Voting with coercion resistance and everlasting privacy using linkable ring signatures

Panagiotis Grontas[0000−0001−7584−0643]12, Aris Pagourtzis[0000−0002−6220−3722]12, and Marianna Spyrakou[0000−0001−5694−8405]1

School of Electrical and Computer Engineering, National Technical University of Athens and Archimedes, Athena Research Center, Greece
pgrontas@corelab.ntua.gr, pagour@cs.ntua.gr, mspyrakou@mail.ntua.gr

Abstract. We present a novel linkable ring signature scheme that possesses unconditional anonymity together with a voting protocol that utilizes it to achieve coercion resistance and everlasting privacy. In our system, all voters create private credentials and register their public counterparts. To vote, they create a ring (anonymity set) consisting of public credentials together with a proof of knowledge of their secret credential by using our novel signature. Its unconditional anonymity prevents an attacker, no matter how powerful, from deducing the identity of the voter, thus attaining everlasting privacy. Additionally, our protocol provides coercion resistance in the JCJ framework; when an adversary tries to coerce a voter, the attack can be evaded by creating a signature with a fake but indistinguishable credential. During a moment of privacy, the voter will cast their real vote. Our scheme also provides verifiability and ballot secrecy.

Keywords: e-voting, coercion resistance, everlasting privacy, linkable ring signatures, unconditional anonymity

1 Introduction

Voting is a distributed decision-making process. Voters submit their opinions, talliers aggregate them, and everyone is bound by the result. Electronic voting aims to improve the speed, cost and accessibility of elections. To successfully do so, it must satisfy a set of conflicting security properties. Verifiability [1] removes the trust in the various system components by letting voters verify each part of the process. Privacy [2] encourages voters to express their true opinions. It can be implemented using secrecy [3] and/or anonymity [4]. In its most basic form, it protects only against other voters and the election talliers through cryptography under computational assumptions. Everlasting privacy [5] protects voters even after such assumptions cease to hold. Extended notions of privacy protect against corrupted voters who wish to sell their votes (receipt-freeness [6]) and active adversaries, who are present during ballot casting, and seek to coerce a particular voting behavior using threats in case of non-compliance or rewards. Coercion resistance [7] is the strongest form of privacy that must be satisfied so that remote e-voting can be broadly adopted.

Overview of our contribution. We make two main contributions in this work: we propose a novel linkable ring signature scheme that possesses unconditional anonymity and a new e-voting protocol build on top of it that achieves coercion resistance, everlasting privacy, along with ballot secrecy and verifiability.

In our linkable ring signature, the private keys (credentials) of the participants are tuples (x, y) of field elements similar to [8]. Linking is performed by using a tag derived from the first component of the private credential, i.e. $\mathtt{t} = e^x$. Public keys are produced from perfectly hiding Pedersen commitments to x, where y can be considered as the randomness. In this manner, a public key corresponds to many possible private keys. This prevents an unbounded adversary from retrieving the full private credential from its public counterpart, even if they can retrieve x from $\mathtt{t}$. A signer can construct a ring of public keys ad-hoc, without any setup or trusted party, by simply selecting a set of public keys belonging to other participants thus using them as an anonymity set. The signature is generated by sequentially proving in zero-knowledge, knowledge of the corresponding private key for one of the public keys in the ring [9].

We then use our signature scheme as a building block in our new e-voting protocol in a way that achieves coercion resistance and everlasting privacy, together with verifiability and ballot secrecy with minimal trust assumptions. The voters register their public signing credentials before the election begins. Each of our ballots consists of an encrypted vote under the public key of the tallying authority together with proofs of well-formedness. It is then signed using our linkable ring signature where the ring is selected ad-hoc from all other registered public keys and posted on a public bulletin board. The unconditional anonymity provided by our signature scheme prevents an adversary from deducing the identity of the signer (voter), even if they can 'break' the encryption of the vote. Coercion resistance can be provided by faking the credentials used to cast the vote when the coercer is present, following the JCJ framework [7]. These fake credentials are generated on the fly and are indistinguishable from the registered ones. In a moment of privacy, the voter can use their genuine credentials to cast their real vote. During tallying, the election authority verifiably filters out the votes with fake credentials, without the coercer being able to identify them. All actions of the voters and the various authorities are accompanied by non-interactive zero-knowledge proofs, ensuring the integrity and public verifiability of the election.

Paper organisation. The rest of the paper is organised as follows: in section 2 we introduce the notation used as well as the cryptographic building blocks that we utilize. In section 3 we present our novel linkable ring signature scheme and analyze its security properties. In section 4 we present our novel e-voting protocol and analyze its security properties in section 5. In section 6 we discuss related work. This section is quite extensive in order to better explain the design choices we made for our scheme. Due to its extent, this review can be considered as a further contribution of this work. Finally, in section 7 we conclude the paper and discuss possible future research directions.

2 Preliminaries

Notation. Let λ denote the security parameter. The number of voters is n and each voter is indexed by $i \in [n] = \{1, \ldots, n\}$. We use the term *vote* to refer to the choice of the voter (in plaintext), while the term *ballot* stands for the encrypted vote together with credentials and proofs of validity. In our scheme, each ballot contains exactly β credentials indexed by $j \in [\beta]$. Sampling uniformly at random from a set is denoted by $\leftarrow\!\!\$$, appending by $\Leftarrow$, assignment by $\leftarrow$, equality by $=$, and concatenation by $|$. All our algorithms take as input the cryptographic parameters, params which include the groups that define the message and signature domains. In particular, they include a group $\mathbb{G}$ of prime order q where the DDH assumption holds, a hash function $\mathsf{H}_{\mathbb{G}}$ that maps binary strings to group elements, and a hash function H_q that maps binary strings to elements of $\mathbb{Z}_q$. For brevity, we omit explicitly including params in each algorithm description. We denote the ring, which is a list of credentials, as L, its length as $|L|$ and a particular credential as L_j. Since our protocol utilizes both an encryption and a signature scheme, we denote the encryption keys by $(\mathsf{pk}, \mathsf{sk})$ and the signature keys (which also double as voter credentials) by $(\mathsf{pc}, \mathsf{sc})$. We also use the dot notation to refer to items of a tuple.

Encryption. We require an IND-CPA-secure public key cryptosystem with homomorphic properties that can support distributed key generation and threshold decryption, like ElGamal [10]. We denote the encryption of a group element $\mathtt{m} \in \mathbb{G}$ with randomness $r \in \mathbb{Z}_q$ under public key pk as $\mathsf{Enc}_{\mathsf{pk}}(\mathtt{m}; r) = (g^r, \mathtt{m} \cdot \mathsf{pk}^r)$. A variation of our scheme may also use the exponential version of ElGamal, where $\mathtt{m} \in \mathbb{Z}_q$ and $g^{\mathtt{m}}$ is encrypted instead of $\mathtt{m}$. Given a ciphertext $c = (c_1, c_2)$, we denote its reencryption with randomness $r' \in \mathbb{Z}_q$ under the same public key pk as $\mathsf{ReEnc}_{\mathsf{pk}}(c, r') = c \odot \mathsf{Enc}_{\mathsf{pk}}(1; r') = (c_1 \cdot g^{r'}, c_2 \cdot \mathsf{pk}^{r'})$, where $\odot$ is the elementwise multiplication. Similarly, the notation c^z is used for (c_1^z, c_2^z).

Non-Interactive Zero-Knowledge Proofs of Knowledge. In verifiable e-voting schemes, the voters and the various authorities must provide proofs that they correctly executed the actions prescribed in the protocol without revealing their private inputs (e.g. candidate choices, secret keys or private credentials). Any interested party can then check these proofs and be convinced that the protocol was executed correctly.

We employ Σ-protocols for this task, which are public-coin 3-move interactive protocols ($\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Vrfy}$) where $\mathsf{NIZK.Prove}$ consists of the following steps: commitment by the prover (Com), challenge by the verifier and response by the prover (Resp). These protocols have the security properties of completeness, special-soundness and special honest-verifier zero-knowledge. The latter property implies the existence of a simulator Sim that creates accepting proofs for any public input. They can be made non-interactive (NIZK), and thus publicly verifiable, with the Fiat-Shamir transform [11]. We use its strong version by including all public parameters in the hash to avoid the attacks of [12, 13].

Most standard Σ-protocols in the literature originate from the proof of knowledge of a discrete logarithm π_S of Schnorr [14] and the proof of equality of discrete logarithm π_{CP} due to Chaum and Pedersen [15]. The NIZK proofs commonly utilized in e-voting schemes are:

- π_{Enc}: proof that the selected vote is a valid candidate encoding, which is constructed as a disjunction [16] of proofs that a ciphertext c corresponds to a known message $\mathtt{m}$. According to [12], the $\mathsf{Enc} + \mathsf{PoK}$ paradigm with an IND-CPA secure encryption scheme together with the strong Fiat-Shamir transform provides NM-CPA security.
- π_{Dec}: proof that a ciphertext $\mathsf{Enc}(\mathtt{m})$ has been correctly decrypted to $\mathtt{m}$ [3].
- π_{ReEnc}: proof that ciphertext c' correctly reencrypts ciphertext c.
- π_{sc}: proof that a user knows a secret credential x, y such that $c = \mathsf{Enc}_{\mathsf{pk}}(g^x h^y; r)$, constructed using a variation of the idea from [17].

In more detail, given ciphertexts $c = (c_1, c_2) = \mathsf{Enc}_{\mathsf{pk}}(\mathtt{m}; r) = (g^r, \mathtt{m} \cdot \mathsf{pk}^r)$ and $c' = \mathsf{ReEnc}(c) = (g^{r+r'}, \mathtt{m} \cdot \mathsf{pk}^{r+r'})$ we construct the proofs $\pi_{\mathsf{Enc}}, \pi_{\mathsf{Dec}}, \pi_{\mathsf{ReEnc}}$ as follows:

- If c is a correct encryption of $\mathtt{m}$ then the tuple $(g, \mathsf{pk}, c_1, c_2\mathtt{m}^{-1}) = (g, \mathsf{pk}, g^r, \mathsf{pk}^r)$ is a valid Diffie - Hellman tuple. This can be proved using π_{CP} of [15]. π_{Enc} can then be constructed as an OR-proof where the various messages are the encodings of the candidates in CS. A similar proof can used to that c correctly encrypts g^m in exponential ElGamal.
- If c correctly decrypts to $\mathtt{m}$ using sk then $(g, c_1, \mathsf{pk}, c_2\mathtt{m}^{-1}) = (g, g^r, g^{\mathsf{sk}}, c_1^{\mathsf{sk}})$ is a valid Diffie - Hellman tuple which can be also proved using π_{CP} of [15].
- If c' is a reencryption of c then $c' \odot c^{-1} = (g^{r'}, \mathsf{pk}^{r'})$ which can be proved using π_{CP} of [15].

A designated verifier proof [18], denoted by δ, receives as an additional input the public key of a 'legitimate' verifier. Thus, only this verifier can be convinced that the proof is valid, as they can simulate proofs using their private key. In our scheme, we utilize a designated verifier proof δ_{ReEnc} that c' is a correct reencryption of c due to [19].

The interactive versions of the proof of knowledge of secret credential corresponding to public credential (π_{sc}) and the designated verifier proof of correct reencryption (δ_{ReEnc}) can be found in Figure 1.

Plaintext Equivalence Test (Proof). An essential component of all JCJ-related schemes is a proof that two ciphertexts, encrypted with the same public key, hide the same plaintext. This proof is generated by a group of players, each holding shares of the decryption key, using the PET primitive from [20], i.e.

$$\mathsf{PET}(\mathsf{Enc}_{\mathsf{pk}}(\mathtt{m}_1); \mathsf{Enc}_{\mathsf{pk}}(\mathtt{m}_2)) = 1 \Leftrightarrow \mathtt{m}_1 \equiv \mathtt{m}_2$$

A PET can be instantiated in a distributed ElGamal setting using the techniques of [13, 20, 21]. Given two ciphertexts c_1, c_2, each player computes $c = c_1 \odot c_2^{-1}$.

