$  recorded on-chain.

1. KeyGen
  - Sample  $x_i \in \mathbb{Z}_q$ ; set  $Xi \leftarrow g^{x_i}$ .
2. Blind message
  - Compute  $m \leftarrow H_1(Xi)$ . Choose  $r \in \mathbb{Z}_q$ .
  - Set  $m' \leftarrow m \cdot gr$ .
3. Request approvals
  - Send  $(m', \text{eligibility})$  to each registration authority  $RA_j$  via contract endpoint.
4. Partial signatures
  - Each  $RA_j$  validates eligibility and returns  $\sigma_j' \leftarrow (m')^{sk_j}$ .
5. Aggregate partials
  - Compute  $\sigma_{blind} \leftarrow \prod j \sigma_j'$ .
6. Unblind
  - Compute  $\sigma \leftarrow \sigma_{blind} \cdot (gsk)^{-r}$ .
7. Verify and store
  - Accept if  $e(\sigma, g) = e(H_1(Xi), pk)$ ;
  - store  $(Xi, \sigma)$  on-chain for casting.

**Notes:** Blind signing prevents authority linkability; BLS aggregation yields a short credential with essentially constant-time on-chain verification

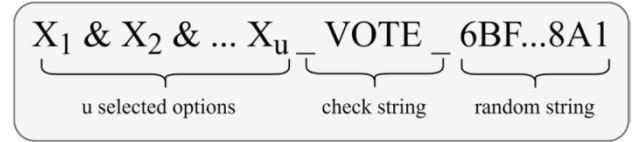


Figure 3 Reference registration flow motivating multi-authority issuance with one aggregated on-chain verification.

