Deniable Vote Updating with Eligibility Verifiability via Trapdoor Chains

Janis Erdmanis[0000−0002−8963−5963]

janiserdmanis@protonmail.org

Abstract. Remote e-voting systems that implement coercion resistance via deniable vote updating require a cleansing phase to discard superseded ballots before tallying. However, achieving a cleansing that is simultaneously publicly verifiable, robust to interrupted ballot chains, and efficient without leaking revoting patterns remains an open challenge. We present a publicly verifiable cleansing protocol that uses trapdoor chains to verifiably force the collector to select a single final ballot for each voter from a shuffled ballot list. As a result, each eligible voter who participates contributes exactly one final ballot without leaking information about superseded ballots. The protocol runs in linear time with a single, publicly verifiable cleansing authority and integrates with both mixnet and homomorphic tallying. The system supports revoting, fake credentials, and credential recovery via the trapdoor-chain mechanism. It further provides individual verifiability under signing key and trapdoor secrecy. This enables revoting-based coercion-resistant systems (such as the Estonian i-voting system) to achieve public cleansing verifiability without requiring trust in internal auditors.

Keywords: e-voting · E2E-V · eligibility verifiability · coercion resistance · deniable vote updating · commitment shuffle

1 Introduction

End-to-end verifiable (E2E-V) voting aims to provide voters and observers with cryptographic evidence that election outcomes reflect the recorded ballots and that only eligible voters cast at most one vote [1]. This property is widely viewed as essential for trustworthy remote electronic voting. However, verifiability creates a fundamental tension with *coercion resistance* [12]: the same mechanisms that let voters and observers verify can also reveal information that allows adversaries to confirm how a voter participated. Voters must therefore be able to update or disguise their choices under coercion without leaving a detectable trace. A related emerging concern is *everlasting privacy* [9]—the fear that future advances in computation may eventually break today's encryption. Although present evidence suggests such capabilities remain distant [6, 21], the concern is already cited to justify restricting public verifiability.

Coercion resistance is commonly realized through *revoting* and *fake credentials*. In revoting schemes, voters can repeatedly update their choices in private,

ensuring that any coerced vote can later be replaced without detection. In contrast, fake-credential approaches allow voters to generate ballots that appear valid but are ultimately excluded during tallying [2, 10, 12, 14]. However, fake credentials introduce usability challenges, as voters must distinguish genuine credentials from decoys. Alternative strategies modify the voting process—for example, by discounting duplicate votes [5] or allowing voters to override remote ballots with paper ballots cast at polling stations [8, 13]. Coercion-resistant systems must also address *forced abstention*, where an adversary attempts to ensure that a voter does not participate, for instance by verifying non-participation on a public bulletin board.

Beyond coercion, remote electronic voting systems must also prevent *impersonation*—unauthorized parties casting ballots on behalf of legitimate voters when authentication mechanisms are weak or credentials are compromised [18]. This requirement reinforces the need for strong eligibility verifiability, for which systems based on auditable hardware credentials—especially smartcards that digitally sign ballots—are widely considered the gold standard [19]. However, in revoting-based systems, this creates a fundamental tension during the *cleansing phase*, where a single ballot must be selected per voter without revealing which ballots were retained or discarded. Otherwise, revoting behavior becomes observable, undermining deniability. Achieving public eligibility verifiability without compromising coercion resistance remains a fundamental open problem in remote electronic voting.

1.1 Our Contribution

We introduce a novel cleansing protocol that ensures each eligible voter contributes at most one final ballot to the tally, while revealing no information about revoting behavior or discarded ballots. The protocol enables observers to detect ballot box stuffing and vote suppression without relying on trusted auditors or privileged access.

At a high level, ballots include commitments to voter public keys and to vote-update trapdoors. In the cleansing phase, the collector shuffles the ballots and matches the blinded public keys of the selected ones against a shuffled, reblinded registration roll, which ensures eligibility. For each dropped ballot, the collector proves knowledge of the committed trapdoor, derived from a fresher ballot of the same voter through a smartcard-seeded trapdoor chain. This mechanism also enables fake credentials and supports credential recovery. The cleansing phase runs in linear time and relies on a single publicly verifiable cleansing authority, while remaining compatible with both mixnet and homomorphic tallying. The protocol also supports practical everlasting privacy and individual verifiability, provided the smartcard secrets remain uncompromised.

Beyond the core mechanism, we also address forced abstention as a coercion strategy. We show how relay networks can provide alternative submission channels and independently verify ballot eligibility, enabling misbehavior to be detected and attributed. These properties reduce a coercer's remaining strategy

to physically preventing participation, for example by separating the voter from their smartcard.

1.2 Related Work

Remote e-voting systems typically compute the final tally using either verifiable reencryption mixnets [7, 22, 23], in which talliers successively shuffle and reencrypt ElGamal ciphertexts with zero-knowledge proofs of correct permutation, or homomorphic tallying [3], which aggregates encrypted votes and reveals the decrypted tally. Recent advances revisit homomorphic tallying with SNARK-based proofs of the result function [11], enabling publicly tally-hiding for complex ballot types at the cost of a trusted setup phase. These tallying approaches can be combined with a prior cleansing phase that ensures at most one vote in the final tally per voter.

JCJ and Encrypted Credential Schemes. The most influential coercion-resistant construction targeting distributed trust is JCJ (Juels–Catalano–Jakobsson) [12], which underlies many modern coercion-resistant voting protocols [2, 18]. JCJ publishes ElGamal-encrypted credentials on a public registration roll; voters submit encrypted votes together with encrypted credentials and zero-knowledge proofs of correct formation. Submitted credential–vote pairs are shuffled and cleansed using plaintext equivalence tests that match submitted credentials against the registration roll.