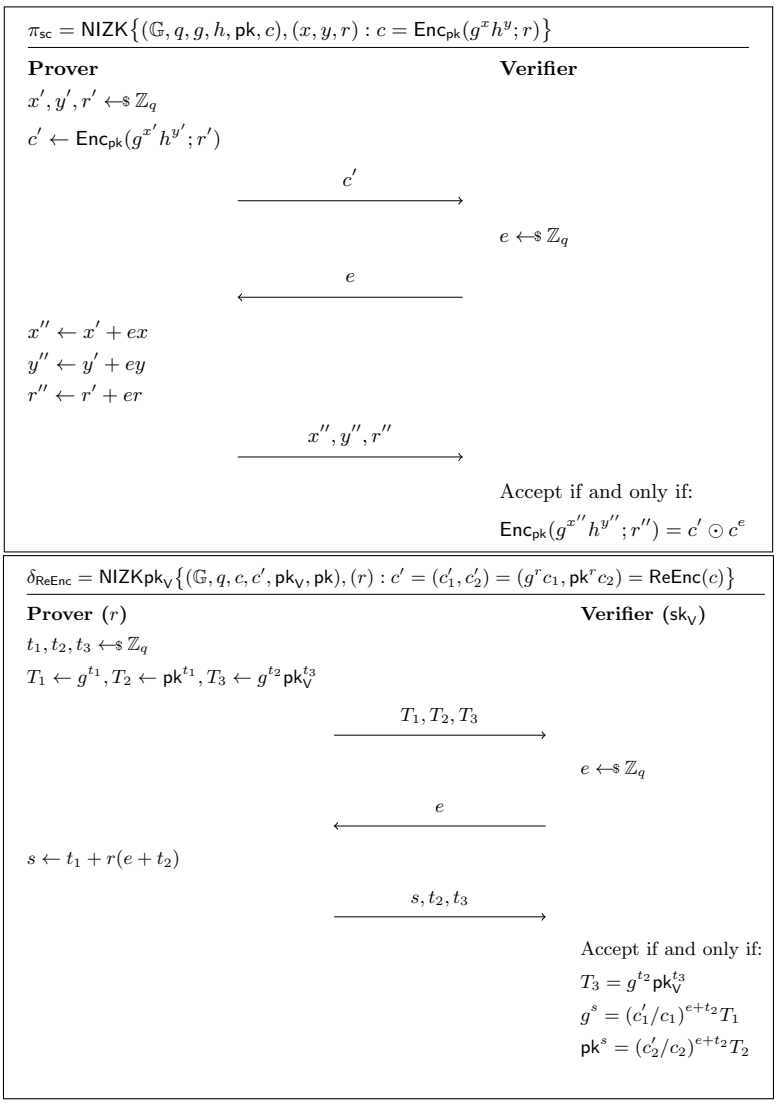

Fig. 1. Proofs π_{sc} and δ_{ReEnc}

The result is blinded by each, using a blinding factor z_i chosen uniformly at random, producing c^{z_i}. Then everyone commits to it and provides proofs π_{CP} of correct construction. After all commitments and proofs have been published and verified, every player multiplies all c^{z_i} together. The result $c' = \prod_i c^{z_i}$ is threshold decrypted and a proof π_{Dec} is generated. If c_1, c_2 indeed hide the same plaintext, the decryption yields 1, otherwise, a random group element. The proofs of decryption and correct construction shall be collectively denoted as π_{PET}. All the provisions of [13] (i.e. strong Fiat-Shamir transform, checks for trivial cases) must be followed for the PET to be a proof even if all parties are dishonest.

Verifiable shuffles. We require a functionality Shuffle that receives a list and anonymizes its contents to stop an attacker from tracing the processing of a specific ballot during the various stages of the tallying process. In our protocol, the items are the ballots and credentials of the voters. The Shuffle functionality is usually instantiated using permutations and reencryptions to alter both the position and the form of the ciphertexts. For verifiability, a NIZK proof of correct processing π_{Shuffle} is returned as an output as well. Verifiable shuffles are a well studied topic, and many such schemes exist. An implementation of our scheme could use a shuffle similar to [21].

3 A strongly linkable ring signature with unconditional anonymity

The proposed voting scheme is based on a new linkable ring signature (LRS) inspired from [8, 9]. Its syntax is presented in Definition 1 below.

Definition 1. *A* LRS *scheme is a tuple of algorithms* (Setup, KGen, Sign, Vrfy, Link)*:*

- $\mathsf{params} \leftarrow \mathsf{LRS.Setup}(\lambda)$. *Generates the system parameters.*
- $(\mathsf{sc}, \mathsf{pc}) \leftarrow \mathsf{LRS.KGen}()$. *Each user invokes the key generation algorithm ad-hoc which produces their secret and public credentials.*
- $\sigma \leftarrow \mathsf{LRS.Sign}(L, \mathtt{ev}, \mathsf{sc}, \mathtt{m})$. Sign *is the algorithm that a user with secret credential* sc *invokes in order to sign a message* m *for an event* ev*. The public credential* pc *corresponding to* sc *is contained in the ring* L*.*
- $0/1 \leftarrow \mathsf{LRS.Vrfy}(L, \mathtt{ev}, \mathtt{m}, \sigma)$ *is the public verification algorithm which outputs* 1 *if the signature is valid or* 0 *if it is not.*
- $0/1 \leftarrow \mathsf{LRS.Link}(\sigma_1, \sigma_2)$ *is the public linking algorithm which outputs* 1 *if valid signatures* σ_1 *and* σ_2 *originate from the same signer for the same event.*

The ev parameter serves as a unique election identifier in our e-voting protocol. It is produced by hashing the public election parameters (voting issue, candidate list, date etc.).

The security properties of our LRS and their relation to the security of our voting scheme are:

- **Unforgeability**: Only the holder of the signing key can produce valid signatures. This property enables verifiability and prevents double voting (in concert with linkability).
- **Linkable Anonymity**: The identity of the voter is hidden inside the ring, which serves as an anonymity set. Multiple votes from the same voter are linkable, but the voter's public credential is not revealed. This will help achieve (everlasting) privacy without sacrificing integrity.
- **Linkability**: Given two signatures created by the same ring member for the same event, the Link algorithm will always return 1. As the anonymity property can allow malicious voters to vote twice, the linkability property can make double voting detectable, without revealing the voter identity.
- **Non-slanderability**: Given a signature created by a ring member, no one can create a valid signature that is linked to it. This ensures that no one can update the vote of a voter except for the voter themselves.

We define our LRS in Figure 2. In order to participate, each player invokes the LRS.KGen function, which generates the pair of public and secret credentials. Then each player publishes pc. As a result, the ring in our construction comprises a public list of ElGamal encrypted values[1] of the form:

$$L = \left(\mathsf{pc}_1, \ldots, \mathsf{pc}_n\right) = \left(\mathsf{Enc}_{\mathsf{pk}}(g^{x_1} h^{y_1}; r_1), \ldots, \mathsf{Enc}_{\mathsf{pk}}(g^{x_n} h^{y_n}; r_n)\right)$$

To create a signature, a signer that knows the secret credential corresponding to an item L_i invokes LRS.Sign. The secret credential is a tuple consisting of the secrets x_i, y_i and the randomness r_i that was used to create L_i. In essence, the signature is a NIZK proof of knowledge of this secret credential. Note that the ring creation is ad-hoc, as each signer can select a subset of the published credentials at will. To verify the signature, everyone can invoke the algorithm LRS.Vrfy. Finally, to check if two signatures are linked, everyone can invoke LRS.Link, which checks if both signatures verify correctly and contain the same linking tag.

3.1 Security Analysis

We now analyze the security properties of our LRS scheme.

Theorem 1 (Unforgeability). *Our* LRS *has the property of unforgeability in the random oracle model, as defined in [9], given that* DLOG *is hard.*

Proof. Assume a PPT adversary $\mathcal{A}$ that with non-negligible probability can forge a signature σ that passes the verification without knowledge of any of the secret credentials. We will construct an algorithm $\mathcal{B}$ that given n DLOG instances

[1] Our signature can be implemented absent an encryption scheme without sacrificing any of its security properties. Encryption in our construction is used to easily integrate the PET functionality later in the voting scheme (cf. section 4)

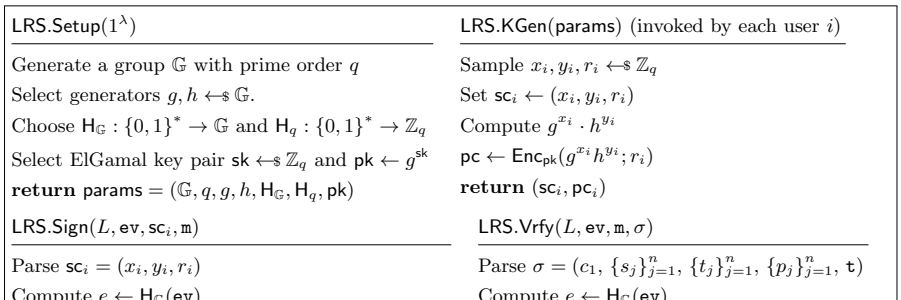

Fig. 2. Our LRS construction

$\{X_i\}_{i=1}^n$ and by using $\mathcal{A}$ outputs the discrete logarithm of at least one of them with non-negligible probability.

The input of $\mathcal{B}$ is $\mathbb{G}, g, q, \{X_i\}_{i=1}^n$ and $\mathcal{B}$ simulates the environment for $\mathcal{A}$. $\mathcal{A}$ may query the random oracle $\mathcal{RO}$, to receive a random element of $\mathbb{Z}_q$ or of $\mathbb{G}$, the joining oracle $\mathcal{JO}$, to add users and their public credentials to the system, the corruption oracle $\mathcal{CO}$, where $\mathcal{A}$ may ask for the secret credential that corresponds to a public credential and lastly the signing oracle $\mathcal{SO}$, where $\mathcal{A}$ receives a signature on behalf of a specific signer. $\mathcal{B}$ generates the system parameters and simulates the oracles that $\mathcal{A}$ has access to. Upon query of the $\mathcal{RO}$, $\mathcal{B}$ returns a random value and replies consistently to all the queries. Upon query of the $\mathcal{JO}$, $\mathcal{B}$ adds public credentials to the system, either by adding $\mathsf{pc}_i \leftarrow (g^{r_i}, X_i h^{y_i} \mathsf{pk}^{r_i})$ using the given DLOG challenges and $y_i, r_i \leftarrow\$ \mathbb{Z}_q$ or by adding $\mathsf{pc}_i \leftarrow (g^{r_i}, g^{x_i} h^{y_i} \mathsf{pk}^{r_i})$, using random values $x_i, y_i, r_i \leftarrow\$ \mathbb{Z}_q$. Furthermore, $\mathcal{A}$ may query the $\mathcal{CO}$ for a public credential, where $\mathcal{B}$ replies with the corresponding private credential and lastly $\mathcal{A}$ may query the $\mathcal{SO}$, where $\mathcal{B}$ programs the random oracle and replies with a signature created on behalf of the queried signer on the queried message.

Let σ_0^* be a forgery that $\mathcal{A}$ produced on the list $\{(g^{r_i}, X_i h^{y_i} \mathsf{pk}^{r_i})\}_{i=1}^n$ of public credentials. We can assume that $\mathcal{A}$ queried all n queries used in the Vrfy algorithm. $\mathcal{B}$ rewinds $\mathcal{A}$, replies consistently to all hash queries but at the hash query where they replied c_{j0} they reply $c_{j1} \neq c_{j0}$. From the rewind-on-success lemma [9] $\mathcal{A}$ will produce another forgery σ_1^* with non-negligible probability, for which $c_{i0} = c_{i1}$, $\forall i \in \{1, \dots n\} \setminus \{j\}$. Therefore for the two forgeries, since they both verify correctly, it holds that:

$$K_{j0} = K_{j1} \Rightarrow g^{t_{j0}} \cdot h^{p_{j0}} \cdot \mathsf{pk}^{s_{j0}} \cdot (L_{j2})^{c_{j0}} = g^{t_{j1}} \cdot h^{p_{j1}} \cdot \mathsf{pk}^{s_{j1}} \cdot (L_{j2})^{c_{j1}} \tag{1}$$

There exist x'_j, y'_j, $r'_j \in \mathbb{Z}_q$ s.t. $L_{j2} = X_j h^{y'_j} \mathsf{pk}^{r'_j} = g^{x'_j} \cdot h^{y'_j} \cdot \mathsf{pk}^{r'_j}$. Then by Eq. 1:

$$g^{t_{j0}} \cdot h^{p_{j0}} \cdot \mathsf{pk}^{s_{j0}} (g^{x'_j} \cdot h^{y'_j} \cdot \mathsf{pk}^{r'_j})^{c_{j0}} = g^{t_{j1}} \cdot h^{p_{j1}} \cdot \mathsf{pk}^{s_{j1}} (g^{x'_j} \cdot h^{y'_j} \cdot \mathsf{pk}^{r'_j})^{c_{j1}}$$

$$g^{t_{j0} + c_{j1} x'_j} \cdot h^{p_{j0} + c_{j1} y'_j} \cdot \mathsf{pk}^{s_{j0} + c_{j1} r'_j} = g^{t_{j1} + c_{j1} x'_j} \cdot h^{p_{j0} + c_{j1} y'_j} \cdot \mathsf{pk}^{s_{j1} + c_{j1} r'_j}$$

$$x'_j = \frac{t_{j0} - t_{j1}}{c_{j1} - c_{j0}}, \quad y'_j = \frac{p_{j0} - p_{j1}}{c_{j1} - c_{j0}}, \quad r'_j = \frac{s_{j0} - s_{j1}}{c_{j1} - c_{j0}} \tag{2}$$

Thus $\mathcal{B}$ solved the DLOG problem for one of the given challenges with non-negligible probability, since x'_j is the discrete logarithm of X_j.