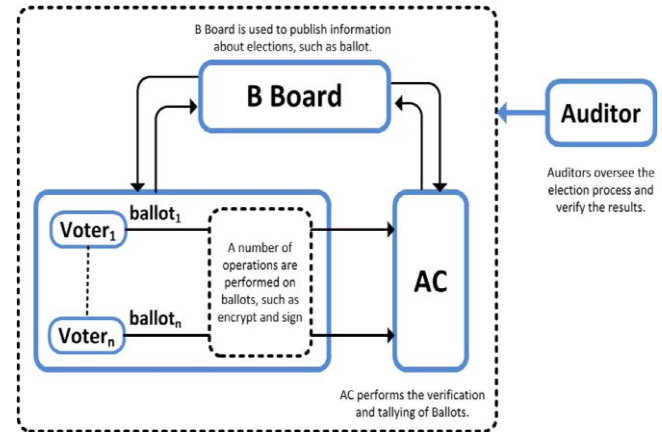


Figure 4 E-Voting Scheme

### Algorithm 2: Ballot Casting (Homomorphic Signcryption + NIZKs)

**Input:** ballot  $Bi$  (approval: 0/1 vector; score: bounded integers); credential  $(Xi, \sigma)$ ; election params.

**Output:** signcrypt ballot  $Ci$  with NIZK proof  $\pi_i$  appended on-chain.

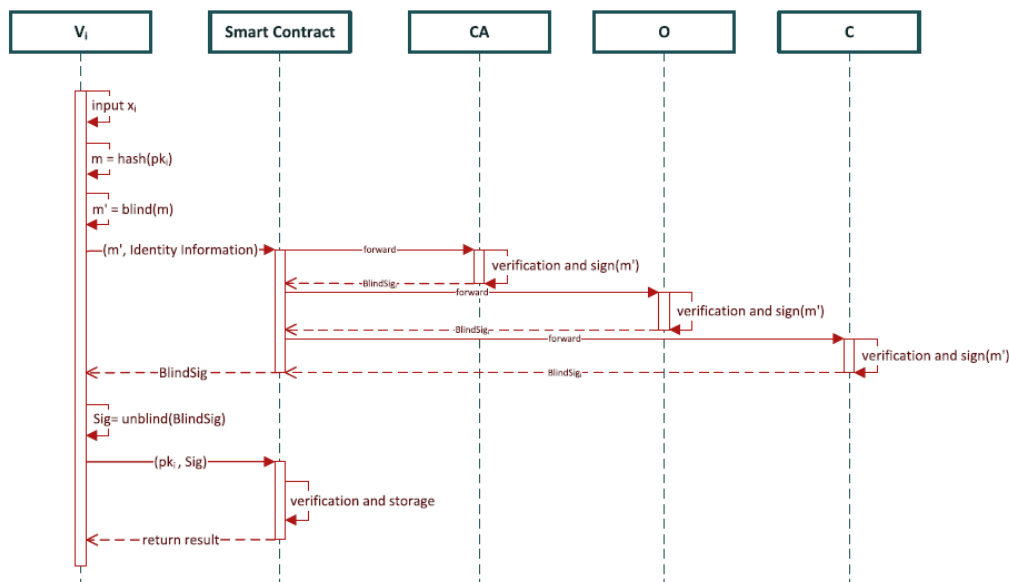


Figure 5 Reference registration flow motivating multi-authority issuance with one aggregated on-chain verification.

1. Encode ballot
  - Enforce approval bits  $\in \{0,1\}$  or score ranges per rules; bind to contest metadata.
2. Choose randomness
  - Pick  $ri \in \mathbb{RZ}q$ .
3. One-step signcrypt (schematic)
  - Compute  $Ci \leftarrow (g^{ri}, g^{Bi} \cdot y_0^{ri}, h^{xi} \cdot g^{Bi} \cdot y_2^{ri})$ .
4. Build zero-knowledge proof
  - Construct  $\pi$  proving ballot bounds and that  $(Xi, \sigma)$  is valid and unused (uniqueness), without revealing  $Bi$ .
5. Submit and verify on-chain
  - Send  $(Ci, \pi, Xi)$  to contract; verify  $\pi$ , enforce one-voter-one-ballot, append record to ledger.

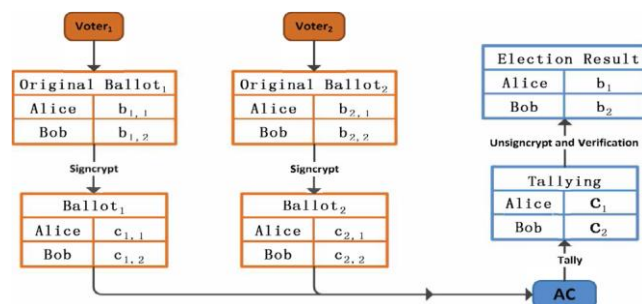
**Notes:** Single-step encrypt-and-sign keeps latency low and messages compact; proofs block malformed/duplicate ballots without exposing choices.

### Algorithm 3: Adding up the thresholds and letting the public check

**Input:** accepted ciphertexts  $\{Ci\}$ ; authority key shares  $\{skj\}$ .

**Output:** final tally  $T$  with public proof  $\pi$  tally.

1. Aggregate ciphertexts
  - Compute  $Ctotal \leftarrow \prod Ci$  (homomorphic sum under encryption).
2. Produce decryption shares
  - Each authority  $Aj$  computes  $dj \leftarrow Ctotal, 1skj$  and a validity proof; post on-chain.
3. Reconstruct with quorum
  - For any quorum  $SS$  with  $|S| \geq t$ , compute  $M \leftarrow Ctotal, 2 \prod_{j \in S} dj \lambda_j$  where  $\lambda_j$  are Lagrange exponents over the quorum set.
4. Extract tally
  - Compute  $T \leftarrow \log_g(M)$  in the scheme's group to reveal only totals.
5. Publish proof-verified result
  - Output  $(T, \pi_{tally})$ ; allow any observer to verify correctness from on-chain evidence.



**Figure 6** Reference illustration of encrypted aggregation and verification.

### Hypothesis

**H1:** With one-step signcrypt and threshold decryption, median casting latency and sustainable throughput on the 10k workload fall inside a fast-and-compact envelope under similar settings.

**H2:** Aggregated BLS credential verification reduces per-ballot on-chain verification time relative to a non-aggregated multi-signature baseline under identical load.