This design introduces privacy, verifiability, and efficiency challenges that have motivated substantial follow-up work. Cortier et al. [2] show that JCJ's cleansing leaks per-credential revote counts, enabling the "1009 attack"; their CHide variant closes this leak with MPC-based cleansing that reveals only the total number of ballots removed while achieving quasi-linear tallying complexity. Additional proposals [18] augment credentials with proofs of logarithm knowledge to prevent colluding cleansing authorities from decrypting registration credentials and impersonating voters. However, these improvements often come at the cost of increased computational or protocol complexity and are not easily combined without sacrificing efficiency. Moreover, the anonymous channel assumption still permits ballot box flooding attacks that can disrupt tallying.

Ledger-Row Schemes. KTV-Helios [14] and DeVoS [15] represent a class of systems in which each voter is associated with a dedicated ledger row, where their ballots are hidden among dummy ballots posted by trustees using disjunctive zero-knowledge proofs. This mechanism prevents coercers from detecting revoting behaviour while also mitigating ballot-box flooding.

In KTV-Helios, the final vote is computed as the product of all ciphertexts in a voter's ledger row. However, this approach is incompatible with homomorphic tallying techniques and requires an uninterrupted chain of recorded ballots for the final vote to remain valid. DeVoS avoids cumulative updates by allowing each new ballot to supersede the previous one, but at the cost of a two-round-trip casting protocol and a submission phase divided into micro-phases. This inflates

the audit trail and makes vote casting sensitive to micro-phase boundaries, requiring late-arriving ballots to be regenerated.

Estonia's Revoting System. Estonia's internet voting system [10, 16, 20], operational since 2005, provides the closest practical precedent for our construction. Both systems rely on smartcard-based digital identity, revoting for coercion resistance, and a single authority performing ballot cleansing. Voters may revote an unlimited number of times, with receipt-freeness following from update deniability. To address a coercer who observes the voter at the last moment, remote voting closes two hours before polling stations, allowing voters to cast superseding paper ballots [13].

A limitation of Estonia's system is that cleansing is not publicly verifiable. To ensure that a malicious cleansing authority has not dropped, added, or manipulated ballots, internal audits with privileged access to pre-mix data are employed. However, a compromised auditor can observe revoting patterns and thereby violate deniable vote updating, creating a fundamental tension between auditing and coercion resistance. Our construction resolves this tension by making the cleansing phase publicly verifiable while revealing no information about superseded ballots beyond the final tally, thereby eliminating the need for privileged auditors and strengthening coercion resistance.

2 System Overview

Property	**Smartcard**	**Voter**	**Registrar**	**Collector**	**Tallier**	**Bulletin Board**
Eligibility Verifiability	●	○	●	○	○	○
Universal Verifiability	○	○	○	●	○	○
	●	○	○	○	○	○
Individual Verifiability	●	●	○	○	○	○
Vote Anonymity	○	●	○	●	○	○
	○	●	○	○	T(●)	○
Coercion Resistance	●	○	○	●	○	○
Participation Privacy	●	○	○	●	○	○

Notation: ● – required to be honest; ○ – may be corrupt; T(·) – threshold trust.

Table 1. Security properties and trust assumptions. Security properties follow the definitions of [1]. Each row lists the minimal trust requirements for the corresponding property. Where multiple rows are given for the same property, each row represents an alternative set of trust assumptions under which the property can be achieved.

This section presents the system architecture and the trust assumptions underlying our security claims. The protocol involves the following principals:

- **Smartcard** $\mathcal{S}$: A tamper-resistant device held by the voter that generates trapdoor commitments and produces proof-of-knowledge signatures for ballots.
- **Voter** $\mathcal{V}$: The voter together with their chosen voting client, responsible for vote encryption, interacting with the smartcard, and submitting the constructed ballot to the collector.
- **Registrar** $\mathcal{R}$: Manages voter registration and publishes the registration roll listing each eligible voter's identity and corresponding public key.
- **Collector** $\mathcal{C}$: Collects ballots, verifies their validity, records the public components on the bulletin board, and performs the verifiable cleansing phase.
- **Tallier** $\mathcal{T}$: A distributed entity that produces the tally from ballots selected in the cleansing phase via homomorphic or mixnet-based tallying.
- **Bulletin Board** $\mathcal{BB}$: An append-only public ledger on which all ballot public components, cleansing and tallying transcripts, and supporting proofs are recorded.

Beyond these principals, an administrator coordinates the setup phase, which falls outside the protocol itself. Any observer may act as an **auditor** by verifying the public evidence on the bulletin board. A **relay network** may also emerge spontaneously when the collector is unavailable, providing an alternative ballot submission channel while holding the collector accountable. The collector nonetheless remains a liveness dependency for cleansing, as it alone holds the private openings; its failure can stall the tally but not enable undetected manipulation.

Table 1 summarizes the trust assumptions underlying our security claims, with each row indicating which parties are required to be honest to guarantee the corresponding property. Compared to the Estonian system, *universal verifiability*[1] can also be achieved even if the collector is corrupted, provided that the smartcard remains honest—an assumption already required for eligibility verifiability. *Individual verifiability* requires trusting the smartcard, as compromised secrets could be used to modify or invalidate a vote in a deniable manner. It also assumes that the voter performs verification on a separate device and therefore the voter is honest although the casting device does not need to be.

Vote anonymity that prevents a voter from being linked to their final counted vote is achieved either by the collector performing a shuffle, which additionally provides everlasting privacy (first row), or by the talliers performing either mixnet-based tallying or homomorphic tallying, where anonymity relies on a threshold of the talliers remaining honest (second row). Since the voter necessarily reveals their selection to the casting device, a compromised device can leak the vote — directly or kleptographically through its choice of randomness — so vote anonymity requires trusting the device that cast the final counted vote. In contrast, *coercion resistance* and *participation privacy* do not require trusting the voter's device: since the voter may recast from another device, evidence

[1] Eligibility and universal verifiability additionally assume the smartcard is not used without the voter's authorization, whether surrendered voluntarily or invoked covertly; this threat and its mitigations are discussed in [16].

from any single compromised device — an observed vote, abstention, or forced participation superseded by a null vote — remains inconclusive to the coercer[2].