Theorem 2 (Linkable Anonymity). *Our* LRS *has perfectly unconditional linkable anonymity, according to the definition of [22].*

Proof. In the anonymity experiment $\mathsf{Exp}_{\mathsf{LRS-Anon}}(\mathcal{A})$ of [22, 23], the challenger runs the setup and key generation algorithms, to create the system parameters and $\mathsf{poly}(\lambda)$ pairs of secret and public credentials. Let $\mathcal{P}$ be the set of all generated public credentials. Furthermore, the challenger chooses uniformly at random a bit $b \leftarrow\$ \{0, 1\}$. Then $\mathcal{A}$, upon receiving the system parameters and $\mathcal{P}$, outputs a set of public credentials $\mathcal{C} \subset \mathcal{P}$ it wants to corrupt. The challenger returns to $\mathcal{A}$ all the secret credentials that correspond to the public credentials in $\mathcal{C}$. The adversary may request for signatures on messages of its choice for any ring that contains public credentials in $\mathcal{P}$. Then it outputs two challenge public credentials $\mathsf{pc}_{i_0}, \mathsf{pc}_{i_1} \in \mathcal{P} \setminus \mathcal{C}$ (i.e. that are not corrupted) for which no signatures have been requested. $\mathcal{A}$ can also query for signatures on behalf of signer i_b or signer i_{1-b} or any other signer in $\mathcal{P} \setminus \mathcal{C}$, for any given message and ring $L \subseteq \mathcal{P}$. The adversary wins the game if they output a bit $b^* \in \{0, 1\}$ such that $b^* = b$.

To prove linkable anonymity in the honest key model, we will create two hybrid games $\mathcal{H}^{(0)}, \mathcal{H}^{(1)}$, where the signatures created by signers i_b and i_{1-b} are swapped and this change is indistinguishable by the adversary. In $\mathcal{H}^{(0)}$, which is the initial game $\mathsf{Exp}_{\mathsf{LRS-Anon}}(\mathcal{A})$, the signatures returned on behalf of signer i_b, i.e:

$$\sigma_b^{(0)} \leftarrow \mathsf{LRS.Sign}\,(L, \mathsf{ev}, (x_{i_b}, y_{i_b}, r_{i_b}), \mathtt{m})$$

correspond to the public credential $\mathsf{pc}_{i_b} = (g^{r_{i_b}}, g^{x_{i_b}} h^{y_{i_b}} \mathsf{pk}^{r_{i_b}})$ and the signatures returned on behalf of signer i_{1-b}, i.e.:

$$\sigma_{1-b}^{(0)} \leftarrow \mathsf{LRS.Sign}\left(L, \mathsf{ev}, (x_{i_{1-b}}, y_{i_{1-b}}, r_{i_{1-b}}), \mathtt{m}\right)$$

Assume now that $h = g^k$ for some $k \in \mathbb{Z}_q$.

In $\mathcal{H}^{(1)}$, the challenger selects $x'_{i_b} = x_{i_b}$ and $y'_{i_b} = \frac{x_{i_{1-b}} - x_{i_b} + y_{i_{1-b}} \cdot k}{k}$ and similarly sets $x'_{i_{1-b}} = x_{i_{1-b}}$ and $y'_{i_{1-b}} = \frac{x_{i_b} - x_{i_{1-b}} + y_{i_b} \cdot k}{k}$. Now the requests for signatures on behalf of signer i_b are answered using

$$\sigma_b^{(1)} \leftarrow \mathsf{LRS.Sign}\left(L, \mathtt{ev}, (x'_{i_b}, y'_{i_b}, r_{i_{1-b}}), \mathtt{m}\right)$$

which correspond to the public credential $\mathsf{pc}_{i_{1-b}} = (g^{r_{i_{1-b}}}, g^{x_{i_{1-b}}} h^{y_{i_{1-b}}} \mathsf{pk}^{r_{i_{1-b}}})$ and similarly the requests for signatures on behalf of signer i_{1-b} are answered using

$$\sigma_{1-b}^{(1)} \leftarrow \mathsf{LRS.Sign}\left(L, \mathtt{ev}, (x'_{i_{1-b}}, y'_{i_{1-b}}, r_{i_b}), \mathtt{m}\right)$$

where $x'_{i_{1-b}} = x_{i_{1-b}}$ and $y'_{i_{1-b}} = \frac{x_{i_b} - x_{i_{1-b}} + y_{i_b} \cdot k}{k}$, which correspond to the public credential $\mathsf{pc}_{i_b} = (g^{r_{i_b}}, g^{x_{i_b}} h^{y_{i_b}} \mathsf{pk}^{r_{i_b}})$.

The signatures $\sigma_b^{(0)}$ and $\sigma_b^{(1)}$ are indistinguishable even from an unbounded adversary. Indeed, given that: $\sigma_b^{(0)} = (c_1^{(0)}, \{s_j^{(0)}\}_{j=1}^n, \{t_j^{(0)}\}_{j=1}^n, \{p_j^{(0)}\}_{j=1}^n, \mathtt{t}_b^{(0)})$ and $\sigma_b^{(1)} = (c_1^{(1)}, \{s_j^{(1)}\}_{j=1}^n, \{t_j^{(1)}\}_{j=1}^n, \{p_j^{(1)}\}_{j=1}^n, \mathtt{t}_b^{(1)})$ observe that $\mathtt{t}_b^{(1)} = \mathtt{t}_b^{(0)} = e^{x_{i_b}}$ and that all the other elements of the signatures are uniformly distributed in $\mathbb{Z}_q$ given that x_{i_b}, y_{i_b} are uniformly distributed in $\mathbb{Z}_q$ for all values of b. A similar indistinguishability argument holds for signatures $\sigma_{1-b}^{(0)}$ and $\sigma_{1-b}^{(1)}$.

Thus the transition from $\mathcal{H}^{(0)}$ to $\mathcal{H}^{(1)}$ is indistinguishable and

$$|\Pr\left[\mathcal{H}^{(0)}(\mathcal{A}) = 1\right] - \Pr\left[\mathcal{H}^{(1)}(\mathcal{A}) = 1\right]| = 0$$

which implies that

$$\mathsf{Adv}(\mathcal{A}) = |\Pr\left[\mathsf{Exp}_{\mathsf{LRS-Anon}}(\mathcal{A}) = 1\right] - \frac{1}{2}| = 0$$

and hence our scheme has perfect unconditional anonymity.

Theorem 3 (Linkability). *Our* LRS *has the property of strong linkability in the random oracle model, as defined in [24], if* DLOG *is hard.*

Proof. Assume a PPT adversary $\mathcal{A}$ that owns $k-1$ secret credentials and with non-negligible probability can produce k pairwise unlinkable signatures. We will construct an algorithm $\mathcal{B}$ that given n DLOG instances $\{X_i\}_{i=1}^n$ outputs the discrete logarithm of at least one of them with non-negligible probability.

$\mathcal{B}$ simulates the environment and the oracles for $\mathcal{A}$ as described in the proof of unforgeability in Theorem 3.1. Let $\{\sigma_i^*\}_{i=1}^k$ be the set of k pairwise unlinkable signatures that $\mathcal{A}$ produced. We assume that $\mathcal{A}$ queried all queries used in the Vrfy algorithm.

Case 1: $\mathcal{A}$ produced at least 2 signatures σ_a and σ_b that had as last queries l_a and l_b respectively for the same index j. Then $\mathcal{B}$ rewinds $\mathcal{A}$ twice, on the l_a and l_b query respectively, replies consistently to all hash queries but at l_a

replies c_{ja1} instead of c_{ja0} and at l_b replies c_{jb1} instead of c_{jb0}. By the rewind on success lemma [9], $\mathcal{A}$ will produce a forgery σ^*_{a1} on the first rewind and σ^*_{b1} on the second rewind with non-negligible probability, for which similarly with the unforgeability proof it holds that:

$$K''_{ja0} = K''_{ja1} \Rightarrow e^{t_{ja0}} \cdot \mathtt{t}_a^{c_{ja0}} = e^{t_{ja1}} \cdot \mathtt{t}_a^{c_{ja1}} \overset{\mathtt{t}_a = e^{x'_a}}{\Rightarrow} x'_a = \frac{t_{ja0} - t_{ja1}}{c_{ja1} - c_{ja0}} \tag{3}$$

$$K''_{jb0} = K''_{jb1} \Rightarrow e^{t_{jb0}} \cdot \mathtt{t}_b^{c_{jb0}} = e^{t_{jb1}} \cdot \mathtt{t}_b^{c_{jb1}} \overset{\mathtt{t}_b = e^{x'_b}}{\Rightarrow} x'_b = \frac{t_{jb0} - t_{jb1}}{c_{jb1} - c_{jb0}} \tag{4}$$

By eqs. (2) and (3) we have that $x'_a = x'_j$ and by eqs. (2) and (4) we have that $x_b = x'_j$. Therefore $\mathsf{LRS.Link}(\sigma^*_a) = \mathsf{LRS.Link}(\sigma^*_b) = e^{x'_j}$, hence σ_a and σ_b are not unlinkable.

Case 2: All signatures produced by $\mathcal{A}$ had distinct index j as the last query. $\mathcal{B}$ does k rewind simulations on each of the last queries of each signature, similarly with the unforgeability proof. Since $\mathcal{A}$ knows only $k-1$ secret credentials, $\mathcal{B}$ solves the DLOG problem for at least one of the challenges $\{X_i\}_{i=1}^n$ with non-negligible probability.

Theorem 4 (Non-slanderability). *Our* LRS *has the property of non-slanderability, in the random oracle model as defined in [8], given that* DLOG *is hard.*

The theorem is implied by the unforgeability and strong linkability of our LRS construction.

4 A Voting Scheme using Linkable Ring Signatures

We now present a formal definition of our voting protocol.

Syntax A voting scheme consists of the following entities:

- a set of n eligible voters $\mathsf{V} = \{\mathsf{V}_1, \ldots, \mathsf{V}_n\}$ identified by their index in V.
- a set of k candidates $\mathsf{CS} = \{\mathsf{cnd}_1, \cdots \mathsf{cnd}_k\}$,
- the registration authority, (RA), distributed among a set of n_{RA} registrars,
- the tallying authority, (TA), consisting of n_{TA} talliers,
- A bulletin board (BB) which can be considered as an authenticated append-only ledger. It contains all public election data. For clarity, we split the BB into sections named after the corresponding phases of the voting protocol.

In an implementation, the voters would be represented by the *voting clients*, which combine software and hardware components to handle cryptographic operations. Clients for JCJ-related schemes require the extra functionality of generating fake credentials, implemented either in the client or by using the more user-friendly mechanism of panic passwords [25]. We also assume an election supervisor (EA) to invoke the various server-side functionalities. The EA is involved with maintenance procedures (where no cryptographic keys are utilised), so it is considered honest.