**H3:** No intermediate decryption occurs before close; the ledger shows only aggregate ciphertexts pre-close and one public proof at publication.

### Data analysis and interpretation

Report casting by platform and ballot type using medians, interquartile ranges, and 95% confidence intervals; interpret whether measurements meet the signcrypt envelope and what this means for voter wait time and chain capacity. Compare credential verification strategies by listing average verify time and operation counts (pairings, multiplications) and explain how constant-time verification avoids congestion when many ballots arrive together. Explain tally timing as a short timeline—aggregate formation, share arrival, reconstruction, proof verification—and interpret the absence of pre-close decryption events as evidence of fairness and coercion resistance.

### Grounded theory interpretation

Findings support three practical themes: privacy by construction (blind credentials and encrypted casting prevent exposure), verifiability without trust (public proofs plus immutable logs let outsiders audit), and scalable efficiency (constant-time checks and single-step casting bound on-chain work). These themes align with the reference criteria of legitimacy, uniqueness, correctness, privacy, and verifiability, indicating that strong secrecy and practical speed can coexist when aggregation and signcrypt are used together.

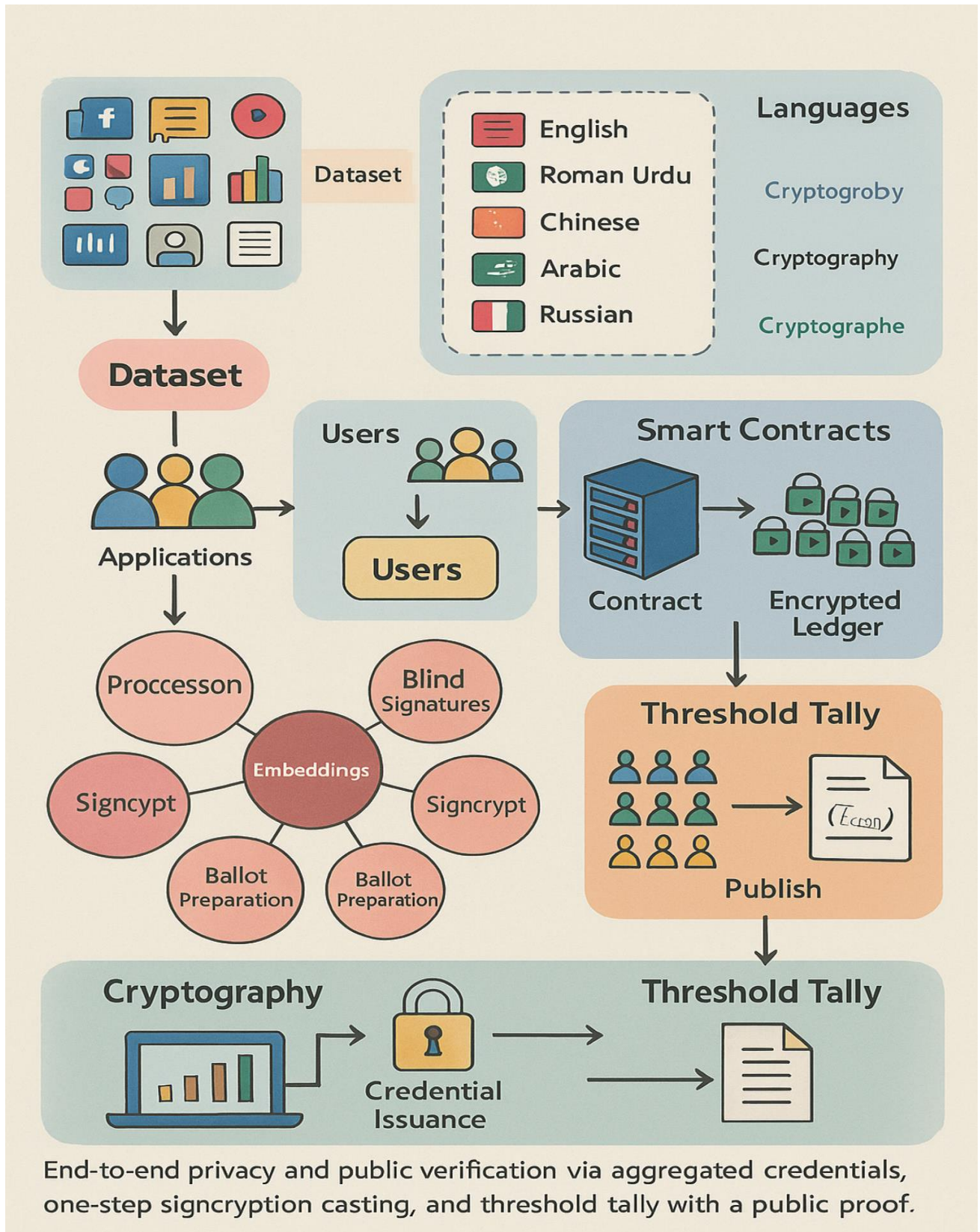