System Timeline. The protocol proceeds in three phases. In the *setup phase*, the registrar issues smartcards and publishes the registration roll. In the *voting phase*, voters cast ballots using their smartcards; after submission, each voter may independently verify that their ballot appears on the bulletin board and is cast as intended. In the *tallying phase*, the collector performs the cleansing procedure and the tallier produces the final tally, both publishing their evidence on the bulletin board for subsequent public audit.

3 Preliminaries

Notation. Let $\mathbb{Z}_q$ be the integers modulo q, $\mathbb{G}_q$ a group of order q in which the discrete logarithm problem is hard, and $\mathbb{S}_n$ the symmetric group. We fix generators $(\mathsf{h}, \mathsf{g}) \in \mathbb{G}_q^2$ such that $\log_{\mathsf{g}} \mathsf{h}$ is unknown. Commitments to $y \in \mathbb{G}_q$ are $\mathrm{Com}(y) = \mathsf{h}^{\xi_y} y$ for $\xi_y \in_R \mathbb{Z}_q$. ElGamal encryption of $m \in \mathbb{Z}_q$ under $\mathsf{pk} = \mathsf{g}^x$ is $\mathrm{Enc}_{\mathsf{pk}}(m) = (\mathsf{g}^r, \mathsf{pk}^r \mathsf{g}^m)$ for $r \in_R \mathbb{Z}_q$; reencryption is $\mathrm{ReEnc}_{\mathsf{pk}}((a,b)) = (a\mathsf{g}^r, b\mathsf{pk}^r)$ and decryption $\mathrm{Dec}_x((a,b)) = b/a^x$. We model $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_q$ as a random oracle.

We also introduce an encryption scheme for field elements based on a Diffie–Hellman key encapsulation mechanism. For $t \in \mathbb{Z}_q$, encryption under the public key $\mathsf{pk} = \mathsf{g}^x$ is defined as $\mathrm{Enc}^\star_{\mathsf{pk}}(t; r) = (\mathsf{g}^r,\ t + \mathcal{H}(\mathsf{pk}^r) \pmod q)$, where $r \in_R \mathbb{Z}_q$. Given a ciphertext (a, s), decryption computes the shared Diffie–Hellman secret a^x and outputs $\mathrm{Dec}^\star_x((a,s)) = s - \mathcal{H}(a^x) \pmod q$. Correctness follows since $a = \mathsf{g}^r$ implies $a^x = (\mathsf{g}^r)^x = \mathsf{pk}^r$, and therefore $\mathrm{Dec}^\star_x(\mathrm{Enc}^\star_{\mathsf{g}^x}(t; r)) = t$.

A relation $\mathcal{R} \subseteq \mathbb{G}_q^* \times \mathbb{Z}_q^*$ associates a vector of public group elements (statement) $\mathbf{s} \in \mathbb{G}_q^*$ with a vector of witnesses $\mathbf{w} \in \mathbb{Z}_q^*$. We write $\pi = \mathrm{PoK}\{\mathbf{w} : (\mathbf{s}, \mathbf{w}) \in \mathcal{R}\}$ to denote a non-interactive proof of knowledge of witnesses $\mathbf{w}$ satisfying the relation with respect to statement $\mathbf{s}$, and $\mathrm{Verify}_{\mathcal{R}}(\mathbf{s}, \pi)$ for verification (e.g., $\mathrm{PoK}\{x : y = \mathsf{g}^x\}$). Witnesses generated internally during proof construction and never leaving the prover's trust boundary (e.g., blinding or reencryption factors) are treated as implicit prover randomness and omitted from the relation and witness list for clarity. Auxiliary data is written $\mathrm{PoK}^{(A;B)}\{\cdot\}$, where A and $\mathcal{H}(B)$ are included in the Fiat–Shamir challenge $\mathcal{H}(\mathbf{s}, A, \mathcal{H}(B), \ldots)$; A is public, while only $\mathcal{H}(B)$ is revealed (e.g., in π), so B remains hidden but bound to the proof. Verification is denoted $\mathrm{Verify}^{(A;B)}_{\mathcal{R}}(\mathbf{s}, \pi)$.

[2] A realistic example is a workplace where employees are provided with company-managed laptops or mobile phones that may be subject to monitoring. A voter wishing to oppose the coercer's instructions may cast the requested vote using the monitored device and subsequently recast their intended vote using a personal device.

4 Protocol Specification

Design Rationale. Our construction reverses the burden of proof in cleansing: rather than proving the selected ballots correct, the collector proves that every *discarded* ballot was superseded, by opening its trapdoor commitment. The smartcard makes this possible—and unavoidable—by stepping backward along a trapdoor chain: each ballot commits to a trapdoor t while handing the collector, encrypted, the value $\mathcal{H}(t)$—precisely the trapdoor committed in the voter's previous ballot. Each ballot thus opens the one it supersedes while the newest stays one preimage out of reach—a voter committing t_5, t_4, t_3 lets the collector open the first two but not t_3—forcing every final ballot into the selection. Eligibility is established by first shuffling accepted ballots and the registration roll independently, and then proving that every selected ballot's public-key commitment matches a reblinded registration-roll entry. Since each reblinded commitment is unique, this simultaneously proves that every selected ballot belongs to a registered voter and that at most one final ballot is selected per voter, without linking ballots to voters.

4.1 Setup Phase

The registrar issues a smartcard to each eligible voter and publishes the registration roll listing each voter's identity and corresponding public key $((\mathsf{Id}_i, \mathsf{pk}_i))_i$, upon which all parties agree. Each smartcard contains a trapdoor seed $\mathsf{seed}_i \in_R \mathbb{Z}_q$ from which the vote-update trapdoors are derived by iterated hashing, forming the voter's trapdoor chain. The collector generates a secret key $y \in_R \mathbb{Z}_q$ and publishes the corresponding public key $\mathsf{pk}_\mathcal{C} = \mathsf{g}^y$, under which smartcards encrypt trapdoors. The tallier generates a secret key and publishes a public key $\mathsf{pk}_\mathcal{T}$ under which voters encrypt their votes.