Definition 2. *A voting scheme is a tuple* $\mathsf{VS} = (\mathsf{Setup}, \mathsf{Register}, \mathsf{SetupElection}, \mathsf{Vote}, \mathsf{IsValid}, \mathsf{Shuffle}, \mathsf{Tally}, \mathsf{Vrfy})$ *where:*

- $(\mathsf{params}, \mathsf{pk}, (\mathsf{pk}_i)_{i=1}^{n_{\mathsf{TA}}}, (\mathsf{sk}_{\mathsf{TA}_i})_{i=1}^{n_{\mathsf{TA}}}) \leftarrow \mathsf{VS.Setup}(\lambda)$*, is a PPT algorithm that generates the cryptographic parameters of the voting scheme.*
- $((pc'_i, \delta_{\mathsf{ReEnc}}), (i, \mathsf{pk}_i, \mathsf{pc}_i)) \leftarrow \mathsf{VS.Register}\langle \mathsf{V}_i(), \mathsf{RA}() \rangle$*, is a protocol that allows voter* i *to generate their keys and credentials and submit their public components to the registration authority. The* RA *reencrypts the public credentials and sends it to the voter together with a designated verifier proof of correct re-encryption proof* δ_{ReEnc}*.*
- $(L, \mathsf{CS}, \mathsf{ev}) \leftarrow \mathsf{SetupElection}()$ *is executed by the* EA *to generate configuration data for a specific election and return the public credential list for eligible voters.*
- $b_i \leftarrow \mathsf{VS.Vote}(\mathsf{cnd}, \mathsf{sc}_i, i, \mathsf{ev})$*, is executed by each voter* V_i *to generate the ballot* b_i*, given the election* ev*, the candidate choice* cnd *and the secret credential of* V_i*.*
- $0/1 \leftarrow \mathsf{VS.IsValid}(b, \mathsf{BB})$ *is an algorithm executed by the* BB*. It returns* 1 *if the ballot* b *can be added to the* BB*.*
- $\pi_{\mathsf{Shuffle}} \leftarrow \mathsf{VS.Shuffle}(\mathsf{BB})$ *is an algorithm invoked by the* EA *to shuffle the ballots in the* BB *and returns a proof of correct operation.*
- $(T, \pi) \leftarrow \mathsf{VS.Tally}(\mathsf{sk}_{\mathsf{TA}}, \mathsf{BB})$*, executed by the* TA*, outputs the result of the election* T*, together with proof(s)* π *of the correctness of* T*.*
- $0/1 \leftarrow \mathsf{VS.Vrfy}(\mathsf{BB}, T, \pi)$ *allows anyone to verify the election result.*

Our construction

Our proposed voting scheme is depicted in Figure 3 and described next.

Setup. In the setup phase the EA initiates a variation of the $\mathsf{LRS.Setup}$ algorithm. The difference from the one in Figure 2 is that a distributed key generation phase is executed. Consequently the TA secret key $\mathsf{sk}_{\mathsf{TA}}$ is split into n_{TA} parts $\mathsf{sk}_{\mathsf{TA}_i}$ with public counterparts $\mathsf{pk}_{\mathsf{TA}_i}$. The joint TA public key is pk. The returned values $\mathsf{params}, \mathsf{pk}_{\mathsf{TA}_i}, \mathsf{pk}$ are posted on the BB.

Registration. The registration phase consists of the $\mathsf{VS.Register}$ protocol and the $\mathsf{VS.SetupElection}$ algorithm. The former is executed by each voter in concert with the RA and its objective is to enable them to create and enroll their real credentials. Each voter V_i executes $\mathsf{LRS.KGen}$ to compute $(\mathsf{pc}_i, \mathsf{sc}_i)$, where sc_i is the real credential that will be used in the voting phase to indicate that the vote reflects the will of the voter. Any other value presented during the voting phase will indicate coercion. The public credential is encrypted using pk. The voter sends their identity i to the RA along with their public credential pc_i and a proof of knowledge π_{sc} of sc_i to avoid replay attacks. The RA reencrypts it as $\mathsf{pc}'_i \leftarrow \mathsf{ReEnc}(\mathsf{pc}_i, r'_i)$, and provides a designated-verifier proof of correct reencryption (δ_{ReEnc}) that does not reveal the randomness used. To create this

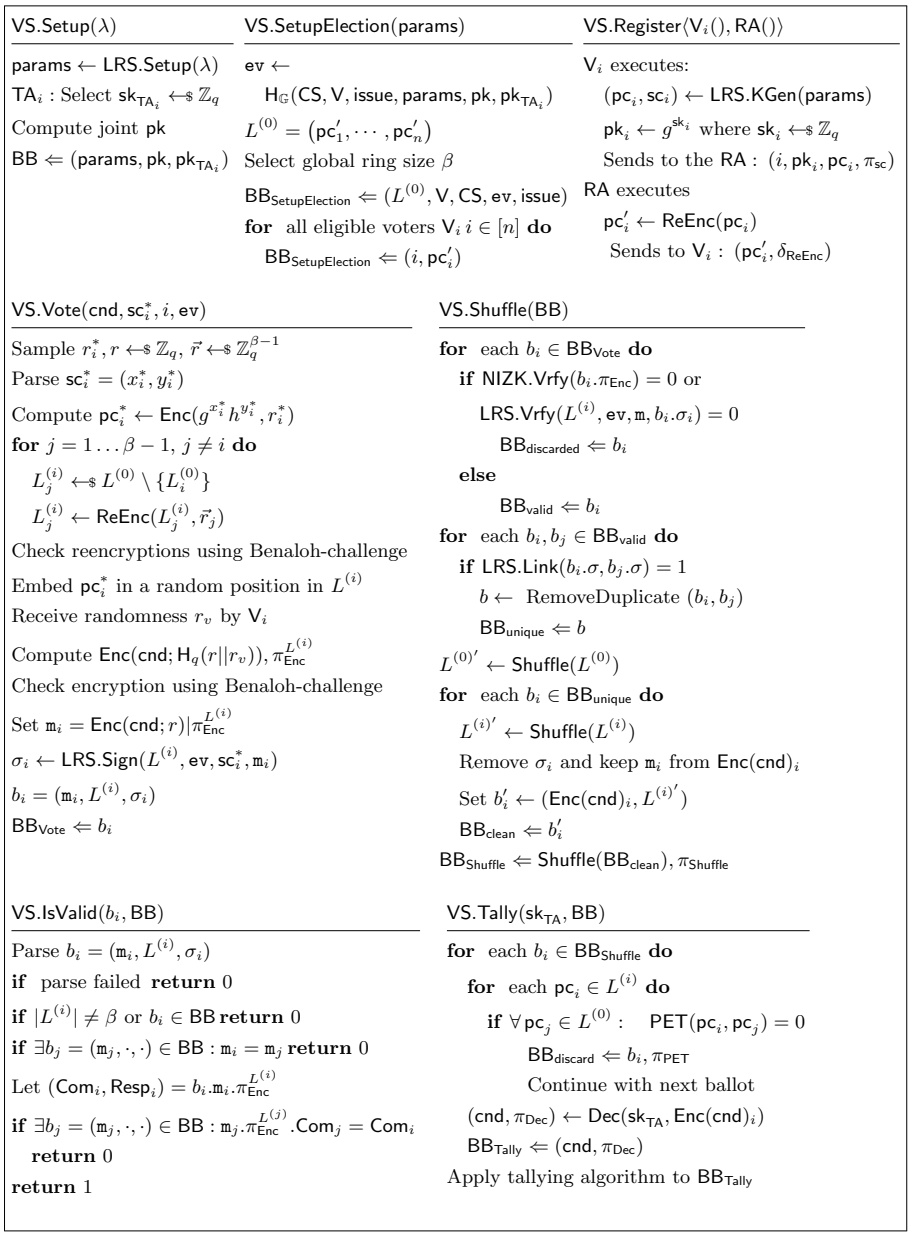

Fig. 3. Our VS construction

proof, every voter must have a public key pk_i which they generate themselves and send to the RA during this registration phase. Using its secret counterpart, sk_i, will allow the voter to fake the proof δ_{ReEnc} for the coercer. After all the voters have registered, and a specific election begins, the RA filters only the eligible

ones V of size n. The EA executes the $\mathsf{SetupElection}$ algorithm to produce the list of candidates CS, and $\mathtt{ev} = \mathsf{H}_{\mathbb{G}}(\mathsf{CS}, \mathsf{V}, \mathsf{issue}, \mathsf{params}, \mathsf{pk}, \mathsf{pk}_{\mathsf{TA}_i})$ as the election identifier. Note that issue denotes a description of the question that the particular election aims to settle. The election authority also selects the global size of the anonymity set, denoted by β. This will be the size of the ring for each signature that comes with a ballot. If a ballot contains a ring of different size it will be rejected as it could be used as a tag to enable coercion (cf. section 5.4) Then the EA posts to the BB the public credentials pc'_i along with the identities of the voters. Thus, after the registration phase, the BB contains the list of (reencrypted) public credentials of the n eligible voters $L^{(0)} = (\mathsf{pc}'_1, \ldots, \mathsf{pc}'_n)$, where each voter only knows their own secret credential sc_i. Since the RA has reencrypted the credentials, V_i does not know the final randomness used in encrypting pc_i.

Voting. In order to generate a vote for candidate cnd, the voter invokes the $\mathsf{VS.Vote}(\mathsf{cnd}, \mathsf{sc}^*_i, i, \mathtt{ev})$ algorithm, where sc^*_i is their secret credential, real or fake.

Cast-as-intended verifiability can be provided through a cut-and-choose mechanism, like the Benaloh-challenge [26]; the voter selects their candidate of choice and then the voting client prepares the ballot. The voter is given the option to audit or cast it. In the former case, the client reveals the randomness and other parameters used to create the ballot, so that the voter can externally replicate its correct creation. An audited ballot is never cast. In order to avoid clash attacks, our scheme generates the randomness used to encrypt the candidate choice in cooperation with the voter [27].

When under coercion, the voter obeys the coercer's instructions but votes using a fake credential sc^*_i generated on the fly by the client. During the moment of privacy, the voter uses the credential created during the registration phase, i.e. $\mathsf{sc}^*_i = \mathsf{sc}_i$ and votes as they wish. In both cases, the voter encrypts sc^*_i to obtain pc^*_i. Afterwards, they select in a random order $\beta - 1$ more credentials from $L^{(0)}$ (found in the BB). These credentials are reencrypted and serve as an anonymity set. In order to check that the voting client correctly reencrypted the credentials, the Benaloh challenge mechanism is employed again. For usability, the audit or cast challenge is performed for all $\beta - 1$ credentials and not for each one individually. Finally, the voter embeds pc^*_i in a random position among the $\beta - 1$ decoys. As a result, a new list $L^{(i)}$ containing β credentials is produced by every voter. Then they encrypt their candidate choice as $\mathsf{Enc}_{\mathsf{pk}}(\mathsf{cnd}; r)$ and produce a proof $\pi^{L^{(i)}}_{\mathsf{Enc}}$ that cnd is a valid candidate choice from CS and that it is encrypted correctly. This proof is made non-interactive using the strong Fiat-Shamir heuristic, where the input to the hash function also contains the actual ring $L^{(i)}$ used during signature generation. Thus, the hash call contains the full statement and as a result the ballot is not malleable [12]. In the end, the voters invoke the $\mathsf{LRS.Sign}$ algorithm to sign $\mathtt{m}_i = \mathsf{Enc}_{\mathsf{pk}}(\mathsf{cnd}; r) \mid \pi^{L^{(i)}}_{\mathsf{Enc}}$. The signature also serves as a proof of knowledge of the secret credential sc_i. The ballot returned from $\mathsf{VS.Vote}$ consists of $(\mathtt{m}_i, L^{(i)}, \sigma)$ and is appended to the BB.

Ballot weeding. The VS.IsValid functionality, which may be publicly invoked, checks the well-formedness of a ballot and protects against replay attacks by making sure that:

- the format of the ballot is the one specified from VS.Vote.
- the signature and proofs verify.
- $L^{(i)}$ contains exactly β items.
- there does not exist an exact copy of the ballot currently in the BB.
- there does not exist a duplicate of $\mathtt{m}_i$ in BB.
- the commitments (inputs to the hash function) for $\pi_{\mathsf{Enc}}^{L^{(i)}}$ cannot be found in any other such proof in the BB.

