Unconditional Individual Verifiability with Receipt Freeness via Post-Cast Isolation

Janis Erdmanis[0000−0002−8963−5963]

janiserdmanis@protonmail.org
June 24, 2025

Abstract. We introduce a trapdoorless tracker construction for electronic voting that fundamentally reimagines verifiability through information flow control. Unlike existing E2E verifiable systems where receipt-freeness compromises individual verifiability, our approach achieves both simultaneously by requiring only temporary isolation of the voting calculator between ballot casting and verification—when voters enter unique challenges to compute trackers for locating their votes on the public tally board. Our construction leverages perfectly hiding Pedersen commitments and a unique tracker challenge mechanism to simultaneously achieve unconditional individual verifiability, practical everlasting privacy, and receipt-freeness while relying only on standard cryptographic assumptions. When verification failures occur, our system provides transparent accountability by precisely identifying whether the voting calculator or voting device is responsible. The system maintains security even with partial compliance with isolation procedures and offers robust protection against various adversaries while requiring minimal trust assumptions.

Keywords: e-voting · E2E-V · individual verifiability · receipt-freeness · everlasting privacy · exponential mix · commitment shuffle

1 Introduction

Individual verifiability, a cornerstone of End-to-end (E2E) verifiable systems [1,8,14], ensures voters can independently verify their vote was correctly recorded and cast as intended without trusting election authorities or software. This poses a fundamental challenge for coercion resistance, where voters must be able to convincingly deceive adversaries who may monitor and direct their voting actions. The state-of-the-art systems that incorporate coercion resistance mechanisms often do so at the expense of individual verifiability, which becomes conditional on the honesty of some of the parties [8, 19].

Typically, remote electronic voting systems address coercion resistance through revoting mechanisms and decoy credentials [16]. These approaches assume an opportunity window during which voters can set up decoy credentials or update their votes without detection. However, this solution becomes problematic when considering client-side security threats. Malware on voter devices can exploit the

deception mechanism itself, enabling adversaries to cast unauthorised votes or manipulate existing ones [19].

Another emerging requirement for e-voting systems is everlasting privacy [13]—ensuring vote anonymity even if future cryptographic breakthroughs undermine today's security assumptions. Whether through the discovery of mathematical trapdoors, advances in cryptanalysis [15], or the development of cryptographically relevant quantum computers [3, 11, 21], systems relying solely on computational privacy face a fundamental vulnerability: future adversaries could potentially determine how each voter voted. This theoretical possibility already impacts elections today: voters may hesitate to express their true choices out of fear, while election authorities might use it as justification to withhold election evidence, creating opportunities for deception.

Various approaches for achieving everlasting privacy have emerged [13]. Some systems employ perfectly hiding commitment schemes to achieve both-verifiability and everlasting privacy. Others attempt to leverage anonymous channels [9], though these often prove impractical due to vulnerabilities to ballot box stuffing and metadata analysis by adversaries tracking bulletin board submissions [13]. Recent work has extended these concepts to everlasting receipt-freeness, particularly in tracker-based systems where even a computationally unbounded coercer cannot determine whether a voter's presented tracker is genuine or fake [18].

Despite these advances, deployment complexity remains a significant barrier to adoption. Systems that rely on threshold ceremonies require coordinating multiple independent parties to decrypt the final tally and assign trackers to voters—a complexity that disproportionately affects smaller, resource-limited communities. This was evident in Ontario's 2022 municipal elections, where municipalities predominantly selected simple black-box systems without individual verifiability, explicitly citing the operational complexity of E2E verifiable systems as the primary deterrent [10]. As Brunet and Essex demonstrate [2], these simpler deployments often resulted in complete dependence on vendors and introduced significant privacy concerns, with voters not remaining anonymous during the count.

Advances in cryptographic protocols have enabled new approaches that avoid the operational complexity of threshold ceremonies. Haenni & Spycher introduced a system that breaks down ceremonial commitments into individual, self-contained pseudonym braiding steps [12]. Subsequent developments in [7] have further simplified deployment by eliminating the need for a trusted bulletin board through history trees [4] that relies on voter devices to track bulletin board consistency without full replication. The resulting system enables anonymisation among entirely untrusted parties while supporting fully centralised deployment - a sharp contrast to systems dependent on threshold decryption ceremonies. Furthermore, the use of pseudonymous voting signatures achieves both participation privacy and public audit eligibility, addressing a significant limitation of approaches without published signatures that remain susceptible to ballot box stuffing [8].

In this paper, we propose a trapdoorless tracker construction that derives security from information flow control rather than from the security of cryptographic secrets. Our approach only relies on the voters who may isolate their voting calculator from outside interference between vote casting and verification, ensuring that each tracker is exclusively bound to a publicly verifiable, randomly derived voter's individual tracker challenge.

The system works as follows: Voters use a voting calculator to cast their ballots, which contain perfectly hiding commitments to both their vote and tracker preimages. After voting phase closes, a common seed is derived from all commitments, generating unique tracker challenges for each voter. By isolating their voting calculator and entering this challenge, voters can compute their personal tracker to locate their vote on the public tally board. To ensure receipt-freeness, voters can select any tracker from the tally board and configure their calculator to display it instead of their actual tracker when coerced.

Our system achieves unconditional individual verifiability without trusted authorities or threshold assumptions. When verification fails, our design precisely identifies whether the voting calculator or voting device is responsible, creating strong incentives for vendors to ensure device security. The system maintains its security properties even with partial compliance with isolation procedures while offering protection against various adversaries with minimal trust assumptions.

1.1 Related Work

Significant advances in addressing these challenges emerged with the Selene voting system [20], which introduced verifiable voting using trackers with deniability through voter-specific trapdoors. Selene achieves individual verifiability when this trapdoor remains secure and receipt-freeness through a deniable channel for tracker material (alpha term) delivery. However, despite its distributed architecture, the system ultimately relies on a single authority for receipt-freeness that combines partial alpha terms to prevent detection of a fake tracking number use.

The Hyperion system [5] extended this approach by creating a personalized bulletin board view for each voter, allowing them to safely select any tracker as a decoy without risking collision with coercer-controlled trackers. Recent developments [18] achieved both everlasting vote privacy and receipt-freeness through commitment consistent encryption, though at the cost of increased complexity from reverse mixnet operations and pairing-based cryptography.