4.2 Voting Phase[3]

Let N denote the number of voting options. The voter selects a voting option and encodes it as an element $v \in \mathbb{Z}_N$. The voter then samples a randomization factor $r \in_R \mathbb{Z}_q$ and commitment blinding factors $(\alpha, \beta) \in_R \mathbb{Z}_q^2$, computes the ElGamal ciphertext $V = \mathrm{Enc}_{\mathsf{pk}_\mathcal{T}}(v; r)$, and sends (α, β, V) to the smartcard.

The smartcard computes a vote-update trapdoor $t_n = \mathcal{H}^{(n)}(\mathsf{seed})$, and encrypts the previous trapdoor for the collector as $T = \mathrm{Enc}^\star_{\mathsf{pk}_\mathcal{C}}(t_{n+1}) = \mathrm{Enc}^\star_{\mathsf{pk}_\mathcal{C}}(\mathcal{H}(t_n))$. It then constructs the credential commitment $E = \mathsf{h}^\beta \mathsf{pk}$ and the trapdoor commitment $C = \mathsf{h}^\alpha \mathsf{g}^{t_n}$. To bind these values it produces the proof

$$\pi_E = \mathrm{PoK}^{(C,V;\alpha,T)}\{(\beta, x) : E = \mathsf{h}^\beta \mathsf{g}^x\}. \tag{1}$$

The smartcard returns (C, E, T, π_E) to the voter.

[3] Voter indices are suppressed throughout this subsection, as each voter's interaction with the smartcard and collector is independent.

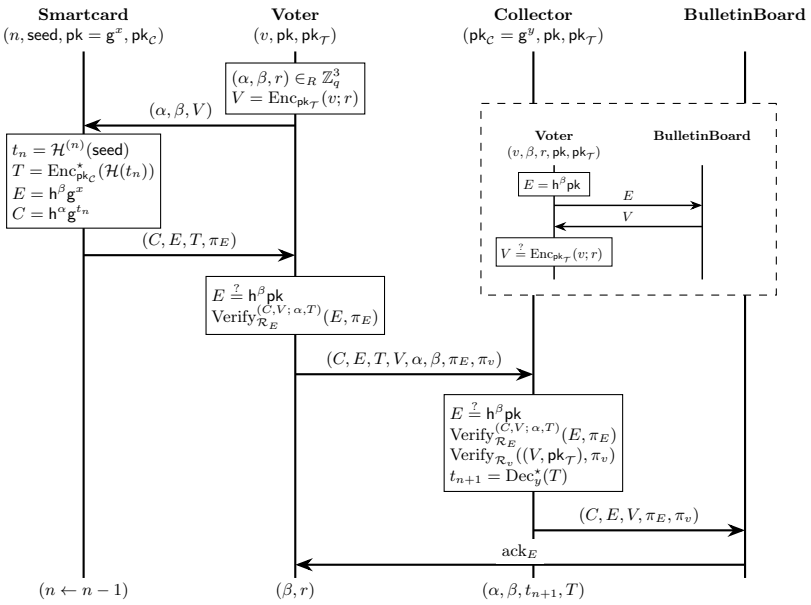

Fig. 1. The vote casting protocol with individual verifiability (inset). The protocol starts with the voter selecting a choice $v \in \mathbb{Z}_N$ and forming its encryption, which is sent to the smartcard for signing. The voter checks the validity of the signature locally before forwarding the ballot to the collector. If the ballot is valid, the collector records the public part on the bulletin board and stores $(\alpha, \beta, t_{n+1}, T)$ required for the cleansing protocol. Once the voting protocol ends, the voter retains (β, r) with which it can verify its ballot from another device using the verification protocol shown in the inset.

The voter verifies that $E \stackrel{?}{=} \mathsf{h}^{\beta}\mathsf{pk}$ and checks $\mathrm{Verify}^{(C,V;\alpha,T)}_{\mathcal{R}_E}(E, \pi_E)$. If all checks succeed, the voter submits the ballot to the collector:

$$\mathsf{ballot} = (C, E, T, V, \alpha, \beta, \pi_E, \pi_v) \tag{2}$$

Here π_v is a proof constructed by the voter demonstrating knowledge of the ciphertext randomization (to prevent the Pfitzmann attack [17]), or a range proof when homomorphic tallying is employed [3] or when forced abstention through vote randomization is a concern.

The collector repeats the verification steps of the ballot and checks the vote encryption $\mathrm{Verify}_{\mathcal{R}_v}(V, \pi_v)$. If all checks succeed, it performs trapdoor decryption $t_{n+1} = \mathrm{Dec}^{\star}_{y}(T)$, stores the private openings and trapdoor encryption $(\alpha, \beta, t_{n+1}, T)$, and publishes the public ballot components (C, E, V, π_E, π_v) on the bulletin board. Since π_E binds $\mathcal{H}((\alpha, T))$ into the Fiat–Shamir challenge, a substituted trapdoor encryption invalidates the ballot at submission; hence a trapdoor encryption T inconsistent with the trapdoor commitment C can only originate from the smartcard, to which the fault is attributed when detected dur-

ing cleansing (Section 4.3). The bulletin board then returns an acknowledgment ack_E to the voter, certifying that the blinded public key E (and consequently the associated public ballot components) has been recorded[4].

Relay-Verifiable Ballots. If the voter cannot reach the collector directly, the ballot may instead be submitted through a relay. A relay receiving the ballot performs the same verification steps as the collector, allowing it to confirm the ballot's eligibility before attempting to record it on the bulletin board. If submission fails, the relay may forward the ballot to other relays, each of which can independently verify and attempt submission. This relay network makes the collector accountable and ensures that voters can cast their ballots even if the collector becomes unreachable.