Mixing. When the voting phase has concluded, the EA discards invalid ballots and invokes LRS.Link to see if there are votes that were cast using the same credential. If such are found, only one is kept according to a pre-specified policy (e.g. the one that was cast last). Then the EA initiates shuffling so that the coercer loses track of their ballot and the position of their credential inside the credential list. Initially the list $L^{(0)}$ is shuffled. For each ballot b_i, only the encrypted voter choice Enc(cnd) and the encrypted credential list $L^{(i)}$ are kept. The latter is shuffled to prevent the use of the credential list as a tag (horizontal shuffle). Subsequently, the list of all the ballots is shuffled again (vertical shuffle).

Tally. The objective of the tally phase is to verifiably remove coerced ballots and to count only the ones that were cast using genuine credentials. To this end, the members of the TA utilize the PET functionality. In particular, for each ballot b_i, each element of the list $L^{(i)}$ is compared with all the elements of the initial list $L^{(0)}$. If each of the credentials in $L^{(i)}$ reencrypts some element in $L^{(0)}$, it means that b_i was cast with the correct credentials and the choice of the voter must be counted. If, on the other hand, there exists an element of $L^{(i)}$ that does not correspond to an element of $L^{(0)}$, then the ballot is considered a product of coercion and thus discarded. Since during the mixing phase both the order of the ballots and the order of the credential list were shuffled the coercer cannot tell if their own ballot was discarded or not.

From this stage on, two types of tallying may be applied. All the acceptable ciphertexts may be homomorphically combined - if supported by the encryption scheme and the election counting function. Then the talliers will jointly decrypt the 'aggregate ballot' and announce the election result along with a proof of correct decryption π_{Dec}. Alternatively, each Enc(cnd) corresponding to an accepted ballot may be decrypted along with a proof π_{Dec} - for conciseness we depict only this case in Figure 3. After all ballots have been decrypted the result calculation function can be applied to the plaintexts corresponding to each accepted ballot. Finally, the election result may be announced. This option is more general, as it can be employed in protocols with elaborate vote counting functions.

Any interested party can use VS.Vrfy to confirm the result by checking the proof $\pi_{\mathsf{Enc}}^{L^{(i)}}$ and verifying the signature in each ballot contained in the $\mathsf{BB}_{\mathsf{Vote}}$

section of the bulletin board. They can also check the proofs for all PETs in $\mathsf{BB}_{\mathsf{discard}}$ and $\mathsf{BB}_{\mathsf{Tally}}$ and all the proofs of correct ballot decryption π_{Dec}. Finally, they can recompute on their own the result function for the voters' choices.

Performance. The voters perform $\beta - 1$ reencryptions, 1 encryption and 1 sign operation which depends on the size of $L^{(i)}$. Hence, the complexity of VS.Vote is $\mathcal{O}(\beta)$. After voting has finished, assume that there are $m > n$ ballots in the BB, since voters may vote multiple times to evade coercion or because they are following the instructions of the coercer. Before shuffling, the validity checks for all ballots require $\mathcal{O}(\beta m)$ time, while discovering duplicates requires $\mathcal{O}(m)$ by sending the linking tags through a hashtable [28]. Assuming shuffling is implemented by reencryption and sorting the results as binary strings, it takes $\mathcal{O}(m \log m)$ time. During tallying, the n credentials in $L^{(0)}$ participate in PETs with the β credentials contained in each of the m ballots. Consequently, tallying takes $\mathcal{O}(\beta m n)$ time. Assuming β is constant, our scheme is of quadratic complexity, similar to most JCJ-related schemes. However, our scheme is also 'vote-and-go' and provides everlasting privacy. These benefits counterbalance in our view the quadratic cost, which can be further managed by partitioning a large voting population to β-sized groups, in a manner similar to precincts in non-electronic elections.

5 Security Analysis

We now formally analyze the security properties of our voting scheme. For each security property, we first state the intuition for our argument and then we provide the formal proof.

5.1 Assumptions

For all security properties we assume a BB that will not delete or reject ballots. The voting client is trusted for all security properties *except* verifiability.

For verifiability, the TA is not trusted and the adversary may corrupt some voters. To prove strong verifiability [29], we require that the BB and the RA are not simultaneously corrupted, in the sense that the BB will not stuff ballots *and* the RA will not handle credentials in a malicious way.

For (everlasting) privacy we assume that the adversary has access to all messages posted by the voters in all phases of the protocol and that all the tallying and registration authorities might be corrupted. We also trust the client not to reveal the voting choice and randomness used. We also allow the adversary to corrupt some voters and use them to cast arbitrary ballots (e.g. for replay attacks).

For coercion resistance, on top of the last two assumptions, we also trust the voting client to generate fake credentials during the coercion attack that are indistinguishable from the registered ones. We also assume that the coercer does not have full control over a voter during the entirety of the election. If

such were the case, they essentially *would become the voter* [21]. In particular, during voting, the voter is allowed a moment of privacy to cast their preference using their real credential. Furthermore, during registration, the adversary cannot simulate a voter. In practice, this can be implemented by trusting the RA like in Selections [25]. Alternatively, we may accept the weaker assumption of Civitas [21] that the registration authority is distributed and the voter is aware of the identity of at least one honest RA member, with which they are connected through an untappable channel. Hence, they can simulate their transcript with the latter to the coercer, through designated-verifier proofs, in order to prove the validity of *any* secret credential. This presumes that the voter has a registration key pair which is only used for this proof, but nowhere else. While the use of an untappable channel contradicts the idea of remote voting, the 'inconvenience' can be mitigated by reusing the credentials in many elections through a 'rebasing' mechanism (e.g. by changing the group generator from g to g^a, where a is randomly selected for each new election). We also accept (as in the uncertainty assumption of JCJ [7]) that the adversary does not have complete knowledge of all the honest voters' behavior (including the revoting patterns for both coerced and non-coerced voters) during the election, which is necessary for the coercer to be able to distinguish the real from the ideal game. This uncertainty can be implemented in practice by allowing some external entities (e.g. non-governmental organizations) to cast decoy ballots. Finally, we require at least one honest TA member for coercion resistance.

5.2 Verifiability

Our scheme satisfies election verifiability [29], which includes individual and universal verifiability. The definition of this property in [29] does not include eligibility verifiability, which our scheme does not satisfy. If it did, *everyone* would be able to check that each successfully tallied ballot was cast by a voter with a correct credential [30]. But this could be used to enable coercion, as the attacker would be able to discover if the credential in a targeted ballot was fake or not. In our scheme, however, eligibility verifiability holds globally, i.e. everyone can verify that *all* tallied votes were cast by voters with the right to participate and the corresponding correct credential, without being able to isolate specific ballots. To intuitively see why our scheme satisfies individual and universal verifiability, we examine how it fares against some well-known attacks.

Individual Verifiability. An adversary $\mathcal{A}$ cannot create a clash attack nor can they invalidate a ballot without the voter finding out, assuming that they contribute to the randomness required by the cryptographic functionalities [30]. Firstly, the RA cannot assign the same credential to two distinct *honest* voters as they generate it on their own using $\mathsf{LRS.KGen}$. Also, the credentials cannot be invalidated by a dishonest RA. The soundness of the designated verifier proof (δ_{ReEnc}) proves that their reencryption was correctly applied. During voting, the Benaloh challenge along with the fact that the voter contributes to the encryption randomness prevents the system from substituting the preferred candidate

and from casting duplicate and invalid ballots. Note that the coercer cannot use the randomness generated by VS.Vote as a receipt to find out if their attack succeeded or not, since contrary to Helios ([31]) not every ballot in the BB is counted in JCJ-compliant schemes. When the ballot is cast, this receipt will be valid for the coercer. However, the ballot will be discarded during tallying since the PET will fail due to the fake credential. The mixing phase in between will prevent the coercer from tracking the ballot from the cast to the tally phase.

Universal Verifiability The architecture of our scheme does not allow the adversary to alter or drop votes belonging to honest voters or to stuff ballots:

- Ballot altering: $\mathcal{A}$ cannot alter a ballot after vote casting (by changing either the candidate selection or the credential list), since they would have to produce an honest signature for the altered ballot. This is prevented by the unforgeability of our LRS. Additionally, $\mathcal{A}$ may try to alter a ballot during mixing, but this is averted by the soundness of the Shuffle proofs. Finally, during tallying, they might try to decrypt a ballot to a different candidate, but this would violate the proof of correct decryption.
- Ballot removal: Since the BB cannot delete ballots they can only be removed during the shuffling and tallying phase. As before, the proof of correct shuffle prevents this attack. During tallying, $\mathcal{A}$ might wrongly mark a ballot cast with real credentials, as being a product of coercion. This is avoided by the soundness of the PET proof.
- Ballot injection: $\mathcal{A}$ will need to associate a credential with an identity. Assuming that the voter roll provided to the RA is trustworthy (a necessary condition for all elections), the attacker must associate the stuffed ballot with an existing identity, which can be detected because the correspondence of credentials to identities is public and our assumption that the RA and the BB are not simultaneously corrupted.

Theorem 5 (Strong verifiability). *The protocol of Figure 3 is strongly verifiable according to [29] assuming that* DLOG *is hard in* $\mathbb{G}$*, that the soundness error of the* NIZK *schemes is negligible and that* Enc *is correct.*

Intuition. We prove that our scheme satisfies the sufficient conditions for weak verifiability defined in [29], i.e. correctness, accuracy and tally uniqueness. Therefore, it is weakly verifiable, that is, verifiable under the assumption that both the RA and the BB are honest). Furthermore, since our LRS has the property of unforgeability, it also satisfies the notion of strong verifiability of [29], i.e. verifiability when the BB and the RA are not simultaneously corrupted.

Firstly we will show that our scheme satisfies the notion of weak verifiability of [29]. According to [29, Theorem 4.1] it suffices to show that our protocol satisfies correctness, accuracy and tally uniqueness. Correctness is self-evident from our scheme specification in Figure 3.

Accuracy intuitively means that every ballot that is deemed valid corresponds to a correct vote and that the proof produced by the tally function will successfully pass VS.Vrfy.

Lemma 1 (Accuracy). *Assuming that the* DLOG *problem is hard in* $\mathbb{G}$ *and that the soundness error of the* NIZK *scheme is negligible, our scheme provides accuracy.*

Proof. To prove the former claim consider a ballot $(\mathtt{m}, L^{(i)}, \sigma)$ where $\mathtt{m}$ contains $\mathsf{Enc}(\mathsf{cnd}; r)|\pi_{\mathsf{Enc}}^{L^{(i)}}$. Assuming that $\mathsf{VS.IsValid}(b, \mathsf{BB}) = 1$ we deduce that the ballot is syntactically correct, that it is unique in the BB and that the ring $L^{(i)}$ is correctly constructed. We also deduce that:

- $\mathsf{NIZK.Vrfy}(\pi_{\mathsf{Enc}}^{L^{(i)}}) = 1$. This means that $\mathsf{Enc}(\mathsf{cnd}; r)$ encrypts a real candidate $\mathsf{cnd} \in \mathsf{CS}$ with soundness error $\frac{1}{q}$.
- $\mathsf{LRS.Vrfy}(L^{(i)}, \mathsf{ev}, \mathtt{m}, \sigma) = 1$ which implies that a credential corresponding to sc_i^* has been included in the ring $L^{(i)}$ unless the signature has been forged. This credential might be real or fake. However, the voter doing the verification, maybe using their own device, is aware if they have included in $L^{(i)}$ the public counterpart of the real or the fake credential.

Furthermore, if $\mathsf{VS.Vrfy}(\{b\}, \mathsf{Tally}(\mathsf{sk}_{\mathsf{TA}}, \{b\})) = 1$ then:

- $\mathsf{NIZK.Vrfy}(\pi_{\mathsf{Dec}}) = 1$. This means that $\mathsf{Enc}(\mathsf{cnd}; r)$ has been correctly decrypted with soundness error $\frac{1}{q}$.
- $\mathsf{NIZK.Vrfy}(\pi_{\mathsf{PET}}) = 1$. This means that the PET has been executed correctly on ballot b.

Combining the statements above, we can reach the conclusion that a ballot which passes validation and yields a tally that passes verification will lead to a vote that will be counted if the real credential has been used.