The core limitation of Selene-family systems is their reliance on trapdoor secrecy, requiring trust in verifying devices to neither leak secrets nor generate them with predetermined randomness. This reliance creates challenges in dispute resolution when verification fails. Additionally, the presence of the trapdoor presents issues for everlasting privacy. It necessitates pairing-based cryptography for its implementation and requires trusting all parties with access to the secret bulletin board not to keep its contents locally [18].

Our post-cast isolation approach offers two key advantages: First, it weakens assumptions from continuous trapdoor secrecy to temporary information flow

control after vote casting. Second, it achieves practical everlasting privacy using standard cryptographic primitives without pairing-based cryptography. For receipt-freeness, our approach uses the voting calculator display itself as the deniable channel for tracker setup, rather than relying on network transmission. This achieves the same deniability benefits as Selene's approach to tracker selection and can be extended with Hyperion-style personalized bulletin board views to prevent selection of coercer-controlled trackers.

2 High Level Overview

We first present a high-level overview of our trapdoorless tracker system before formalizing the security framework in section 3 and give a details on the security mechanism in section 4. Our system creates a verifiable voting mechanism balancing transparency with privacy, with a tracker construction that derives security from information flow control using a verifiably random seed computed after voting phase closes, rather than relying on potentially leakable cryptographic secrets.

2.1 Key Parties and Roles

The voting system involves several distinct entities:

- **Voting Calculator**: A secure device that constructs and signs encrypted votes. It processes voter selections and computes trackers during verification.
- **Voting Device**: The voter's chosen interface for interacting with the voting calculator and bulletin board. Responsible for delivering votes to the bulletin board.
- **Registrar**: Manages voter registration and maintains the eligible voting calculator public keys list.
- **Bulletin Board**: Public ledger that collects vote envelopes and publishes vote commitments, tally board, and associated evidence.
- **Tallier**: Performs vote decryption and commitment shuffle operations to produce the final tally board.
- **Braiders (Optional)**: Independent external parties that anonymise voter pseudonyms through sequential exponentiation mixes before voting phase begins.

2.2 Information Flow Control and Voting Calculator Isolation

Our system's security relies on Information Flow Control (IFC), extending traditional IFC concepts to temporal, user-enforced restrictions that prevent devices from learning information that could compromise their intended operation. For voting calculators, IFC ensures each device can only compute trackers for its designated voter's vote by controlling external information access during the verification phase.

Definition (Information Flow Control): *Information Flow Control is a temporal, user-observable security policy that restricts device access to externally generated information, ensuring the device operates only on pre-existing data or direct user input during critical verification periods.*

Isolation can be achieved through hardware means (preferred) or software controls. Hardware isolation provides stronger security guarantees as it is bound by physical constraints, while software implementation requires additional trust in the computing environment. Voters must be able to independently verify that isolation is properly implemented through physical indicators (disconnected cables, Faraday cage closure) or software confirmation (network interface status, airplane mode indicators).

The voting calculator isolation creates a temporal security barrier. After casting their vote, the calculator knows its own voter's tracker preimage but cannot access other voters' verification challenges or any trackers that will appear on the tally board, which are only generated after voting phase closes. This ensures that when the voter enters their unique challenge, the isolated calculator can only compute its designated voter's tracker. It cannot show trackers from other voters because it doesn't know their challenges, and any attempt to guess would produce an invalid tracker not listed on the tally board.

This temporal isolation shifts security foundations from trapdoor secrecy assumption [5, 18, 20] to information flow control, requiring trust only in the voter's implementation of IFC[1], achieving unconditional individual verifiability.

2.3 System Operation

The voting system operates in four sequential phases that work together to ensure both verifiability and privacy:

The Setup Phase Each voter receives a voting calculator containing a secret private and registered public key. These public keys are anonymised through sequential braiding operations in an exponentiation mix, producing a new braided generator and a list of eligible pseudonyms. This anonymization serves as the first layer of privacy protection, ensuring votes cannot be traced back to individual voters.

Voting Phase During the voting phase, voters use their voting calculator and voting device to securely cast and record their votes as shown in fig. 1. The voting process begins when the voter retrieves a voting proposal with its inclusion proof from the bulletin board using their voting device. After reviewing the proposal, the voter makes their selection on the voting device, which forwards it to the voting calculator.

[1] We treat IFC correct implementation as part of the steps the voter takes to verify their vote. This is justified because during casting of the vote, the adversary does not know whether the voter will or will not isolate their voting calculator to verify their vote, hence risks inevitable detection.

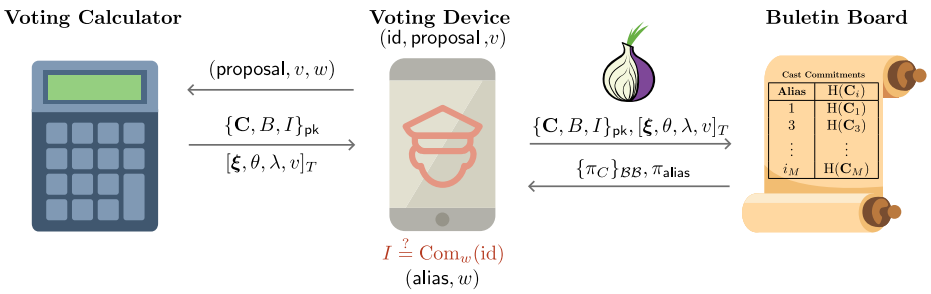

Fig. 1. Vote casting protocol. The voter makes their selection on their voting device, which coordinates with their voting calculator to construct an encrypted and signed vote which is submitted to the buletin board. Buletinboard publishes a hash of $\boldsymbol{C} = \text{Com}_{\boldsymbol{\xi}}(\theta, \lambda, v)$ during the voting phase and releases it at the E2E verifiability phase wheras encrypted openings $[\boldsymbol{\xi}, \theta, \lambda, v]_T$ are undisclosed to the public for the practical everlasting privacy. The voting calculator and device may communicate via NFC, while vote submission occurs through anonymous channels such as Tor [6].

The voting calculator then constructs an encrypted vote with two main components: a publicly verifiable part containing cryptographic commitments and an encrypted part containing opening values only the tallier can access:

$$\text{vote} = \{\boldsymbol{C}, B, I\}_{pk_V}, [\boldsymbol{\xi}, \theta, \lambda, v]_T \tag{1}$$

- $\{..\}_{pk_V}$ represents a signature using the voter's pseudonymous public key
- $[..]_T$ represents encryption that only the tallier can decrypt
- $\boldsymbol{C} = \text{Com}_{\boldsymbol{\xi}}(\theta, \lambda, v)$ are tracker preimage (λ, θ) and voters selection v commitments
- $B = \text{Hash}([\boldsymbol{\xi}, \theta, \lambda, v]_T)$ binds the encrypted openings to prevent tampering
- $I = \text{Com}_w(\text{id})$ is an identity commitment to the voter's identifier id with blinding factor w that binds the voter's identity to vote commitment $\boldsymbol{C}$.