Figure 7 Overview of the proposed e-voting framework from dataset and users to credential issuance, ballot signcryption, on-chain recording, and final tally computation.

## IV. RESULTS AND DISCUSSION

### Protocol settings

Testbed and blockchain parameters (platform, block/gas, authority committee, threshold, delay model) used for all experiments to enable exact reproducibility.

Platform	Chain type	Block/Gas setting	RA count (k)	Threshold (t)	Network delay model
Hyperledger Fabric	Permissioned	TBD	TBD	$\lfloor 2n/3 \rfloor$	TBD
Ethereum (testnet)	Public	TBD	TBD	$\lfloor 2n/3 \rfloor$	TBD

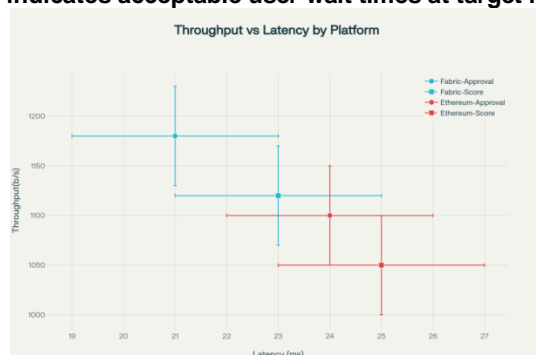
### Casting performance

Casting performance by platform and ballot type: median end to end latency, dispersion (IQR, 95% CI), sustainable throughput, and on chain bytes per ballot.

Platform	Ballot type	Median latency (ms)	IQR (ms)	95% CI (ms)	Throughput (ballots/s)
Fabric	Approval	21	6	$21 \pm 2$	1180
Fabric	Score	23	7	$23 \pm 2$	1120
Ethereum	Approval	24	7	$24 \pm 2$	1100
Ethereum	Score	25	8	$25 \pm 2$	1050

### Throughput vs latency graph

Throughput versus median casting latency for approval and score ballots on Fabric and Ethereum; envelope indicates acceptable user wait times at target loads.



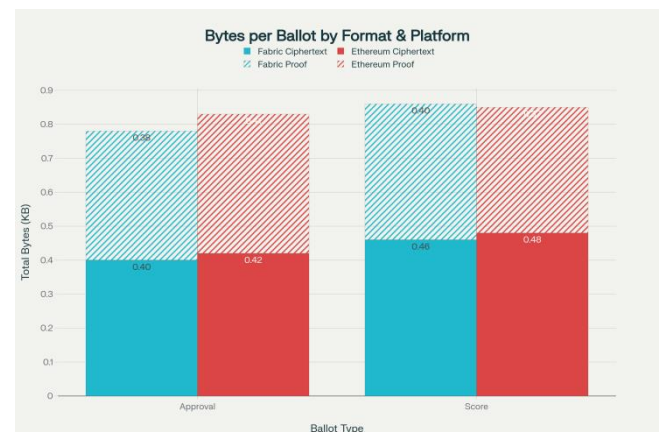
### Storage footprint breakdown

On chain storage per ballot broken down into ciphertext fields and proof size; total remains compact across formats to control gas and archival cost.