Individual Verifiability. The voter verifies their ballot from another device as follows. They reconstruct the credential commitment $E = \mathsf{h}^{\beta}\mathsf{pk}$ from their public key pk and blinding factor β, and use it to locate their ballot on the bulletin board. They then check, using r and v, that $V \stackrel{?}{=} \text{Enc}_{\mathsf{pk}_\mathcal{T}}(v; r)$.

4.3 Tallying Phase

The tallying phase begins with cleansing, which is performed entirely by the collector. The collector first applies a shuffle to the list of tuples $((C_i, E_i, V_i))_i$ using a secret permutation $\psi \in_R \mathbb{S}_I$. It then publishes the shuffled list $((C'_i, E'_i, V'_i))_i$ together with a zero-knowledge proof of the correctness of the shuffle:

$$\pi_{CEV} = \text{PoK}\{\psi : C'_i = \text{Com}(C_{\psi(i)}) \wedge E'_i = \text{Com}(E_{\psi(i)}) \wedge V'_i = \text{ReEnc}(V_{\psi(i)})\} \tag{3}$$

where the statement holds for all recorded ballots $i \in [I]$ and the blinding factors are chosen so that every E'_i is unique.

From the shuffle output, the collector selects the index set $L \subseteq [I]$ corresponding to each voter's last cast ballot. To verify eligibility of each selected index $i \in L$, a new shuffle is formed from the registration roll public keys $(\mathsf{pk}_k)_k$ via a permutation $\phi \in_R \mathbb{S}_K$. The shuffle produces blinded public keys $(E''_k)_k$ with blinding factors chosen so that $\{E'_i\}_{i\in L} \subseteq \{E''_k\}_k$, and is supported by a zero-knowledge proof:

$$\pi_{\text{Elig}} = \text{PoK}\{(\phi, \{\beta_{\psi(i)}\}_{i\in L}) : E''_k = \text{Com}(\mathsf{pk}_{\phi(k)}) \wedge \{E'_i\}_{i\in L} \subseteq \{E''_k\}_k\} \tag{4}$$

where the statement holds for all registered public keys $k \in [K]$.

Finally, to demonstrate that the selected indices L correspond to each voter's last cast ballot and that no eligible ballot has been omitted, a proof of knowledge is constructed for the trapdoor associated with each dropped ballot:

$$\pi_{\text{Drop}} = \bigwedge_{i \notin L} \text{PoK}\{(\alpha'_i, t_i) : C'_i = \mathsf{h}^{\alpha'_i}\mathsf{g}^{t_i}\} \tag{5}$$

[4] The acknowledgment may simply consist of the bulletin board's digital signature on E, or of a commitment to a batch of published entries, such as a digitally signed Merkle tree root containing E as one of its leaves.

Together, these proofs form a publicly verifiable transcript of the cleansing phase published on the bulletin board:

$$\mathsf{tr} = \big(((C_i, E_i, V_i, \pi_{E_i}, \pi_{v_i}))_i\,,\ ((C'_i, E'_i, V'_i))_i\,,\ (E''_k)_k\,,\ \pi_{CEV},\ \pi_{\mathrm{Elig}},\ \pi_{\mathrm{Drop}}\big)$$

which anyone can verify using the registration roll $((\mathsf{Id}_k, \mathsf{pk}_k))_k$. The selected vote ciphertexts $\{V'_i\}_{i\in L}$ are then passed through a reencryption mixnet [22] or homomorphically tallied [3].

Trapdoor Reduction. The drop proof π_{Drop} requires the collector to know a full opening (α'_i, t_i) of each shuffled commitment C'_i with $i \notin L$. The blinding factor α'_i is always known to the collector, being the sum of the α_i received at submission and its own reblinding factor from the shuffle. The trapdoor component t_i, however, must be derived; the collector obtains it by finding the trapdoor of lowest index in each voter's trapdoor chain among the recorded ballots.

For a given trapdoor t_*, define the set of reachable trapdoors

$$F(t_*) = \{\mathcal{H}^{(n)}(t_*) : n \in [M]\}, \tag{6}$$

where M is an upper bound on the trapdoor-chain length that the smartcard is allowed to use. For each public key, the collector considers the trapdoor values decrypted from that key's ballots and selects the value t_* of lowest index, i.e., the unique decrypted value such that every other decrypted value lies in $F(t_*)$. Such a value exists because revotes with the genuine credential step backward through the chain: each new ballot commits the preimage of the previously committed trapdoor, and its encrypted trapdoor decrypts to that previously committed value, so the decrypted values all lie on a single chain, one step ahead of the corresponding commitments. Every superseded ballot therefore has its trapdoor $t_j \in \{t_*\} \cup F(t_*)$, and its commitment $C_j = \mathsf{h}^{\alpha_j}\mathsf{g}^{t_j}$ can be opened. The remaining commitment $C_i = \mathsf{h}^{\alpha_i}\mathsf{g}^{t_i}$ commits to the preimage t_i of t_*, i.e., $\mathcal{H}(t_i) = t_*$, which the collector cannot obtain without inverting $\mathcal{H}$; it is therefore forced to select ballot i as the final one.

If trapdoor reduction fails, the collector publishes all $(\alpha_i, \beta_i, t_i, T_i, \pi_{\mathrm{Dec}^\star_i})$ associated with the given public key, where $\pi_{\mathrm{Dec}^\star_i} = \mathrm{PoK}\{y : t_i = \mathrm{Dec}^\star_y(T_i)\}$ proves that t_i was correctly derived from the encrypted trapdoor T_i. This provides verifiable evidence that the corresponding voter's smartcard has been compromised, allowing the vendor to be held accountable.