The voting calculator constructs and signs the vote with a pseudonym and returns it to the voting device, which verifies the signature and submits it to the bulletin board through an anonymous channel. Upon successful submission, the bulletin board publishes the hash of vote commitment $\boldsymbol{C}$ and returns proof of inclusion to the voting device.

At this point, a critical security step occurs: the voter isolates their voting calculator using a kill switch or Faraday cage. This isolation is essential for individual verifiability as it prevents the calculator from learning other voters' tracker challenges or trackers after the tally board is announced.

Tally Board Announcement After voting phase closes, the tallier processes the collected votes to create a public tally board. This phase bridges the gap between individual ballots and the final election results while maintaining the security properties of the system.

The tallier first decrypts the encrypted openings in secret received from the bulletin board and verifies consistency with the published vote commitments. If inconsistent entry is found the tallier publishes associated $[\boldsymbol{\xi}, \theta, \lambda, v]_T$ that is binding to a public B and verifiably decrypts it as justification for it's omission. Next, the tallier computes a common seed from all vote commitments[2]:

$$\text{seed} = \text{Hash}(\{Q_i, R_i\}_i) \tag{2}$$

This seed is crucial for unpredictability of trackers during the voting phase, which supports unconditional individual verifiability as detailed in section 5. From this seed, the tallier derives verifiably random tracker challenges for each pseudonym. These individual challenges ensure that a compromised voting calculator cannot compute trackers for other voters' votes, supporting unconditional individual verifiability.

The system publishes a tally board containing all votes with corresponding tracking numbers. At this stage, voters can configure their voting calculators to display an override tracking number by selecting any tracker from the published tally board that way achieving receipt freeness if coercer asks to show their voting calcualtor after the voting phase closes.

E2E Verifiability Phase The final phase enables voters to verify their votes were correctly recorded and counted, completing the end-to-end verifiability of the system.

The tallier releases individual tracker challenges and all supporting integrity evidence. Voters retrieve their tracker challenge $e \in \{0,1\}^*$ from the bulletin board along with identity commitment proving it's exclusive ownership and enters it into their isolated voting calculator as shown in fig. 2. The voting calculator computes the tracker via the formula:

$$t = e\,\theta + e^*\lambda \pmod q \qquad\qquad e^* = \text{Hash}(e) \tag{3}$$

The dual-component tracker preimage design prevents coercers from linking trackers computed with arbitrary challenges during voting phase to actual trackers after voting phase closes. The challenge-dependent construction without fixed points ensures trackers are unpredictable during voting and prevents corrupt calculators from computing trackers for other voters' votes.

[2] When the voter enters challenge e into the voting calculator, this constitutes a non-interactive sigma protocol proving knowledge of discrete logarithms (θ, λ) in commitments (Q, R). The voting calculator's response $t = e\,\theta + e^*\lambda \pmod q$ allows public verification via $T = Q^e R^{e^*} = \mathsf{h}^r \mathsf{g}^t$, where the blinding factor $r = e\,\alpha + e^*\beta \pmod q$ remains hidden. As generators g, h are fixed at the setup phase, the Fiat-Shamir transformation only needs commitment pairs $\{Q_i, R_i\}_i$ as input to generate verifiably random challenges, ensuring protocol soundness while preventing challenge manipulation by any single party. The completeness of all commitment pairs in the seed is necessary to ensure no tracker t_i can be computed before the voting phase closes.

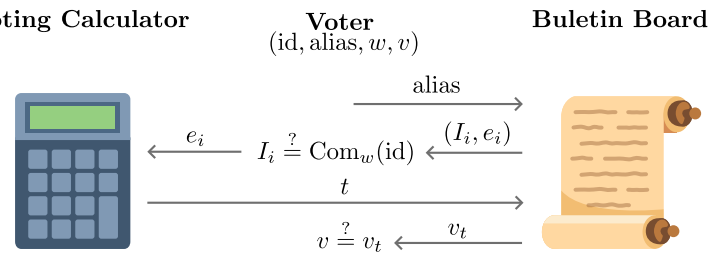

Fig. 2. Individual verifiability mechanism. Once voting phase closes and tracker challenges are announced, the voter uses their cast receipt's listed alias to retrieve their tracker challenge e_i and identity commitment I_i from the bulletin board. The alias is an index into the eligible pseudonym list, verifiable via Merkle inclusion proof π_{alias} from their cast receipt and root received with proposal. The identity commitment opening (w, id) proves the voter owns the pseudonym at that alias position. They then enter the tracker challenge e_i into their isolated voting calculator to compute a tracker t, with which they can locate their vote v_t on the tally board, confirming it was cast as intended. The protocol's security relies on the isolation of the voting calculator post-casting that prevents verification manipulation, while verifiably random tracker challenges with correct commitment construction proofs prevent tracker precomputation attacks. This is compatible with user experience, where the voting device generates a link encoding the verification parameters. When opened in a web browser, this link automatically establishes an anonymous channel to the bulletin board, verifies the identity commitment, and provides a form where voters can see their identity next to the tracker challenge and enter their computed tracker to retrieve their vote.

Using this tracker, voters can locate their vote on the tally board, confirming it was recorded and counted correctly. The complete discussion of this construction is provided in section 4. If a voter had previously set up a tracker override, the voting calculator will display the selected alternative tracker instead of the actual one, allowing voters to hide their true vote from potential coercers.

Conceptually, this design creates a cryptographic link between a voter's choice and a unique tracker that only becomes computable after voting phase closes. The voter's ability to verify their vote relies on information flow control through the isolation of their voting calculator after ballot submission, preventing access to trackers that could be exploited to deceive voters into verifying other voters' choices. This approach fundamentally differs from trapdoor-based systems, as security derives from information isolation rather than cryptographic secrets.

Even if voters fail to execute the isolation procedure perfectly, a corrupt voting calculator must risk detection, as it cannot know in advance which voters will properly isolate their devices. This creates a powerful deterrent against manipulation. When verification fails, our system enables clear attribution of blame to either the voting calculator or the voting device, providing accountability that incentivises honest behaviour from technology vendors.