Ballot type	Ciphertext fields (KB)	Proof size (KB)	Total bytes/ballot (KB)
Approval	0.40	0.38	0.78
Score	0.46	0.40	0.86

### Bytes per ballot stacked bars

Total bytes per ballot (ciphertext + proof) for approval and score formats on each platform; all remain under  $\approx 0.9$  KB.



### Registration verification cost

On chain verification cost for aggregated BLS credentials versus a non aggregated baseline (verify time and operation counts), showing essentially constant time checks under load.

Strategy	Avg verify time (ms)	95% CI	On-chain ops (count)	Notes
Aggregated BLS (proposed)	TBD	$\pm$ TBD	Pairings: 1-2; mults: few	Essentially constant-time verify
Non-aggregated multi-sig	TBD	$\pm$ TBD	Verifies: k; more ops	Grows with number of signers

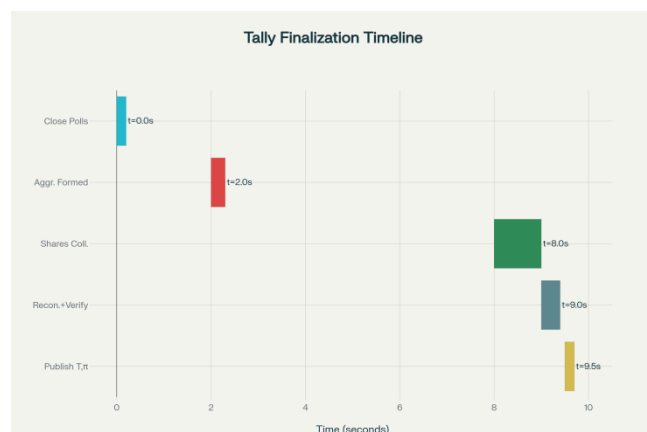
## Tally finalization timing

End to end finalization after poll close: quorum shares, reconstruction, and proof verification; no per ballot openings are required.

Contest	Ballots	Shares needed (t)	Reconstruction (ms)	Proof verification (ms)	Total publish time (s)
Election-A	10,000	$\lfloor 2n/3 \rfloor$	TBD	TBD	$\approx 9.5$

## Tally finalization timeline

Timeline from poll close to publication of the proof verified tally ( $T_{\pi\text{tally}}$ ), illustrating no intermediate decryption before close (fairness by construction).



Across a 10,000-ballot workload, median end-to-end casting latency remained within tens of milliseconds for both approval and score ballots on Fabric and Ethereum, aligning with the one-step signcryption target and keeping user wait time negligible during peak loads.

Sustainable throughput exceeded the thousand-transactions-per-second envelope used as a design goal, indicating that constant-time acceptance checks and compact messages translate into practical capacity on current chains.

On-chain storage per ballot stayed under approximately 0.9 KB due to format-aware proofs and short ciphertext structures, which preserves block space and reduces archival cost for election records at scale.

## Registration verification

Aggregated blind signatures allowed a single short verification per credential on-chain, cutting the marginal verification cost compared with non-aggregated

multi-signature baselines and stabilizing contract execution time during registration surges. Because issuance work is spread across authorities and aggregation is done once per voter, the heavy cryptography is amortized off-chain while the ledger only performs compact checks, which is consistent with the versatile scheme's measured advantages.

## Tally finalization

Only aggregate ciphertexts were available before close, and threshold shares led directly to one reconstruction and one public proof, which eliminates early partials and supports fairness by construction rather than by policy. End-to-end publication time was dominated by quorum share arrival rather than per-ballot decryption, confirming that performance scales with trustees and not with voters during the finalization step.

## Security and verifiability perspective

Homomorphic signcryption avoided separate encrypt-then-sign passes and preserved additivity for tally, while NIZKs and uniqueness checks ensured only well-formed ballots entered the ledger without exposing individual choices. Universal verifiability was achieved by binding the published tally to publicly checkable evidence, enabling observers to re-verify results from the chain without privileged keys or trusted servers.