Lemma 2 (Tally uniqueness). *Assuming that* Enc *is correct and the* NIZK *schemes* $\pi_{\mathsf{Dec}}, \pi_{\mathsf{PET}}$ *are sound, our scheme provides tally uniqueness.*

Proof. Assume that an adversary has managed to create a BB and $(T_1, \pi_1), (T_2, \pi_2)$ such that $T_1 \neq T_2$ and $\mathsf{Vrfy}(\mathsf{BB}, T_1, \pi_1) = \mathsf{Vrfy}(\mathsf{BB}, T_2, \pi_2) = 1$. This implies that there exists at least one ballot $b \in \mathsf{BB}$ for which $\mathsf{Enc}(\mathsf{cnd}; r)$ can be decrypted both as $\mathsf{cnd}_1, \mathsf{cnd}_2 \in \mathsf{CS}$ with $\mathsf{cnd}_1 \neq \mathsf{cnd}_2$ for π_{Dec} (which is the same in both cases as there is a single bulletin board) correctly verifies. But this goes against the correctness properties of the encryption and proof schemes. Alternatively, there may be two ballots in the BB b_1, b_2 that encrypt different $\mathsf{cnd}_1, \mathsf{cnd}_2 \in \mathsf{CS}$ with $\mathsf{cnd}_1 \neq \mathsf{cnd}_2$ and the adversary can obtain T_1 by counting cnd_1 and T_2 by counting cnd_2 due to the validity of the credentials. This violates the soundness of the proof π_{PET}.

Theorem 6 (Weak verifiability). *Our protocol is weakly verifiable assuming that the* DLOG *problem is hard in* $\mathbb{G}$*, that the soundness error of the* NIZK *schemes is negligible and that* Enc *is correct*

Proof. Implied by Lemma 1 and Lemma 2 according to [29, Theorem 4.1].

We can now prove that our scheme is strongly verifiable (Theorem 5).

Proof. In Theorem 1 we proved that our LRS construction is unforgeable. Therefore, according to [29, Theorem 4.3] since our scheme is weakly verifiable (Theorem 6) it also satisfies the property of strong verifiability.

5.3 Privacy

Ballot secrecy Our scheme satisfies the BPRIV definition [2] which essentially states that the cryptographic components of a voting system do not leak information that could help an attacker uncover the vote of an honest voter beyond what is deducible from the election result.

Again, we first examine how our scheme fares agaist some well-known attacks in the literature. The use of the strong Fiat-Shamir transform protects from the attacks of [12]. Secondly, ballot weeding via the $\mathsf{VS.IsValid}$ functionality and later through the tallying process, provides ballot independence and prevents replay attacks that seek to break privacy [32]. In particular, these checks (cf. section 4) together with the proof $\pi_{\mathsf{Enc}}^{L^{(i)}}$ prevent an adversary from replaying entire ballots or only their individual $\mathtt{m}_i$ components, verbatim or through malleability, to protect privacy [32]. Importantly, this is done without leaking the identity of the voter. We stress that even if the ballots were malleable, their binding with the ring $L^{(i)}$ would force a replay adversary to know a secret credential belonging to the particular $L^{(i)}$ in order to sign the copied ballot.

More formally, in the BRPIV definition [2], the adversary $\mathcal{A}$ must distinguish between two bulletin boards $\mathsf{BB}_0, \mathsf{BB}_1$ which are built by the ballots cast by honest and corrupt voters. The former may cast different choices to each of $\mathsf{BB}_0, \mathsf{BB}_1$ (selected by the adversary), while the latter casts the same ballot to both. The tally algorithm always executes on BB_0 while $\mathcal{A}$ observes one of $\mathsf{BB}_0, \mathsf{BB}_1$ chosen by the challenger uniformly at random. $\mathcal{A}$ must guess which of $\mathsf{BB}_0, \mathsf{BB}_1$ they viewed.

Theorem 7 (Ballot secrecy). *Our scheme provides ballot secrecy according to BPRIV, assuming the soundness of the* NIZK *proofs and that the ElGamal encryption scheme together with the proof* $\pi_{\mathsf{Enc}}^{L^{(i)}}$ *satisfy NM-CPA security.*

Intuition $\mathcal{A}$ succeeds in BPRIV only with negligible probability. To argue about this, it suffices to prove that the challenger can successfully swap all honest ballots between $\mathsf{BB}_0, \mathsf{BB}_1$ without the adversary noticing. Since our scheme supports two tallying methods, this must be proved for both. The case of homomorphic tallying is easier. If the attacker observes BB_1 the challenger must simulate the proof of correct decryption π_{Dec} only for the aggregate result. On the other hand, if each individual ballot is decrypted, then the challenger must simulate the proof of correct decryption π_{Dec} for each ballot. This is not a problem since in both cases π_{Dec} possesses the (special honest-verifier) zero-knowledge property. Hence, the challenger can always decrypt each ballot in BB_0 to the corresponding honest vote in BB_1 and provide a (simulated for different honest votes) proof that the adversary can distinguish with negligible probability.

There is a subtle issue with [2] in case each ballot is decrypted. Since the adversary dictates the honest choices, they may dictate a particular and very rare choice for one of the bulletin boards and easily win the game by checking in which of $\mathsf{BB}_0, \mathsf{BB}_1$ it appears after decryption. For instance, if the voters must rank a lot of candidates, then the adversary may require that a particular

permutation of unpopular candidates appears in the last places of the ranking so that they can distinguish on which of $\mathsf{BB}_0, \mathsf{BB}_1$ it appears (i.e. the well-known Italian attack [21]). This attack applies to all schemes where individual ballots are decrypted and not only to ours and allows the adversary to defeat the definition of ballot secrecy. As this is a general characteristic of the definition and not of particular systems, in order to overcome it, we assume that all voter choices dictated by the adversary in $\mathsf{BB}_0, \mathsf{BB}_1$ are equal as multisets.

To prove Theorem 7 we adapt the BPRIV model [2] to our scheme.

Proof. We define a sequence of games that transition the view of the privacy adversary $\mathcal{A}$ from an election where a bulletin' board BB_0 is both observed and tallied, to an election where BB_1 is observed by $\mathcal{A}$ and BB_0 is tallied.

The adversary may corrupt some voters and cast ballots on their behalf. This will be represented using an oracle $\mathcal{OC}(i, b)$ which casts the same ballot to both $\mathsf{BB}_0, \mathsf{BB}_1$. As a result, they do not need to be swapped and the challenger does not deal with them any further.

Regarding honest votes, the adversary selects their choices by using an oracle $\mathcal{OV}$. A call $\mathcal{OV}(i, \mathsf{cnd}_0, \mathsf{cnd}_1)$ dictates that an honest voter casts a ballot for cnd_0 in BB_0 and cnd_1 in BB_1. We assume that $\mathcal{OV}$ always uses a valid credential, as voting with a fake one would not help the privacy attacker.

In what follows, the challenger maintains a table containing tuples created by the calls to $\mathcal{OV}$ containing the values $(i, \mathsf{cnd}_0, b_0, \mathsf{cnd}_1, b_1)$. As explained in section 5.3, we assume that the set of choices $\{\mathsf{cnd}_0\}$ selected in BB_0 is equal as a multiset to the set of choices $\{\mathsf{cnd}_1\}$ selected in BB_1.

Assume that there are m honest ballots cast in each of $\mathsf{BB}_0, \mathsf{BB}_1$. Game_0 is the BPRIV game where BB_0 is both observed and tallied. In $\{\mathsf{Game}_i\}_{i=1}^{m}$ the challenger swaps the i-th ballot of BB_0 with the i-th ballot of BB_1. As in the proof of section D.1 of [2], given an adversary that can distinguish between Game_i and Game_{i-1} with non-negligible probability, we can construct an adversary that can break the NM-CPA property of an ElGamal ciphertext c accompanied by the proof π_{Enc} with the same non-negligible probability. It is easy to see that Game_m is the game where the adversary observes BB_1. The challenger must now tally BB_0 without the adversary knowing. The functionalities $\mathsf{Shuffle}, \mathsf{PET}$ do not aid $\mathcal{A}$ any further, since they all take as input any bulletin board and operate on the ballot ciphertexts. Furthermore, PET returns 1 for all honest ballots in both cases, since $\mathcal{OV}$ utilizes the correct credentials.

The only way the attacker could distinguish the bulletin boards is from the result, since they can calculate what the result is in BB_0 from all the votes $\{\mathsf{cnd}_0\}$ they selected, and the corresponding result in BB_1 from $\{\mathsf{cnd}_1\}$. The adversary expects to see the result from $\{\mathsf{cnd}_0\}$, but in Game_m the tally algorithm is applied to $\{\mathsf{cnd}_1\}$. To fool the adversary the challenger utilizes the existence of the simulator implied by the zero-knowledge property of π_{Dec}. This proves that $\mathsf{Dec}(c_1, c_2) = \mathtt{m}$ by showing $(g, c_1, \mathsf{pk}, c_2\mathtt{m}^{-1})$ is a valid Diffie - Hellman tuple using the Chaum-Pedersen protocol [15]. This proof can be simulated to show this for any value $v \in \mathbb{G}$ instead of $\mathtt{m}$.

As a result, in the case that all votes are homomorphically decrypted the challenger announces the tally T_0 of BB_0 and creates a proof for $(g, c_1, \mathsf{pk}, c_2 T_0^{-1})$ where c_1, c_2 are the 'aggregated ballots' in BB_1. In the case that each ballot is individually decrypted and then tallied in plaintext form, the challenger creates proofs $\left\{(g, c_{1i}, \mathsf{pk}, c_{2i}{\mathsf{cnd}_{0i}}^{-1})\right\}_{i=1}^{m}$ where $\{c_{1i}, c_{2i}\}$ comprise the individual ballots in BB_1. Note that since $\{\mathsf{cnd}_0\}$ and $\{\mathsf{cnd}_1\}$ are equal as multiset the challenger may always find a cnd_0 for all cnd_1 selected by the adversary in their $\mathcal{OV}$ calls.

To conclude the proof, we also argue that our scheme satisfies the properties of strong consistency and strong correctness of [2].

Regarding strong consistency, since our ballots do not contain voter identities as in the version of Helios studied in [2] we define an $\mathsf{Extract}$ algorithm that is not required to output a voter identity. Instead, our $\mathsf{Extract}$ algorithm outputs all credentials in $L^{(i)}$. In more details, it accepts as input the tallier secret key $\mathsf{sk}_{\mathsf{TA}}$ and parses a ballot b as $(\mathsf{Enc}(\mathsf{cnd}; r)|\pi_{\mathsf{Enc}}^{L^{(i)}}, L^{(i)}, \sigma)$ and decrypts $\mathsf{Enc}_{\mathsf{pk}}(\mathsf{cnd}; r), L^{(i)} = \{\mathsf{Enc}_{\mathsf{pk}}(g^{x_i} h^{y_i})\}_{i=1}^{\beta}$ to obtain $\mathsf{cnd}, \{g^{x_i} h^{y_i}\}_{i=1}^{\beta}$. We also define an independent valid function $\mathsf{IndValid}$ that verifies the signature and the proof $\pi_{\mathsf{Enc}}^{L^{(i)}}$. It is easy to see that the first and second conditions of strong consistency are satisfied by construction. Furthermore, an adversarial ballot box will yield the same result both when the Tally algorithm is applied and when the $\mathsf{Extract}$ algorithm is applied on the ballots and the values $\mathsf{cnd}, \{g^{x_i} h^{y_i}\}$ are counted according to the voting rules. The reason for this is that in the latter case the credentials $g^{x_i} h^{y_i}$ will be compared in plaintext form and not through the PET but the output will be the same. Consequently, in both cases the same ballots will be counted.

Regarding strong correctness, it is easy to see that an honest ballot is going to be accepted even for an adversarially created bulletin board identically to [2]. According to our scheme's specification in Figure 3 an honestly generated ballot will only be discarded if the adversary has managed to output an identical one before. But this has negligible probability to occur since ElGamal and all other primitives that are used in the construction of our ballot are probabilistic.

Everlasting privacy Our scheme provides practical everlasting privacy [33]. Given only the publicly available election data in the BB, a powerful attacker that can break the encryption of the published credentials and the ballots, still cannot map honest ballots to honest voters. This is due to the unconditional anonymity of our LRS construction. Indeed, assume that $\mathcal{A}$ retrieves all $\{g^{x_i} h^{y_i}\}_{i=1}^{n}$ from to $\{\mathsf{pc}_i' = \mathsf{Enc}(g^{x_i} h^{y_i})\}_{i=1}^{n}$ in $\mathsf{BB}_{\mathsf{SetupElection}}$ and all $\{g^{x_i} h^{y_i}\}_{i=1}^{\beta}$ for each ballot in $\mathsf{BB}_{\mathsf{Vote}}$. Then $\mathcal{A}$ knows which identities comprise the anonymity set selected by V_i. However, the unconditional anonymity (cf. Theorem 2) of our signature does not allow to pinpoint exactly which of these identities actually signed and submitted the ballot. Even, assuming a potential attack where the adversary tries to distinguish if a credential is used in fewer than β ballots, they don't have a significantly better probability of pinpointing the ballot of this voter, since they cannot be sure whether the voter that owns this credential actually

voted or abstained. The expected number of times that each credential is used, given that N valid votes are in the final tally is $\frac{\beta \cdot N}{n}$, since each credential has probability $\frac{\beta}{n}$ to be in $L^{(j)}, j \in \{1, \dots N\}$.