3 Security Framework

3.1 Security Goals

Our trapdoorless tracker construction addresses the fundamental challenge of balancing transparency with privacy in electronic voting. We consider several essential security goals:

- *Individual Verifiability*: Voters can confirm that their votes were cast as intended and recorded as cast.
- *Universal Verifiability*: Independent observers can validate the integrity of the entire election process.
- *Receipt-Freeness*: Voters cannot prove to third parties how they voted.
- *Practical Everlasting Privacy*: Vote privacy is maintained even against computationally unbounded adversaries.
- *Accountability*: Clear procedures exist to identify the source of verification failures.

We shall not consider active coercers who observe voters during the voting phase or who attempt to execute forced abstention attacks. We consider coercion resistance to be a possible extension that can be built on top of this outlined system.

The security goals give rise to specific requirements that our tracker-based system needs to satisfy:

- *Tracker Unlinkability*: Trackers must be unlinked from voters in the public evidence (vote privacy)
- *Tracker Unpredictability*: It must be impossible to compute a valid tracker before the voting phase closes (individual verifiability)
- *Tracker Exclusivity*: The voting calculator must not be able to compute a tracker for another voter's vote (individual verifiability)
- *State Indistinguishability*: The system must prevent detection of tracker override setup from voting calculator outputs (receipt freeness)
- *Tamper Resistance*: The system must prevent extraction of tracker preimage parameters from voting calculator computations (receipt freeness)

3.2 Adversarial Model

Our security analysis considers several types of adversaries with different capabilities which can all collude simultaneously:

- **Coercer**: An entity that attempts to influence voters by demanding proof of specific voting behavior after the voting phase closes.
- **Corrupt Tallier**: An authority that may attempt to manipulate the tally or falsely claim that valid votes are invalid.
- **Corrupt Voting Calculator**: A device that may attempt to deceive voters about their tracker and show other voters tracker that had voted in the same way while silently casting manipulated vote.

– **External Observer**: An entity with access to the public bulletin board who attempts to link voters to their votes that may be computationally unbounded.

We shall not consider a corrupt registrar signing up fake members or preventing eligible members from getting the means to vote, as this lies within the scope of the digital identity infrastructure setup that needs to be present or can be bootstrapped[3]. The adversary model is not different from passport forgery or deniability of citizenship and can be verified with independent auditors that check a sample of members to be legitimate.

We shall also trust the bulletin board to include all correctly signed votes and not drop ones once recorded, which is a standard assumption among voting systems. In practice, this can be implemented with distributed collection servers replicating collected votes. Alternatively, the bulletin board can be implemented in a centralised setting with the help of history trees and voting devices that keep consistency-proof chains, which resolves bulletin board immutability [4, 7].

We also exclude from the adversarial model a voting calculator that uses other voters' pseudonyms to sign the votes. For sparsely cast votes, the voter can identify their pseudonym on the public buletin board by the time it was used to cast the vote; it can also set a sealing code or a voter's identity commitment that appears next to the signed vote as shown in fig. 1, fig. 2.

We trust the voting device for privacy, where voters input their vote choices and which is also responsible for delivering the votes over an anonymous channel to the bulletin board. The voting calculator is trusted for tamper resistance, so coercers can not distinguish whether voters try to deceive them with a fake tracker, ensuring receipt freeness.

We don't need to trust the voting calculator for vote anonymity as long as side-channel communication with vote submission is eliminated. This involves three steps:

– Voting device's randomness contribution when generating private key
– Sanitisation of witnesses used in the signatures via voting device randomness contributions
– Deriving tracker openings and blinding factors from a shared Diffie-Hellman secret.

All these points can be resolved with implementation details which allow the voting device to supervise the voting calculator for it to follow protocol honestly. That ensures that the it's constructed vote is not marked in any way that could be used to link the voter to the vote by a corrupt adversary who knows some secret.

3.3 Security Properties and Assumptions

Individual Verifiability The security model of individual verifiability relies on the voter's ability to isolate their voting calculator from communication after

[3] Identity commitments shown in figure 1, 2 can open to voters passport numbers that are widelly available.

Property	Calc	Device	Reg	BB	Tallier	Braiders
Recorded as Cast	🕵	🕵	🕵	🕵	🕵	🕵
Cast as Intended	🕵	🕵	🕵	🕵	🕵	🕵
Tallied as Recorded	🕵	🕵	🕵	🕵	🕵	🕵
Eligibility	🕵	🕵	🔒	🕵	🕵	🕵
Participation Privacy	🕵	🔒	🕵	🕵	🕵	🕵 one 🔒
Vote Anonymity	🕵 🔒	🔒 🕵*	🕵	🕵	🕵 🔒	🕵 one 🔒 🕵
Receipt Freeness	🔒	🕵*	🕵	🕵	🔒	🕵

Table 1. Security properties and trust assumptions. Each row shows which parties must be honest (marked with a lock) or can be corrupt (marked with a spy) while maintaining the listed E2E verifiability properties defined in [1]. The system achieves E2E verifiability even when all parties except the registrar are corrupt. For privacy properties marked with asterisk, security is maintained when voters are aware their device is compromised and can use an alternative device to cast their vote.

casting a vote until they verify their vote with a correct tracker challenge. Using the cast receipt, voters verify their pseudonym ownership and retrieve their unique tracker challenge from the public bulletin board. The voter enters the challenge into their isolated voting calculator to compute their tracker, allowing them to locate and verify their vote on the tally board.

Individual verifiability is unconditional as it prevents malware interference and doesn't depend on trusted hardware or third parties. This requires three key assumptions:

- The voting calculator is isolated from communication starting from once the vote is cast until voter enters their challenge to compute their tracker
- Tracker challenges are computed verifiably random after voting phase closes with all tracker preimage commitments fixed.
- Discrete logarithm hardness assumption

To illustrate the security mechanism, consider an adversary with full control over both the voting calculator and the voter's main device. Any attempt to create a superseding vote at casting would be revealed by the computed tracker. The adversary can not implant a tracker preimage to force a specific vote, as each voter receives a unique tracker challenge paired with their identity commitment.

Universal Verifiability The system's universal verifiability relies on zero-knowledge proofs bound by the discrete logarithm hardness assumption. Proofs verify correct tracker preimage commitment construction and random challenge generation, ensuring unpredictable trackers during voting. Tally board integrity employs Wikstrom's shuffle proof using vote and tracker commitment inputs.