Fake Credentials. The scheme supports fake credentials implemented as smartcard PIN codes by assigning each a higher trapdoor-chain index than the genuine credential. Each PIN behaves as a genuine credential until a higher-precedence PIN is used, after which it behaves as a fake credential relative to that PIN while remaining genuine with respect to lower-precedence PINs. Consequently, this provides a weaker form of coercion resistance than [2,12,14]; however, it still defends against coercion strategies in which a coercer arrives at the last moment to observe the voter casting the final ballot.

The genuine credential traverses the trapdoor chain backward ($n \leftarrow n-1$), making each new trapdoor commitment $\mathsf{g}^{t_i} = C_i/\mathsf{h}^{\alpha}$ unopenable and ensuring that the last genuine ballot survives. Fake credentials traverse the chain forward ($n \leftarrow n+1$), making their commitments openable once a higher-precedence credential has been used. This permits an arbitrary number of revotes, limited only by the trapdoor-chain length.

Interleaving credentials requires care. Trapdoors must never be reused, as losing a ballot with a lower-index trapdoor could leave multiple commitments unopenable, preventing the collector from identifying a unique surviving ballot. Since ballot delivery is not guaranteed, the smartcard advances a fake credential to the highest unused chain position after each genuine invocation. The resulting ballot remains openable from the preceding fake ballot's encrypted trapdoor, preserving first-ballot precedence among fake credentials.

A useful side effect of this design is that it enables credential recovery without exposing the voter to additional coercion risks. A recovery credential can be printed on paper and used if the voter forgets their genuine credential. The genuine credential remains secret and can still be used to evade coercion if the coercer demands the printed credential. This approach also improves usability: since remembering multiple PIN codes can be error-prone, the voter need only remember one credential, from which fake credentials can be derived whenever needed.

5 Security Analysis

This section provides an overview of the security mechanisms that ensure the properties outlined in Table 1 under the specified trust assumptions.

Eligibility Verifiability Eligibility verifiability ensures that every selected ballot for tallying is authorized by an eligible voter who contributes at most one ballot. Assuming the smartcard's signing key has not been compromised and the smartcard has not been used without the voter's authorization [16], the proof proceeds as follows. The proof π_{E_i}, being a proof-of-knowledge signature over the ballot, establishes that the submitter knows the private key underlying $E_i = \mathrm{Com}(\mathsf{pk}'_i)$ and binds this credential to the vote encryption V_i. The shuffle proof π_{CEV} ensures that the shuffled commitments E'_i open to the same public keys as the originals. The registration-roll shuffle $\mathcal{R}_{\mathrm{Elig}}$ produces commitments $E''_k = \mathrm{Com}(\mathsf{pk}_{\phi(k)})$ to the eligible keys. The verified subset relation $\{E'_i\}_{i\in L} \subseteq \{E''_k\}_k$ then forces each selected ballot's credential to be an eligible public key. Since the registration roll contains one key per voter, at most one selected commitment E'_i maps to each registration key, giving at most one selected ballot per voter for tallying.

Universal Verifiability Universal verifiability ensures that the final tally consists only of accepted ballots that have not been manipulated or dropped, with

each voter contributing at most one valid vote. Here we focus on cleansing verifiability with corrupt collector, as tallying verifiability of the selected ballots is well understood [3,7,11,22]. We show that the selected set L contains the final ballot of every voter, as determined by the smartcard's trapdoor chain, and that no eligible voter's final ballot is omitted, assuming the smartcard neither leaks the trapdoor seed to the collector nor is used without the voter's authorization [16], preventing an adversary from generating a nullifying trapdoor.

The shuffle proof π_{CEV} ensures that each shuffled commitment C'_i opens to the same trapdoor value g^{t_i} as its pre-shuffle counterpart. The proof π_{Drop} establishes that the collector knows an opening of C'_i for every $i \notin L$. Since each ballot's encrypted trapdoor T_i reveals $\mathcal{H}(t_i)$ to the collector, the trapdoors of superseded ballots are derivable by iterating $\mathcal{H}$, so an honest collector can always open the commitments outside L.

For the final ballot, however, obtaining t_i from the known value $\mathcal{H}(t_i)$ would require inverting $\mathcal{H}$. Under the one-wayness of $\mathcal{H}$ and the binding property of Pedersen commitments, the collector cannot open C'_i and is therefore forced to place each voter's final ballot in L. Because every commitment opened in π_{Drop} carries the credential of a voter whose final ballot is thereby in L, no eligible voter's cast final ballot is dropped, and the collector cannot substitute a superseded ballot for the final one.

If no consistent trapdoor chain exists within the bound M, the collector publishes the stored openings together with proofs of correct trapdoor decryption, providing publicly verifiable evidence of smartcard misbehavior and allowing the affected ballots to be discarded.

Individual Verifiability Individual verifiability requires that an adversary cannot substitute the voter's verification target with another voter's ballot matching the same vote selection, and that the verified ballot is the final one tallied. In the proposed system, these properties hold assuming the smartcard is uncompromised and has not been used without the voter's authorization [16].

The voter locates their ballot (E, V) on the bulletin board and knows from their smartcard invocation history (genuine or fake) whether it will be selected for tallying. To verify ownership, the voter opens their credential commitment $E = \mathsf{h}^{\beta}\mathsf{pk}$ using their known blinding factor β, where pk is listed on the registration roll. The proof π_E establishes knowledge of (β, x) such that $E = \mathsf{h}^{\beta}\mathsf{g}^{x}$, and since g and h are independent generators, the commitment cannot be opened to a different public key under the discrete logarithm assumption, even with knowledge of x. This binds the ballot to the voter's credential, and the voter confirms their selection by checking the vote ciphertext $V \stackrel{?}{=} \text{Enc}_{\mathsf{pk}_{\mathcal{T}}}(v; r)$ using their secret randomization factor r and chosen vote v.