More formally, we can show that our scheme provides the computational equivalent of practical everlasting privacy, namely weak everlasting privacy as defined in [34]. In more detail, in the game-based definition of weak everlasting privacy of [34] is given access to two bulletin' boards BB_0 and BB_1 that are built through successive interactions with oracle calls to the vote functionalities (like in section 5.3). However, since in weak everlasting privacy the adversary is not allowed to affect the election (since the attack is in the future) they are not allowed to cast adversarial ballots through $\mathcal{OC}$. In this regard, the adversary dictated choices in $\mathcal{OV}$ should be equal as multisets per bulletin board, i.e. the same number of votes for cnd_0 and cnd_1 should be cast in BB_0 and BB_1, otherwise the adversary can easily win by counting the votes for each candidate (as we discussed in section 5.3). Consequently, there is no need for the challenger to simulate the proof of the election tally, as in the proof of Theorem 7. Thus, the proof of weak everlasting privacy is similar (and in fact simpler). When the adversary calls $\mathcal{OV}(i, \mathsf{cnd}_0, \mathsf{cnd}_1)$, the ballots that are produced from the two choices are signed by the voter i. The only way for the adversary to win is to pinpoint that *for voter* i a ballot for cnd_0 is actually located in BB_0 and a ballot for cnd_1 is actually located in BB_1. However, assuming that the adversary could do this, then they could break the anonymity of our scheme, which is unconditional. Thus, the following theorem follows.

Theorem 8. *Our scheme provides weak everlasting privacy as defined in [34].*

It must be also noted that a similar analysis also applies to the contemporary election talliers. Indeed, the TA might not be computationally powerful but is in possession of the private decryption key $\mathsf{sk}_{\mathsf{TA}}$. As a result, it is conceptually equivalent to the powerful future attacker of everlasting privacy. The TA, thus, may legitimately decrypt the individual ballots in order to compute the result, but they cannot deduce the identity of the voter, but only whether the ballot is valid or not. As a result, the common trust assumption in most voting schemes that the TA is trusted for privacy does not need to apply for our scheme.

Regarding the notion of strong everlasting privacy of [34], our scheme does not provide it as described in Figure 3, since the adversary may use insider data, like the IP address of the voters to identify the ballot of voter i and check if cnd_0 is located in BB_0 and cnd_1 is located in BB_1. However, if voters combine this unconditionally anonymous 'channel' implemented at the endpoints with countermeasures such as a Tor client or even a public computer to hide networking information (e.g. IP addresses), they erase all traces of insider data that are possible to obtain. In this case, our scheme would also provide strong everlasting privacy.

5.4 Coercion Resistance

Our system satisfies the JCJ definition of coercion resistance [7, 25] which consists of two games, the real and the ideal. In the real game, a coin is flipped to determine whether the voter will provide the real or a fake credential to the adversary. If a fake credential is provided, the voter may cast a ballot using their real credential. The adversary $\mathcal{A}$ can cast a ballot using the provided credential, as well as ballots using corrupt credentials. Honest voters cast their ballots in an order dictated by the adversary. The goal of $\mathcal{A}$ is to guess whether their attack succeeded, namely, whether the vote cast with the coerced credential was counted in the final tally. The ideal game is similar to the real game with the difference that $\mathcal{A}$ always gets the real credential, cannot cast ballots using corrupted credentials, doesn't have access to any cryptographic material nor BB and an idealized tally is produced that includes only the valid votes. $\mathcal{A}$ wins if they can distinguish the real from the ideal game with non-negligible probability.

Intuitively, the fake but indistinguishable credentials allow the voter to seemingly obey the instructions of the coercer, but 'undo' them during their moment of privacy. The anonymization offered by the mixing and our signature prevents the attacker from discovering if their ballot was counted or not. Linking is of no use to the coercer, since the ballots ordered by them and the ones during the moment of privacy are cast with different credentials. Additionally, our scheme avoids tagging attacks caused by allowing each voter to select their own size of anonymity set. Indeed, if each voter could select the number of decoy credentials, then the coercer could tag the ballot by forcing the use of a particular size for the anonymity set (e.g. 1009 [28]) and then with very high probability find out if it was discarded or not. This is the reason why the anonymity set has a global constant size β. As far as receipt-freeness is concerned, a voter may claim ownership of a ballot, by disclosing pc and the randomness used to create it, but cannot prove whether it was counted in the final tally.

Theorem 9 (Coercion resistance). *If the DDH assumption holds in $\mathbb{G}$, then our scheme is coercion-resistant according to the model of [7].*

Intuition When voter j is under coercion, the voting device generates a fake credential $\mathsf{sc}_j^* = (x^*, y^*)$. The adversary cannot distinguish whether the credential is real, since this would be equivalent to deciding whether $(g, \quad L_{i1}^{(j)} = g^r, \quad \mathsf{pk}, \quad \frac{L_{i2}^{(j)}}{g^{x^*}h^{y^*}} = \frac{g^x h^y}{g^{x^*}h^{y^*}}\mathsf{pk}^r)$ is a DDH tuple, assuming that the coercer hasn't fully corrupted the RA and the TA, and thus neither the coercer nor the voter know the randomness of the encryption of pc_j. The ballot created by $\mathsf{VS.Vote}$ and then posted on the BB is indistinguishable from a ballot created with real credentials, since it contains a valid signature, valid proofs and the list $L^{(j)}$ that contains an encryption of pc_j^*. We note that, a signature produced by the voter during the moment of privacy cannot be detected by the coercer, since the signature contained in each ballot has unconditional anonymity, as described in Theorem 2 and the linking tag is different, since the ballot was cast with a different credential. After $\mathsf{VS.Shuffle}$ the coercer loses track of their ballot and in

the $\mathsf{VS.Tally}$ the proofs of the tally do not reveal anything more to the coercer, since the ballots and the credential list where shuffled.

In order to prove coercion resistance we adopt the model of [7, 25], where a non-adaptive adversary is assumed. Coercion resistance is defined with two games, the real and the ideal. The real game aims to model the behavior of the adversary, the honest voters and the coerced voter in a real coercion scenario, where the adversary tries to decide whether the attack succeeded or not. The goal is the same in the ideal game, except that $\mathcal{A}$ doesn't have access to any cryptographic material or the BB, therefore they have no advantage in distinguishing the success or failure of the attack.

We now prove Theorem 9.

Proof. In the real game, the challenger sets up the election and the adversary $\mathcal{A}$ chooses the set of voters they wish to corrupt. Then the challenger registers all the voters and yields to $\mathcal{A}$ the secret credentials of the corrupted voters. Next, $\mathcal{A}$ chooses a voter j to coerce. A random coin $\mathfrak{b}$ is flipped, to model the behavior of the coerced voter. If $\mathfrak{b} = 0$ the voter provides a fake credential to the adversary, whereas if $\mathfrak{b} = 1$ the voter provides the real credential. Afterwards, the adversary programs the voting of the corrupted and the honest voters in whatever order it suits them. The honest voters vote according to a distribution $\mathcal{D}$ which aims to model uncertainty in their behavior and the inability of $\mathcal{A}$ to predict it. For instance, some voters may abstain from voting, or cast an invalid vote. If the attacker could fully predict the behavior of the honest voters then they would trivially win the game by checking the tally. If $\mathfrak{b} = 0$ the coercer casts a ballot with the fake credential, while in the moment of privacy the voter casts their ballot with their real credential. If $\mathfrak{b} = 1$ the voter hands control to the coercer who casts a ballot with the real credential. After the tally is counted, the adversary tries to guess $\mathfrak{b}$ to see if their coercion attempt succeeded.

The ideal game is similar to the real one, with the difference that the coercer always gets the real credential, does not have access to cryptographic material and the BB, and an idealized tally is posted, containing only the valid votes maintained by the challenger in the reduction.

We note that in both the real and ideal game there is one more vote in the final tally if $\mathfrak{b} = 0$. This information cannot be used by $\mathcal{A}$ to distinguish whether $\mathfrak{b} = 0$ or $\mathfrak{b} = 1$, since the honest voters behave according to the distribution $\mathcal{D}$ and thus $\mathcal{A}$ cannot predict the total number of votes in the tally.

We can prove coercion resistance by a sequence of games, from the real to the ideal, where the advantage of the adversary to distinguish which game they are playing is negligible. The initial game Game_0 is the real coercion resistance game. In Game_0 if $\mathfrak{b} = 0$ the challenger casts a ballot using the real credential $\mathsf{sc}_j = (x, y)$, and constructs a fake credential, by choosing a random $\mathsf{sc}_j^* = (x^*, y^*) \leftarrow\!\!\$\ \mathbb{Z}_q^2$ and giving it to the coercer. If $\mathfrak{b} = 1$ the challenger hands the real credential $\mathsf{sc}_j = (x, y)$ to the coercer. Then the honest and the corrupted voters vote, as well as $\mathcal{A}$ using the credential (real or fake) of the coerced voter, according to the order $\mathcal{A}$ instructed. The tally is produced and the adversary must guess whether $\mathfrak{b} = 0$ or $\mathfrak{b} = 1$.

The intuition behind Game_1 is that the real vote of the coerced voter doesn't give advantage to the coercer. Game_1 is similar to Game_0 with the difference that if $\mathfrak{b} = 0$ the challenger casts a ballot using a random value $(z_1, z_2) \leftarrow\!\!\$\ \mathbb{Z}_q^2$ as a credential. We note that $\mathcal{A}$ cannot distinguish whether they are playing Game_0 or Game_1, since if $\mathfrak{b} = 1$ the games are identical and if $\mathfrak{b} = 0$ then the only difference is that the ballot $b_j = (\mathtt{m}, L^{(j)}, \sigma)$ of Game_0 is replaced by the ballot $\hat{b_j} = (\hat{\mathtt{m}}, \hat{L}^{(j)}, \hat{\sigma})$ in Game_1. In this case, assuming that the credential of voter j is in the i-th position of the list $L^{(j)}$ and in the i'-th position of $L^{(0)}$, the values

$$\left(g, \quad \frac{L_{i1}^{(j)}}{L_{i'1}^{(0)}} = g^{r_{ji} - r_{0i'}}, \quad \mathsf{pk}, \quad \frac{L_{i2}^{(j)}}{L_{i'2}^{(0)}} = \frac{g^x h^y \mathsf{pk}^{r_{ji}}}{g^x h^y \mathsf{pk}^{r_{0i'}}} = \mathsf{pk}^{r_{ji} - r_{0i'}}\right)$$

in Game_0 form a Diffie-Hellman tuple, since the real credential was used in the $\mathsf{VS.Vote}$ algorithm, but in Game_1 the last element of this tuple is $\frac{\hat{L}_{i2}^{(j)}}{L_{i'2}^{(0)}} = \frac{g^{z_1} h^{z_2} \mathsf{pk}_1^r}{g^x h^y \mathsf{pk}^{r_{0i'}}}$, which is a random element of $\mathbb{G}$, since a random value was used as credential in the vote algorithm. Thus the adversary cannot distinguish them, assuming that the DDH problem is hard.

Game_2 aims to model that the adversary cannot distinguish whether the coerced voter supplied the correct or a fake credential. It is similar to the Game_1 with the difference that if $\mathfrak{b} = 0$ the challenger provides the real credential sc_j to the coercer, instead of a random value. We note that $\mathcal{A}$ cannot distinguish whether they are playing Game_1 or Game_2, since if $\mathfrak{b} = 1$ the games are identical and if $\mathfrak{b} = 0$ the adversary has to decide whether a tuple is a DDH tuple, similarly to the previous case. In this case, by using the credential provided to the adversary, the values

$$\left(g, \quad L_{i1}^{(j)} = g^r, \quad \mathsf{pk}, \quad \frac{L_{i2}^{(j)}}{g^x h^y} = \mathsf{pk}^r\right)$$

in Game_2 form a Diffie-Hellman tuple, since the real credential was provided, but in Game_1 the last element of this tuple is $\frac{L_{j2}}{g^{x^*} h^{y^*}}$, which is a random element of $\mathbb{G}$, since a fake credential was provided. Thus the adversary cannot distinguish them, assuming that the DDH problem is hard.