Vote Anonymity The system ensures vote privacy through two complementary mechanisms. First, it employs braiding to anonymise voter pseudonyms via exponentiation mix before voting. This process can involve multiple independent parties sequentially without complex coordination, as braiding proofs immediately verify braider honesty. Pseudonymously signed votes require anonymous delivery, achievable through Tor [6] or physical communication networks that make tracking difficult due to brief transmission bursts.

Second, the system implements a commitment shuffle between vote commitments and the tally board. The Wikstrom commitment shuffle employs generalised Pedersen commitments to protect permutation secrecy. This provides practical everlasting vote privacy against future adversaries, assuming they can not access bulletin board-voting device and communication is confidential secured by post-quantum TLS initiated through anonymous channel.

Like other tracker-based systems, vote anonymity could be compromised during the verification step if an adversary observes which tracker a voter looks up. To prevent this, voters must retrieve their verification information from the bulletin board through anonymous channels or private information retrieval protocols [17].

Receipt Freeness The receipt freeness prevents voters from proving how they voted to a third party after the vote, even if they are willing to do so. The voting calculator that computes their tracker for individual verifiability can also serve as a receipt to the coercer. To prevent that a tracker override mechanism is used where voter can set up an override tracker that is displayed once a tracker challenge is typed in. The voters can pick a tracker from the published tally board. For this mechanism to work we hence need a following assumptions:

- Coercer can not observe internal state of the voting calculator
- Voter does not pick coercer's tracker from the tally board
- Bulletin board and tallier is not observed by the coercer
- Coercer can not identify specific votes through pattern matching in vote options (Italian attack)

To ensure the voting calculator's internal state remains hidden from coercers, the calculator must provide state indistinguishability. This property means that no time-bounded adversary can distinguish between different voting sessions or states by observing the calculator's responses, with a probability no better than random guessing. In the present tracker construction to protect from tracker preimage extraction that requires limiting tracker computation to one challenge.[4]

[4] This requirement can be relaxed by the voting calculator showing only the hash of the tracker as $t^* = \text{Hash}(t)$. However, this complicates the receipt-freeness mechanism as showing a fixed tracker value for all typed challenges would allow a coercer to detect the tracker override setup. To prevent that, authentication bits can be appended to the tracker challenge that can trigger override setup on the first correctly authenticated tracker challenge.

The tracker override setup is vulnerable to the fact that it can only be done after the tally board is announced. A coercer can use this fact to easily check its subjects that they had voted accordingly to their preferences by monitoring them until they verify their votes with their tracker challenges.

To prevent that, a coercion mitigation strategy is used by delaying tracker challenge release so that voters would have an extended opportunity window within which they can evade such coercer by selecting a preferable tracker from the tally board and setting up an override on the voting calculator.

Accountability In case the individual verifiability fails the responsible party needs to be identified to provide accountability. First the voter or its trusted party would check that the proofs published on the bulletin board are binding and the tally board represents the inputs. This excludes a corrupt tallier manipulating trackers. Then voter checks that its vote commitment signed by their owned pseudonym is present among the tally board shuffle inputs assuring that the voting device have properly delivered the vote as it assured after casting.

For cast-as-intended verification, disputes between the voting calculator and voting device are resolved transparently. When casting a vote, the device displays a numerical value, which should match the voter's selection. If this value matches what appears on the tally board but differs from the voter's intention, the voting device is proven at fault for incorrectly recording the selection. Conversely, if the value differs from the tally board display, the voting calculator is demonstrated to be the source of the discrepancy.

4 Security Mechanism

4.1 Cryptographic Primitives

The system employs a set of fundamental cryptographic primitives to achieve its security properties. We work in a cyclic group $\mathbb{G}_q$ of prime order q where the discrete logarithm problem is hard, with security parameter ω. Generators $\mathsf{g} \in \mathbb{G}_q$ and $\mathsf{h} \in \mathbb{G}_q$ are chosen so that discrete logarithm relationship is computationally hidden, ensuring the binding property of our commitment scheme. We use Pedersen commitments, defined as $\mathrm{Com}(x, r) = \mathsf{h}^r \mathsf{g}^x$ for a value x with randomness r. Our construction employs a cryptographic hash function $\mathrm{Hash} : \{0,1\}^* \to \mathbb{Z}_q$ modeled as a random oracle, and a pseudorandom generator $\mathrm{PRG} : \{0,1\}^* \to \mathbb{Z}_q^M$ that extends a seed to multiple tracker challenges.

4.2 Vote Construction

The voting calculator constructs a vote with a several interconnected components. The core of the vote is the vote commitment $V = \mathsf{h}^\gamma \mathsf{g}^v$, where v represents the voter's selection and γ is a blinding factor. This commitment conceals the vote while allowing for later verification. The vote also contains tracker preimage commitments $Q = \mathsf{h}^\alpha \mathsf{g}^\theta$ and $R = \mathsf{h}^\beta \mathsf{g}^\lambda$, where (θ, λ) is the tracker preimage, and

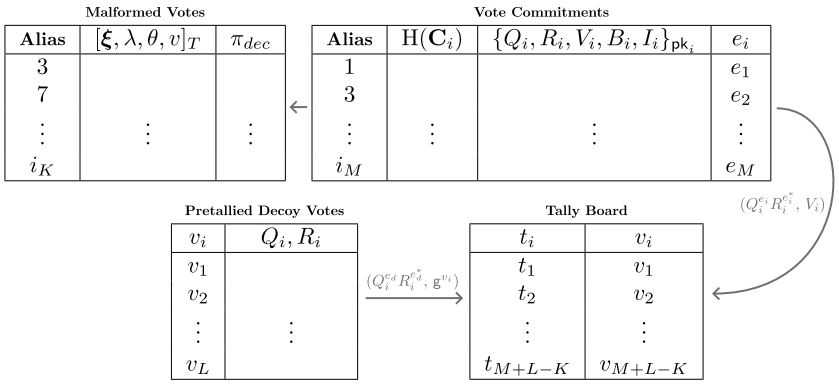

Fig. 3. Universal verifiability mechanism. During voting, only tracker preimage and vote commitments are published that are signed by eligible voters. Once the voting phase closes the tallier generates tracker challenges verifiably random e_i, the tracker commitments are constructed as $T_i = Q_i^{e_i} R_i^{e_i^*}$ where $e^* = \mathrm{Hash}(e)$ is a challenge conjugate and are passed through a commitment shuffle along with vote commitments for the tally board. The resulting trackers $t_i = e_i\theta_i + e_i^*\lambda_i \pmod q$ and votes are published as commitment openings (t_i, v_i) on the tally board. The tally board can be diluted with pretallied decoy votes using a common challenge e_d, allowing voters to compute safe decoy trackers.