Participation Privacy Participation privacy comprises two properties: participation anonymity, where no observer can determine from public evidence whether a voter participated, and participation receipt-freeness, where the voter cannot prove to a third party whether they participated or abstained. A ballot

(C, E, V, π_E, π_v) posted on the bulletin board contains no information identifying the voter. The credential commitment $E = \mathsf{h}^{\beta}\mathsf{pk}$ is perfectly hiding, so without knowledge of β every public key on the registration roll is an equally plausible opening. The proof π_E is a proof of knowledge that reveals nothing beyond the submitter's ability to open the commitment, and the shuffled commitments $\{E'_i\}_i$ are never opened. A coercer therefore cannot determine from the public record whether a given voter's key appears among the posted ballots, and the voter cannot produce a convincing proof of abstention. Defense against forced participation, where a coercer compels the voter to cast a specific vote, is addressed under coercion resistance via deniable vote updating and null vote casting.

Coercion Resistance Coercion resistance requires that a voter can execute a counter-strategy such that the coercer cannot distinguish whether the voter complied with their demands or deviated in favour of their own intent. The proposed system supports two mechanisms: deniable vote updating and fake credentials. Both assume an honest smartcard and collector. We additionally require that the voter's chosen plaintext is not unique in the final tally if plaintext-based tallying is used.

Receipt-Freeness. A voter who wishes to sell their vote or prove compliance to a coercer can disclose the ballot contents $(C, E, T, V, \alpha, \beta)$ and demonstrate that this ballot appears on the bulletin board. However, to issue a conclusive receipt the voter must also prove that this ballot is included in the final tally rather than superseded by a ballot with a lower order trapdoor commitment. This requires establishing a link between the pre-shuffle ballot and a post-shuffle entry in L which is not possible as π_{CEV} and π_{Drop} are zero knowledge. Moreover, the cleansing phase reveals no revoting statistics—the set L discloses only the number of distinct voters, not how many times any voter cast a ballot—thereby excluding the "1009 attack" [2]. The voter therefore cannot produce a convincing receipt that their disclosed ballot is the tallied one. A separate receipt channel arises if the voter's plaintext choice is unique on the tally board (the Italian attack); homomorphic tallying is inherently immune to this as it hides individual votes, whereas mixnet-based tallying must employ additional countermeasures.

Vote Update Deniability. By receipt-freeness, the coercer cannot verify that any observed ballot is included in the final tally. The voter's counter-strategy is therefore to comply with the coercer and later recast their vote in private using the genuine credential. The coercer cannot detect the voter's subsequent session from the public record, since each session uses fresh blinding factors (α, β) and a new trapdoor t_n, producing indistinguishable credential and trapdoor commitments making cast ballots unlinkable. The trapdoor index n is hidden by encrypting the trapdoor. Consequently, the smartcard outputs do not allow a coercer to determine whether the voter has already voted or submitted a revote.

Fake Credentials in the proposed system can still cast valid ballots, making them a weaker form than those in [2, 12], where ballots cast with fake credentials are discarded entirely. A fake credential occupies a higher position in the

trapdoor chain than the genuine credential, so a ballot cast with the genuine credential always takes precedence during cleansing. This protects against coercion scenarios in which a coercer arrives at the last moment and forces the voter to cast the final ballot under observation: if the voter uses a fake credential under coercion, they can cast a ballot with their genuine credential at any time during the voting phase, which then takes precedence. A residual attack vector arises if the coercer demands both the genuine and fake credentials; however, this is mitigated because the coercer does not know how many fake credentials the voter may have generated and therefore always faces the possibility of being one credential short of the genuine one.

Everlasting Privacy The system offers practical everlasting privacy [9] assuming an honest collector and that the submission channel resists external timing analysis. The channel need not hide the voter's identity from the collector, who necessarily learns it through the private openings (α, β, t_{n+1}). A possible mechanism for providing such timing-analysis resistance is discussed in Section 6.

A ballot posted on the bulletin board contains the commitments (C, E) and vote ciphertext V. A future adversary with unlimited computational power could decrypt the vote ciphertext V. However, the Pedersen commitments $E = \mathsf{h}^{\beta}\mathsf{pk}$ and $C = \mathsf{h}^{\alpha}\mathsf{g}^{t_n}$ are perfectly hiding: for any public key pk' on the registration roll, a suitable blinding factor β' exists such that $E = \mathsf{h}^{\beta'}\mathsf{pk}'$, and similarly for C. The shuffled commitments (C', E') inherit the same property. Since the proofs π_E, π_{CEV}, π_{Drop} are zero-knowledge and therefore of no additional help, a future adversary cannot find out who cast the corresponding encrypted vote V.

6 Discussion and Implementation

Strengthening Individual Verifiability Currently individual verifiability depends on the smartcard's secrets remaining confidential and on no unauthorized ballots having been issued with it [16]. These trust assumptions could be relaxed by supplementing the protocol with vote trackers. Since cast-as-intended verifiability is already assured, the tracker need only prove that a ballot was selected for tallying, requiring only the publication of tracker values. This can be achieved with tracker-based systems that offer receipt-freeness [4]. A further goal would be to assign trackers to non-participating voters, enabling them to verify non-participation in a deniable manner, but this necessitates further research.

Resisting Metadata Analysis Everlasting privacy can be strengthened by releasing ballots in a shuffled order. To make this accountable, the collector first publishes blinded commitments $E_i^* = \mathsf{g}^{\rho_i} E_i$ during the voting phase and returns (ρ_i, E_i^*) as an acknowledgment to the voter. Once the voting phase closes, the collector performs a shuffle with permutation $\tau \in \mathbb{S}_I$ supported by a proof $\pi_{\text{Anon}} = \text{PoK}\{\tau, \{\rho_i\}_i : E_i = \mathsf{g}^{-\rho_i} E^*_{\tau(i)}\}$ and releases $(E_i)_i$ together with the remaining ballot components.