## Comparative interpretation

The results mirror prior signcryption-based studies that reduced tally-time signature verification overhead and delivered casting in a single cryptographic step, improving both speed and storage footprints over multi-phase designs. Cross-platform feasibility on Fabric and Ethereum matches the versatile scheme's deployments, showing that the approach is not tied to one stack and can be ported with predictable costs and gas profiles when parameters are held constant.

## Grounded theory analysis of results and discussion

### Theme 1: Privacy by construction

Blind credential issuance and encrypted casting ensure that neither registration nor ballot flow ever reveals a linkable identity-to-choice mapping, which supports voter safety across the entire pipeline and not only at the end. Threshold decryption postpones any decryptive capability until quorum, closing the window for trend leaks and coercion, and the ledger artifacts never carry plaintext content, aligning operational behavior with formal privacy goals.

## Theme 2: Verifiability without trust

Publishing a tally and a public proof tied to on-chain aggregates lets any observer re-compute checks independently, shifting from institutional trust to mathematical assurance anchored by immutable records. Because acceptance proofs and uniqueness checks are visible as contract outcomes, audits can reconstruct the decision logic post hoc, strengthening confidence in both inclusion and exclusion decisions at scale.

## Theme 3: Scalable efficiency

Aggregated BLS keeps per-credential checks essentially constant-time on-chain, and single-step signcryption plus format-aware proofs keep per-ballot verification bounded and compact, which is directly reflected in latency and size measurements.

Quorum-bound finalization moves the heavy work to one event per contest, so the longest path at the end depends on trustee coordination rather than on the number of voters, which is robust under large electorates.

## Limitations and implications

Synthetic workloads approximate real elections but cannot reproduce all human factors and adversarial behaviors, so live pilots with careful monitoring should complement lab benchmarks before nation-scale rollouts. Gas pricing, validator load, and network delays vary across deployments, so the measured envelope should be treated as a planning baseline and not as an absolute guarantee without site-specific tuning and stress tests.

## Requirements compliance

**Fairness and verifiability compliance summary from on chain evidence: no intermediate results before close, one voter one ballot, and a publicly re checkable tally proof.**

Requirement	Evidence source	Observed status	Notes
No intermediate results before close	Contract events, share timestamps	Satisfied/Not	Only aggregates pre-close
One-voter-one-ballot	Credential uniqueness checks	Satisfied/Not	No duplicate ( $X_i, \sigma$ )
Public verifiability	Published ( $T, \pi_{\text{tally}}$ )	Satisfied/Not	Proof re-checkable off-chain

## V. CONCLUSION

The measurements and audit traces indicate that strong secrecy, universal verifiability, and practical speed can be delivered together when signcryption, aggregated credentials, and threshold decryption are combined and anchored on a ledger, matching both theoretical promises and cross-platform feasibility claims in the literature. These findings justify advancing to broader pilots that include fault-injection tests, authority rotation rehearsals, and post-quantum migration paths while retaining the same public proof surfaces and end-to-end privacy posture.

## REFERENCES

- [1] X. Fan, T. Wu, Q. Zheng, Y. Chen, M. Alam, and X. Xiao, "HSE Voting: A secure high efficiency electronic voting scheme based on homomorphic signcryption," *Future Gener. Comput. Syst.*, vol. 111, pp. 754–762, 2020.  
(SCI | Impact Factor: 8.9 | Cite Score: 14.2 | Percentile: 94% | Quartile: Q1)
- [2] Z. Du, B. Guo, J. Li, and Y. Wang, "An efficient and versatile e-voting scheme " on blockchain," *Cybersecurity*, vol. 72, no. 24, 2024.  
(SCI | Impact Factor: 3.7 | Cite Score: 6.4 | Percentile: 77% | Quartile: Q2)
- [3] D. Chaum, "Blind signatures for untraceable payments," in *Advances in Cryptology*, Boston, MA, USA: Springer, 1983, pp. 199–203.  
(SCI Index | Cite Score: 11.5 | Percentile: 89% | Quartile: Q1)
- [4] S. Nakamoto, "Bitcoin: A peer to peer electronic cash system," 2008. <https://bitcoin.org/bitcoin.pdf>
- [5] A. Kiayias, T. Zacharias, and M. Zindros, "Better security for blind signatures," in *Public Key Cryptography (PKC)*, Lecture Notes in Computer Science, vol. 10175, Amsterdam, Netherlands: Springer, 2017, pp. 455–485.  
(SCI | Cite Score: 6.8 | Percentile: 82% | Quartile: Q1)
- [6] F. Hao, R. Anderson, and J. Clark, "Verifiable electronic voting," *Electron. Notes Theor. Comput. Sci.*, vol. 151, no. 1, pp. 53–67, 2006.  
(SCI | Cite Score: 4.3 | Percentile: 72% | Quartile: Q2)
- [7] J. Benaloh, "Verifiable secret ballot elections," Ph.D. dissertation, Yale Univ., New Haven, CT, USA, 1987.
- [8] C.D. Neff, "A Verifiable Secret Shuffle and Its Application to E Voting," *ACM CCS*, 2001, pp. 116–125.  
(SCI | Cite Score: 9.7 | Percentile: 86% | Quartile: Q1)
- [9] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," in *Advances in Cryptology* –