(α, β) are corresponding blinding factors. The vote includes encrypted openings $[\boldsymbol{\xi}, \theta, \lambda, v]_T$, where $\boldsymbol{\xi} = (\alpha, \beta, \gamma)$ are blinding factors that are encrypted for the tallier as shown in eq. (1). The vote is signed with the voter's pseudonym as shown in fig. 1 and fig. 2 optionally also including an identity commitment.

4.3 Tracker Challenge Generation

After the voting phase concludes, the system generates individual tracker challenges verifiably random. A global seed is derived from all tracker preimage commitments by computing $\text{seed} = \mathrm{Hash}(\{Q_i, R_i\}_i)$. This seed serves as input to the pseudorandom generator to create individual challenges $\{e_i\}_i = \mathrm{PRG}(\text{seed})$. Each voter is then assigned a unique challenge e_i that will be used to compute their personal tracker. This approach ensures that challenges cannot be predetermined during the voting phase, which is crucial for unconditional individual verifiability.

4.4 Tracker Computation

When a voter receives their challenge e, their tracker is computed from tracker preimage (θ, λ) using the formula:

$$t = e\,\theta + e^*\lambda \pmod q \tag{4}$$

where $e^* = \text{Hash}(e)$ is a challenge conjugate that ensures the only fixed point for the voting calculator is the degenerate case $t = 0$ when tracker preimage $(\theta, \lambda) = (0, 0)$. This construction ensures both tracker unpredictability and exclusivity, binding each tracker uniquely to a voter's individual challenge—critical for preventing corrupt voting calculators from showing other voters' trackers. The dual-component design with challenge conjugate prevents coercion. Without the second component ($e^* = 0$), a coercer could ask a voter to compute a tracker with an arbitrary challenge e' during voting and verify compliance by recomputing the tracker with the actual challenge e later.

The tracker construction allows public computation of a tracker commitment from its preimage commitments as:

$$T = Q^e R^{e^*} = \mathsf{h}^r \mathsf{g}^t \tag{5}$$

where $r = e\,\alpha + e^*\beta \pmod q$ serves as a blinding factor for the tracker t.

4.5 Tally Board Construction

The final tally board is created through a verifiable shuffle process that anonymises votes while preserving vote-tracker pairings. Once tracker commitments T_i are computed for all votes, each vote commitment V_i is paired with its corresponding tracker commitment to form commitment pairs $\{T_i, V_i\}_i$. These pairs jointly undergo a verifiable commitment shuffle that permutes their order while maintaining the link between each tracker-vote pair.

This shuffle is performed using Wikström's protocol [23] with the following zero-knowledge proof:

$$\pi_{PoS} = \mathsf{PoK}\Big\{(\psi, r_i, \gamma_i) : T_i = \mathsf{h}^{r_i}\mathsf{g}^{t_{\psi(i)}} \wedge V_i = \mathsf{h}^{\gamma_i}\mathsf{g}^{v_{\psi(i)}}\Big\} \tag{6}$$

where $\psi : \mathbb{Z}_N \to \mathbb{Z}_N$ represents the shuffle permutation, $r_i = e_i\alpha_i + e_i^*\beta_i \pmod q$ is the blinding factor for tracker $t_{\psi(i)}$, and γ_i is the blinding factor for vote $v_{\psi(i)}$.

In practice, the shuffle is implemented using the Verificatum verifier specification [24], setting both generator and public key equal to h, reencryption exponents as $\{r_i, \gamma_i\}_i$, and ElGamal ciphertext components as $a = b = (T, V)$, ensuring perfect hiding of the shuffled elements.

After the shuffle, the system opens the shuffled commitment pairs to reveal the tally board entries $\{t_i, v_i\}_i$ while maintaining the permutation's secrecy. This ensures trackers remain unlinked from their originating voters. The final tally is computed by aggregating the revealed vote values according to the election rules.

4.6 Tracker Override for Receipt-Freeness

To enable receipt freeness, the voting calculator implements a tracker override mechanism. After the tally board is published, the voter selects a preferred tracker t' and enters it in the voting calculator to set t' as the override. When

the voter subsequently enters their tracker challenge e, the calculator displays t' instead of the actual tracker t.

This construction ensures that coercers cannot determine whether the voter is showing their true tracker or an override, as the internal state of the voting calculator remains hidden and produces identical output in both cases. The integrity of this mechanism relies on the tamper-resistance of the voting calculator and the voter's ability to set the override without coercer observation. Combined with the delayed release of challenges, this provides an effective mechanism for receipt-freeness in the presence of coercers who cannot physically monitor the voter during the entire verification process.

5 Security Analysis

We now analyze how our trapdoorless tracker construction defends against the adversary types defined in our threat model section 3.2. Our security arguments rest on three primary assumptions:

- *Discrete Logarithm Hardness (DLH)*: The computational infeasibility of solving the discrete logarithm problem in the chosen cryptographic group.
- *Random Oracle (RO)*: The computational infeasability of finding seed with a target tracker challenge.
- *Information Flow Control (IFC)*: The ability to isolate the voting calculator after vote casting until tracker verification, as detailed in section 2.

5.1 Against the Coercer

A coercer attempts to force voters to vote in a specific way and demands proof through tracker verification. Our defence relies on the tracker override mechanism described in section 4.6, with the following unique properties:

- *Tracker Extrapolation Protection*: A dual-component tracker design prevents extrapolation from trackers computed with arbitrary challenges during voting to actual challenges after voting phase closes;
- *Opportunity Window*: The delayed release of tracker challenges creates an opportunity window between tally board announcement and challenge distribution. During this window, voters can configure override trackers without coercer observation. The probability that a coercer detects the override setup is bounded by:
$$\Pr[\text{detection}] \leq \frac{\tau_{\text{setup}}}{\tau_{\text{window}}}$$
where τ_{setup} is the time needed to configure an override and τ_{window} is the total opportunity window duration.
- *Local Configuration Without Network Evidence*: Unlike systems requiring online interaction for receipt-freeness, our tracker override is configured locally on the isolated voting calculator, leaving no network traces that would reveal the override setup.