Implementation Ballot creation requires two Pedersen commitments with one opening proof and one ElGamal encryption, amounting to seven group exponentiations and three multiplications in $\mathbb{G}_q$, which is well within the capabilities of existing smartcards used for digital signatures. A single trapdoor chain is sufficient when voting is non-concurrent; for concurrent elections, multiple chains can be installed. With 128-bit trapdoor security and 1000 revotes shared across genuine and fake credentials, the full chain requires up to 15 kB of storage.

The shuffle proofs can be implemented using existing ElGamal reencryption shuffle proofs such as Verificatum [7,22,23], with the generator and public key for commitment shuffles set equal to the blinding generator, and the vote reencryption shuffle using the standard generator and tallier's public key combination. The drop proof is linear in the number of dropped ballots, and trapdoor root finding scales with the chain length. The cleansing phase therefore scales linearly, making it practical for large-scale elections.

7 Conclusion

We have presented a novel construction to provide public verifiability for the ballot cleansing phase. The protocol offers perfect cleansing privacy, scales linearly with a footprint approaching two shuffle proofs, imposes minimal smartcard requirements, and decouples from the tallying scheme, enabling integration with both homomorphic and mixnet-based tallying.

The scheme enables Estonian-style voting systems to achieve public cleansing verifiability without exposing internal data to auditors, eliminating a potential access point for coercers. Although votes are ElGamal encrypted, blinding the public keys provides practical everlasting privacy, enabling the publication of election evidence. Resilience against corrupt collectors is provided by relay networks that independently verify ballot eligibility and offer alternative submission channels. Individual verifiability is also strengthened: voters verify their ballot directly from the bulletin board, including the signer's public key, precluding verification of another voter's ballot.

Future work should address smartcard trust assumptions by coupling the protocol with tracker constructions, which may also enable verifiable and deniable proof of non-participation.

Acknowledgment

We are particularly grateful to the reviewer who suggested replacing the span-proof machinery of an earlier version with direct proofs of knowledge of the dropped ballots' trapdoor openings — a simplification that substantially improved this paper. Reviewer feedback also prompted us to simplify the trapdoor-chain construction. We also thank Véronique Cortier and Jan Willemson for helpful feedback on an earlier draft of this manuscript.

References

1. Benaloh, J., Rivest, R.L., Ryan, P.Y.A., Stark, P.B., Teague, V., Vora, P.L.: End-to-end verifiability. ArXiv (2015)
2. Cortier, V., Gaudry, P., Yang, Q.: Is the JCJ voting system really coercion-resistant? In: IEEE Computer Security Foundations Symposium (2024)
3. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Advances in Cryptology — EUROCRYPT '97. Springer (1997)
4. Damodaran, A., Rastikian, S., Rønne, P.B., Ryan, P.Y.A.: Hyperion: Transparent end-to-end verifiable voting with coercion mitigation. IACR ePrint (2024)
5. Grewal, G.S., Ryan, M.D., Bursuc, S., Ryan, P.Y.A.: Caveat Coercitor: Coercion-evidence in electronic voting. In: IEEE Symposium on Security and Privacy (2013)
6. Gutmann, P., Neuhaus, S.: Replication of quantum factorisation records with an 8-bit home computer, an abacus, and a dog. IACR Cryptology ePrint Archive (2025)
7. Haenni, R., Locher, P., Koenig, R., Dubuis, E.: Pseudo-code algorithms for verifiable re-encryption mix-nets. In: Financial Cryptography and Data Security. Springer (2017)
8. Haenni, R., Spycher, O.: Secure internet voting on limited devices with anonymized DSA public keys. In: USENIX EVT/WOTE (2011)
9. Haines, T., Mosaheb, R., Müller, J., Pryvalov, I.: Sok: Secure e-voting with everlasting privacy. Proceedings on Privacy Enhancing Technologies (2023)
10. Heiberg, S., Martens, T., Vinkel, P., Willemson, J.: Improving the verifiability of the Estonian internet voting scheme. In: Electronic Voting. Springer (2017)
11. Huber, N., Küsters, R., Krips, T., Liedtke, J., Mueller, J., Rausch, D., Reisert, P., Vogt, A.: Kryvos: Publicly tally-hiding verifiable e-voting. ACM CCS (2022)
12. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: ACM Workshop on Privacy in the Electronic Society (2005)
13. Krips, K., Willemson, J.: On practical aspects of coercion-resistant remote voting systems. In: Electronic Voting. Springer (2019)
14. Kulyk, O., Teague, V., Volkamer, M.: Extending Helios towards private eligibility verifiability. In: E-Voting and Identity. Springer (2015)
15. Mueller, J., Pejó, B., Pryvalov, I.: Devos: Deniable yet verifiable vote updating. IACR Cryptol. ePrint Arch. **2023**, 1616 (2024)
16. Pereira, O.: Individual verifiability and revoting in the Estonian internet voting system. In: Financial Cryptography and Data Security. Springer (2023)
17. Pfitzmann, B.: Breaking an efficient anonymous channel. In: EUROCRYPT. LNCS, vol. 950, pp. 332–340. Springer (1995)
18. Rønne, P.B.: JCJ with improved verifiability guarantees. EVoteID (2016)
19. Smyth, B., Frink, S., Clarkson, M.R.: Computational election verifiability: Definitions and an analysis of Helios and JCJ. IACR Cryptology ePrint Archive (2015)
20. Springall, D., Finkenauer, T., Durumeric, Z., Kitcat, J., Hursti, H., MacAlpine, M., Halderman, J.A.: Security analysis of the Estonian internet voting system. In: ACM Conference on Computer and Communications Security (2014)
21. Waintal, X.: The quantum house of cards. Proceedings of the National Academy of Sciences of the United States of America (2023)
22. Wikström, D.: A sender verifiable mix-net and a new proof of a shuffle. In: ASIACRYPT. Springer (2005)
23. Wikstrom, D.: How to implement a stand-alone verifier for the verificatum mix-net. verificatum.org (2011)