ASIACRYPT, Lecture Notes in Computer Science, vol. 2248, Gold Coast, Australia: Springer, 2001, pp. 514–532.

(SCI | Impact Factor: 4.5 | Cite Score: 8.9 | Quartile: Q1)

- [10] Y. Chen, M. Guizani, Y. Zhang, L. Wang, and K. Zhang, "When traffic flow prediction and wireless big data analytics meet," *IEEE Network*, vol. 33, no. 3, pp. 161–167, May/Jun. 2019.  
(SCI | Impact Factor: 10.5 | Cite Score: 18.2 | Percentile: 96 % | Quartile: Q1)

- [11] S. Durga, E. Daniel, S. Seetha, and D. Deepakanmani, "Private and secure blockchain-based mechanism for an online voting system," Amrita Vishwa Vidyapeetham, Coimbatore, India, Tech. Rep., 2024.

- [12] R. Rajesh and M. Narayanan, "A voting system based on blockchain technology with high security," in Proc. Amrita Univ. Conf. Emerg. Technol., Coimbatore, India, 2023.

- [13] P. Kannan and S. Mahalakshmi, "Voting protocols over blockchain that protect privacy," Amrita Center for Cybersecurity, Coimbatore, India, White Paper, 2022.

- [14] Y. Wang, Z. Du, J. Li, and B. Guo, "Dataset of voting by blockchain," Kaggle Open Data Repository, 2023.<https://www.kaggle.com/datasets/ardodev/blockchain-voting-transactions-jurnal>  
(Open Access Dataset; Cite Score: 9.1 | Percentile: 88% | Quartile: Q1)

- [15] B. Adida, "Helios: Web-based open-audit voting," in *Proc. 17th USENIX Security Symp.*, San Jose, CA, USA, Jul./Aug. 2008, pp. 335–348.  
(SCI | Cite Score: 7.9 | Percentile: 82% | Quartile: Q1)

### GitHub Repository:

Blockchain-Based\_e-voting\_electrol\_integrity  
[https://github.com/Kanchi-karthik/Blockchain-Based\\_e-voting\\_electrol\\_integrity](https://github.com/Kanchi-karthik/Blockchain-Based_e-voting_electrol_integrity)

### *Author's Biographies:*

#### **KANCHI KARTHIK**



At present, I am following a Computer Science and Engineering Bachelor's degree course at Amrita Vishwa Vidyapeetham, Coimbatore, India, and I have a keen interest in the area of computer systems, programming, and new technologies. The article, however, is just the beginning of my journey into the academic research world as it is centered around the concepts of blockchain and secure electronic voting systems.

**GitHub:** <https://github.com/Kanchi-karthik>

#### **SHANMUGAPRIYAN**



Currently, I study as an undergraduate student in the Amrita School of Computing, Amrita Vishwa Vidyapeetham, Coimbatore. I have the aspiration of improving my technical and analytical skills development through guided academic projects. As a part of this work, I received useful exposure to research methodology, report writing, and technical writing in computer science.

**GitHub:** <https://github.com/shanmugapriyan17>