5.2 Against the Corrupt Tallier

The primary threat from a corrupt tallier is the potential manipulation of tracker assignments to compromise vote integrity. Our system's unique defenses include:

– *Tracker Unpredictability*: The tallier might attempt to force predictable tracker assignments by manipulating the last published tracker preimage commitments that enter the seed. Algorithm 1 illustrates why this is computationally infeasible:

Algorithm 1 Target Tracker Generation Attack

Input: $\{Q_i, R_i\}_{i \neq N}$, target tracker, voter index i
for $z \in \mathbb{Z}_q$ **do**
  $Q_N \leftarrow \mathsf{h}^{\alpha_N} \mathsf{g}^z$
  $R_N \leftarrow \mathsf{h}^{\beta_N}$
  seed $\leftarrow$ Hash($\{Q_i, R_i\}_i$)
  $\{e_j\}_j \leftarrow$ PRG(seed)
  $t_N \leftarrow e_N \cdot z \pmod q$
  if $t_N =$ target **then**
    return z
  end if
end for
return $\perp$

The attack demonstrates a corrupt tallier attempting to generate a target tracker which can be prebaked into voting calculators for a many-to-one attack, deceiving voters about their cast vote. The attacker can simplify the attack by setting $\lambda = 0$ and only varying $\theta = z$ to find the target tracker. The probability that a corrupt tallier successfully generates a target tracker in m attempts is approximately:

$$\Pr[\text{success}] \approx \frac{m}{2^{\min(|\text{seed}|, |q|)}} \tag{7}$$

Theorem 1: *Under the DLH and RO assumption, the probability that a corrupt tallier can manipulate the tracker assignment without detection is negligible in the security parameter ω.*

5.3 Against the Corrupt Voting Calculator

A corrupt voting calculator presents a unique threat to individual verifiability by potentially showing incorrect trackers. Our system counters this through:

– *Challenge Exclusivity*: The system uses individual challenges (as described in section 4) that are only paired with voter identities. This means a corrupt calculator cannot compute valid trackers for other voters' votes without knowing their challenges.

– *Tracker Range*: The probability that a corrupt calculator successfully guesses a valid tracker is bounded by:

$$\Pr[\text{guess}] \leq \frac{1}{q}$$

– *Cast-as-Intended Mutual Accountability*: During casting, the voting calculator displays a numerical vote value, establishing mutual accountability. If this value matches the tally board but contradicts the voter's intention, the voting device is at fault; if it differs from the tally board, the voting calculator is implicated.

Theorem 2: *Under the IFC, DLH and RO assumption the probability of a corrupt voting calculator presenting valid tracker for another voter's vote is negligible in the security parameter* ω.

5.4 Against External Observer

Our system also provides protection against the remaining adversary types from our threat model:

– *Pedersen Commitments*: Through the commitment shuffle described in section 4 and the use of perfectly hiding Pedersen commitments, the system ensures vote-privacy against both external observers with access to the bulletin board and computationally unbounded future adversaries. This provides information-theoretic protection of the vote-voter relationship, achieving everlasting privacy and receipt-freeness. Unlike systems relying on computational indistinguishability, our approach ensures that neither current observers nor future adversaries can link voters to their votes or determine whether a voter presented a real or fake tracker.

This comprehensive security approach ensures our trapdoorless tracker construction meets all the security goals defined in section 3 while minimizing trust assumptions compared to previous tracker-based systems.

6 Extentions

Short Trackers with Verifiable Randomness Our system can be enhanced with publicly verifiable randomness sources—such as physical entropy generators or distributed randomness protocols [22]—to produce tracker seeds that can only be determined after voting closes. This allows for shorter tracker challenges while maintaining security, with tracker length scaling proportionally to group size to mitigate collision probability. The tallier can employ a deterministic counter to resolve potential tracker collisions, creating a more efficient and practically deployable system without compromising security properties.

Individual Tally Board Views Building on Hyperion's approach, our system can be extended to create personalized tally board views for each voter, preventing coercers from detecting deception through controlled votes. Rather than using different tracker exponents, this implementation requires the tallier to create override tracker preimage commitments unique to each voter's view, with one real tracker commitment among decoys. Though this introduces quadratic tallying complexity, modern optimization of shuffle proofs makes it practical with high-performance hardware, significantly enhancing receipt-freeness guarantees.

Distributed Receipt Freeness The trapdoorless tracker construction can integrate with ElGamal reencryption mixnets by having vote commitments contain hashes of further ElGamal reencrypted ciphertexts for votes and tracker preimages. This approach distributes trust for receipt-freeness across multiple parties, where trackers from the first cascade act as challenges for the second ElGamal reencryption, creating a chained verification process. The link between tracker and individual challenge remains protected provided either the first tallier is honest or a threshold number of mixing authorities are honest and don't collude, effectively reducing reliance on a single honest authority while maintaining core security properties.

7 Conclusion

This paper presents a novel electronic voting system that achieves unconditional individual verifiability alongside receipt freeness through a post-cast isolation mechanism. The system demonstrates that these traditionally conflicting properties [19] can be combined effectively through careful protocol design and innovative security mechanisms, without requiring trust in election authorities, software, or hardware for basic verifiability.

The system's core innovation lies in its approach to vote verification and privacy. It employs verifiably random individual tracker challenges computed from a common seed computed only after a voting phase closes. Voters enter these challenges into their isolated voting calculators to compute trackers that locate their votes on the tally board. This mechanism ensures unconditional individual verifiability while enabling deniability through a tracker override mechanism, allowing voters to either select from existing votes or compute decoy vote trackers in coordination with tallier.

The system achieves practical everlasting privacy through perfectly hiding Pedersen commitments in the shuffle between cast records and the tally board. This protects against future adversaries who might break current cryptographic assumptions, while pseudonymously signed vote envelopes prevent ballot box stuffing and enable external monitors to enforce vote inclusion. Any attempts at vote manipulation or key extraction are publicly verifiable through forced decryption with an honest tallier, incentivizing vendors to ensure voting calculator security.

These innovations demonstrate the possibility of achieving unconditional individual verifiability alongside receipt freeness in practical electronic voting systems with centralized deployment model. By carefully balancing security properties with deployment considerations, the system provides a promising approach for real-world election scenarios where both verifiability and voter privacy are essential and is also suitable for smaller elections where deployment of threshold ceremonies is impractical.

References

1. Benaloh, J., Rivest, R.L., Ryan, P.Y.A., Stark, P.B., Teague, V., Vora, P.L.: End-to-end verifiability. ArXiv **abs/1504.03778** (2015), https://api.semanticscholar.org/CorpusID:14314175
2. Brunet, J., Essex, A.: Online voting in ontario municipalities: A standards-based review. In: Electronic Voting. pp. 52–68. Springer Nature Switzerland, Cham (2023)
3. Cai, J.Y.: Shor's algorithm does not factor large integers in the presence of noise. ArXiv **abs/2306.10072** (2023), https://api.semanticscholar.org/CorpusID:259203333
4. Crosby, S.A., Wallach, D.S.: Efficient data structures for tamper-evident logging. In: Proceedings of the 18th Conference on USENIX Security Symposium. p. 317–334. SSYM'09, USENIX Association, USA (2009)
5. Damodaran, A., Rastikian, S., Rønne, P.B., Ryan, P.Y.A.: Hyperion: Transparent end-to-end verifiable voting with coercion mitigation. IACR Cryptol. ePrint Arch. **2024**, 1182 (2024), https://api.semanticscholar.org/CorpusID:271488910
6. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The Second-Generation onion router. In: 13th USENIX Security Symposium (USENIX Security 04). USENIX Association, San Diego, CA (Aug 2004)
7. Erdmanis, J.: PeaceFounder: centralised E2E verifiable evoting via pseudonym braiding and history trees. Cryptology ePrint Archive, Paper 2024/1040 (2024), https://eprint.iacr.org/2024/1040
8. Finogina, T., Cucurull Juan, J., Costa, N.: Selective comparison of verifiable online voting systems. Security and Privacy (2024). https://doi.org/https://doi.org/10.1002/spy2.394, https://onlinelibrary.wiley.com/doi/abs/10.1002/spy2.394
9. Giustolisi, R., Garjan, M.S.: Efficient cleansing in coercion-resistant voting. In: Duenas-Cid, D., Roenne, P., Volkamer, M., Budurushi, J., Blom, M., Rodríguez-Pérez, A., Spycher-Krivonosova, I., Castellà Roca, J., Barrat Esteve, J. (eds.) Electronic Voting. pp. 72–88. Springer Nature Switzerland, Cham (2025)
10. Goodman, N., Spycher-Krivonosova, I., Essex, A., Brunet, J.: Verifiability experiences in ontario's 2022 online elections. In: Electronic Voting. pp. 87–105. Springer Nature Switzerland, Cham (2023)
11. Google: Quantum error correction below the surface code threshold. Nature (2024), https://api.semanticscholar.org/CorpusID:274608118
12. Haenni, R., Spycher, O.: Secure internet voting on limited devices with anonymized DSA public keys. In: Proceedings of the 2011 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections. p. 8. EVT/WOTE'11, USENIX Association, USA (2011)

13. Haines, T., Mosaheb, R., Müller, J., Pryvalov, I.: Sok: Secure e-voting with everlasting privacy. Proc. Priv. Enhancing Technol. **2023**, 279–293 (2023), https://api.semanticscholar.org/CorpusID:255921010
14. Jamroga, W., Ryan, P.Y.A., Schneider, S., Schürmann, C., Stark, P.B.: A declaration of software independence. In: Protocols, Strands, and Logic: Essays Dedicated to Joshua Guttman on the Occasion of his 66.66th Birthday, pp. 198–217. Lecture Notes in Computer Science, Springer International Publishing (2021). https://doi.org/10.1007/978-3-030-91631-2_11, https://doi.org/10.1007/978-3-030-91631-2_11
15. Joux, A.: A new index calculus algorithm with complexity $L(1/4 + o(1))$ in small characteristic. In: ACM Symposium on Applied Computing (2013), https://api.semanticscholar.org/CorpusID:3091762
16. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Workshop on Privacy in the Electronic Society (2005), https://api.semanticscholar.org/CorpusID:193733
17. Menon, S.J., Wu, D.J.: Spiral: Fast, high-rate single-server PIR via FHE composition. Cryptology ePrint Archive, Paper 2022/368 (2022), https://eprint.iacr.org/2022/368
18. Mosaheb, R., Rønne, P.B., Ryan, P.Y.A., Sarfaraz, S.: Direct and transparent voter verification with everlasting receipt-freeness. In: Duenas-Cid, D., Roenne, P., Volkamer, M., Budurushi, J., Blom, M., Rodríguez-Pérez, A., Spycher-Krivonosova, I., Castellà Roca, J., Barrat Esteve, J. (eds.) Electronic Voting. pp. 124–140. Springer Nature Switzerland, Cham (2025)
19. Rønne, P.B., Finogina, T., Herranz, J.: Expanding the toolbox: Coercion and vote-selling at vote-casting revisited. In: Duenas-Cid, D., Roenne, P., Volkamer, M., Budurushi, J., Blom, M., Rodríguez-Pérez, A., Spycher-Krivonosova, I., Castellà Roca, J., Barrat Esteve, J. (eds.) Electronic Voting. pp. 141–157. Springer Nature Switzerland, Cham (2025)
20. Ryan, P.Y.A., Rønne, P.B., Iovino, V.: Selene: Voting with transparent verifiability and coercion-mitigation. In: Clark, J., Meiklejohn, S., Ryan, P.Y., Wallach, D., Brenner, M., Rohloff, K. (eds.) Financial Cryptography and Data Security, vol. 9604, pp. 176–192. Springer Berlin Heidelberg (2016). https://doi.org/10.1007/978-3-662-53357-4_12, http://link.springer.com/10.1007/978-3-662-53357-4_12, series Title: Lecture Notes in Computer Science
21. Skosana, U., Tame, M.: Demonstration of shor's factoring algorithm for n=21 on IBM quantum processors. Scientific Reports **11**(1), 16599 (2021). https://doi.org/10.1038/s41598-021-95973-w, http://arxiv.org/abs/2103.13855
22. Syta, E., Jovanovic, P., Kogias, E.K., Gailly, N., Gasser, L., Khoffi, I., Fischer, M.J., Ford, B.: Scalable bias-resistant distributed randomness. In: 2017 IEEE Symposium on Security and Privacy (SP). pp. 444–460. IEEE (2017). https://doi.org/10.1109/SP.2017.45, http://ieeexplore.ieee.org/document/7958592/
23. Wikström, D.: A sender verifiable mix-net and a new proof of a shuffle. In: Roy, B. (ed.) Advances in Cryptology - ASIACRYPT 2005. pp. 273–292. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)
24. Wikstrom, D.: How to implement a stand-alone verifier for the verificatum mix-net. verificatum.org (2011)