Game_3 aims to model that the adversary cannot obtain any information by the actions of the honest voters. It is similar to Game_2 with the difference that the challenger casts a vote for *each* honest voter using random credentials instead of their real credentials. The challenger produces a modified tally in Game_3, containing only the result of the elections, without any proofs. In the final result the votes of coerced voters are counted and the votes of the honest voters are counted only if they were intended to, despite that random credentials were used. Therefore the final tally is the same as the final tally of Game_2. Similarly to the previous case, $\mathcal{A}$ cannot distinguish whether they are playing Game_2 or Game_3, due to the hardness of the DDH problem.

We note that the ideal game is essentially Game_3 since all the values included in each ballot not constructed by $\mathcal{A}$ are random elements of $\mathbb{Z}_q$ or $\mathbb{G}$. As a result,

our scheme provides coercion resistance, since the adversary cannot distinguish between the real and the ideal coercion resistance games. □

6 Related work

To the best of our knowledge, this work is the first *vote-and-go* voting scheme in the fake-credentials paradigm that achieves coercion resistance, everlasting privacy, and verifiability. As a bonus, these properties are achieved with minimal trust assumptions. In particular it suffices to have only one honest member of the registration authority and of the tally authority to ensure coercion resistance. We also achieve, ballot secrecy, everlasting privacy and verifiability even if all parties are corrupt. To justify this claim and our design choices, we review related work, regarding linkable ring signature schemes and their applications to e-voting, coercion-resistant protocols and schemes providing everlasting privacy. We summarize our findings in Table 1.

Ring Signatures. Voting schemes have been closely related to zero-knowledge proofs and signatures as means to force untrusted players to behave correctly ([1, 3, 12, 31, 35]) and as proofs of knowledge of private credentials used in voter authentication ([7, 21]). Additionaly, the versatility of some signature schemes has also been employed to provide privacy. For instance, the protocols of [36, 37] use blind signatures [38] and variations [39] to provide unconditional unlinkability and protect the voters' choice, while allowing them to authenticate themselves with a registrar and prove their eligibility to vote.

The seminal work of [9] was the first to introduce linkable ring signatures to prevent double voting and provide strong levels of anonymity, while eliminating the need for a registration authority. However, the accompanying voting scheme does not provide coercion resistance and none of its other security properties are formally defined or analyzed. The same architecture was later improved in [40], which compressed the signature to constant size, yielding a more efficient scheme, without providing any new features, like coercion resistance. Recently, [41] adapted [9] to a blockchain-based setting, without any changes to the main architecture or any new feature. In contrast, the work of [42] added receipt-freeness - a weaker property than coercion resistance - to [9] by allowing the voters to cast multiple ballots and by combining the linkable ring signature with a (strongly) designated-verifier signature. This change, according to the authors, allowed their scheme to satisfy receipt-freeness, i.e. protection from vote-selling. However, they did not protect against the stronger adversary of the JCJ framework, who is present during voting and can force the voter to cast a specific ballot or abstain. Furthermore, the fact that the talliers in [42] need to provide their secret keys in order to verify the designated signatures on the ballots, means that the tally is not publicly verifiable, even if the voters can pinpoint their own votes in the bulletin board. Another common characteristic of [9, 40–42] is that they do not formally define the proposed voting protocols and their security properties, nor do they provide a rigorous security analysis. Finally, despite

the fact that linkable ring signatures provide stronger anonymity guarantees by construction, neither of the above works provide everlasting privacy, since the linking tag, used to prevent double voting, can also reveal the identity of the voter to a future adversary that can break the underlying cryptographic assumptions.

A linkable ring signature with unconditional anonymity can remedy this issue, as we show in our work. The work of [43] exploited the use of such signatures over a commodity internet anonymous channel to create a voting scheme with unconditional anonymity and weak receipt freeness. The first linkable ring signature with unconditional anonymity was proposed in [8], but had a major drawback: it only provided weak linkability, meaning that an adversary in possession of n secret keys could produce more than n unlinkable signatures, allowing them to double vote undetected and break the integrity of the election. Our own linkable ring signature scheme overcomes this limitation by providing strong linkability, which means that an adversary with n secret keys can only produce n unlinkable signatures. Thus, our signature scheme can be viewed as an improvement over [8] which is a contribution in its own right, considering the fact that creating efficient ring signatures with strong linkability has been a challenging problem. Most schemes originating from the DualRing paradigm [44] only provide weak linkability, as is the case with [45, 46] despite their claims to the contrary, as proved in recent attacks [46, 47] respectively. In our scheme, however, using a logarithmic size signature would not improve the ballot size asymptotically, since each ballot must include a reencryption of the whole anonymity set, as described in section 4, forcing a linear size. As a result, strong linkability is more important for the security of our voting scheme than signature size.

Coercion Resistance. The JCJ framework [7], implemented in Civitas [21], introduced the fake credential paradigm, which has been the blueprint for many well-known coercion-resistant schemes. Its threat model considers three attacks: randomization, where the voter is forced to vote randomly, forced abstention, where the voter must not cast a vote at all, and simulation, where the coercer essentially takes total control of the voter. Coercion resistance is achieved by not allowing the adversary to be certain if their attack succeeded, i.e. if the voter followed their instructions in any of the attack scenarios. This is made possible by using two methods: multiple votes per voter and anonymous credentials. The voter follows the coercer's instructions when they are present, providing a fake and indistinguishable credential, but during *a moment of privacy* (a necessary assumption) they cast their real vote. The coercer cannot tell in which case the posted ballot was counted, because it is anonymized. Of course, if the coercer has perfect knowledge of the voting pattern of all the voters, then coercion is unavoidable. As a result, the JCJ framework makes an 'uncertainty assumption', i.e. that the coercer is not certain about the behaviour of all the voters. To maintain verifiability, the talliers must prove that they counted the correct vote without leaking the credential that came with it. For this they employ a *plaintext equivalence test* (PET) [20].

One of the main drawbacks of both JCJ and Civitas is the *quadratic* tallying time to cleanse the ballots, i.e. weed out real and fake votes using the PET. Much

work has been done to make this cleansing process more efficient. For instance, the scheme of [48] and its improvement [49] rely on bilinear maps to achieve linear complexity by using structured credentials that contain information allowing to efficiently group and identify fake ballots. The second version of CHide [50] and its improvement [51] achieve a theoretical quasilinear runtime. They propose using MPC techniques on the binary representation of credentials to perform an encrypted PETbetween the registered credentials and the ones included in the ballots, which allows them to sort the ballots and separate the real from the fake ones in quasilinear time. The motivation for this approach is the claim that the 'uncertainty assumption' of JCJ does not apply to the number of revotes which in certain (extreme) scenarios can aid the coercer in identifying if their victim followed their instructions or not. However, the use of bitwise MPC techniques accompanied with extra rounds of communication adds a significant overhead to the protocol by a large constant (depending the security parameter), constituting a load that is not accounted for in the asymptotic analysis (which is a function of the number of votes) thus yielding practical inefficiencies. Finally, the work of [52] achieves linear complexity by dispensing with the removal of votes with fake credentials. Instead, it builds a list for each ballot, such that its last item is the one to be counted. While the tallying phase is linear per se, the scheme has a cleansing phase the complexity of which depends on the number of noise ballots per voter, thus amounting to quadratic runtime. However, its most important drawback is that it requires fully trusting the election authority (registrars and talliers) for coercion resistance.

Another problem of JCJ-related schemes, is that they require the voters to handle cryptographic material. Specialized voting devices or more user-friendly techniques like panic passwords [25] may be used to make such schemes practical and usable. Selections [25], uses this latter technique, and furthermore it associates tallying complexity with the degree of coercion resistance. This is achieved by allowing the voter to select an anonymity set of size β during ballot casting. This parameter also allows the trade-off between efficiency and coercion resistance to be adjusted. In this work, we also employ this technique, where β is the size of the ring in our signature.

While theoretically more efficient than our scheme, none of the previous works provides everlasting privacy, since all the ballots and related data cast by the voters can be revealed by a future adversary that can break the underlying cryptographic assumptions.

An alternative technique used to achieve coercion resistance is through *revoting* or *deniable vote updating*. In the revoting paradigm, when the voter is under coercion, they provide their real credential to the coercer. When the coercion attack is over, they can deniably update their ballot without being detected by the coercer. During tallying, a deniable filtering phase takes place where only the last vote cast with each credential is counted. As a result, the coercer cannot be certain that their directions were followed. This approach has the benefit that voter does not have to (convincingly) lie to the coercer. However, it makes a strong assumption, namely that the *moment of privacy* occurs after the coercion

attack and before the voting period ends. Another important drawback of revoting is that if the voter is coerced, they don't have the option to abstain from the election, since a coerced ballot can only be nullified by casting a valid one. The first scheme that provided coercion resistance through revoting, such that the voter did not have to remember previously cast ballots in order to revote, was the one of Lueks et al. [53] proposed a voting scheme of $\mathcal{O}(n \log n)$ complexity that achieves coercion resistance using the revoting technique, but requires that for each ballot cast, the voter has to authenticate with the election authority to obtain ephemeral voting tokens. Although it was claimed that this scheme was suitable for large scale elections, it was later proved in [54] that it requires trust to the election authority for all security properties, thus making it insecure. In [55] a similar protocol with [53] is presented, that reduced the tally complexity to linear, but providing a weaker notion of coercion resistance, and relying on the election authority for it together with verifiability, and privacy. The difference between strict coercion resistance and coercion resistance is that the former does not include the property of vote selling. The voter is able to evade coercion by casting a ballot after the coercion is over, but if they wish to sell their vote, they can prove how they voted with non-negligible probability.

The works of [56–58] consider a stronger notion of coercion-resistance, called *silent coercion*, which allows voter credentials to leak. Caveat Coercitor [56] evades silent coercion by discarding multiple votes cast with the same credential, requires cubic complexity, as it extends the JCJ framework but doesn't support revoting, in the case the voter wants to change their vote. On the other hand, VoteXX [57, 58] reduces the complexity to quadratic by using a mechanism to nullify votes cast by the adversary. The nullification of a vote can be performed either by the voter themselves or by a hedgehog, an entity trusted by the voter. Furthermore, VoteXX protects the voter from key loss by using inalienable authentication.

Everlasting privacy. Everlasting privacy can be achieved by either protecting the ballot, or the identity of the voter, or both. We classify related works in two main categories, depending on whether they use an anonymous casting phase [36, 59, 60] or not [5, 33, 35, 61]. The latter schemes have the voters cast ballots by posting a perfectly hiding commitment of their choice on a public bulletin board. The openings are later sent through private channels to the talliers, who can then verify through reconstruction the ballots and count the votes. Everlasting privacy is achieved by allowing only the part of the ballot box that contains the commitments to be revealed to the public. The election server maintains, but never discloses, a secret ballot box that contains the openings and other relevant data. It is *trusted* to delete these values after the election ends, thus making them unavailable in the future. Whether this trust is justified or not, is an open problem [34].

The degree of everlasting privacy achieved can be distinguished in two variations: *practical* (or *weak*) and *strong*. Practical everlasting privacy [33] (or the equivalent notion of weak everlasting privacy of [34]) protects against an adversary that can break cryptographic assumptions, but only has access to publicly

available data, i.e. the public part of the ballot box. Strong everlasting privacy [34] on the other hand, also protects against adversaries that have access to insider data maintained by the election authorities or intermediaries (such as decommitment values and metadata like network addresses). It is clear that schemes based on public commitments cannot achieve strong everlasting privacy because an insider can access the contents of the secret ballot box and reveal voter preferences. Anonymous casting is considered by some [62] a strong assumption, difficult to combine with verifiability. If this problem is remedied, though, there are some clear advantages in its use [34]. Firstly, it can provide strong everlasting privacy since there is no need for a secret ballot box. Everything submitted by the voters can be made public. Secondly, it can be used to provide (contemporary) ballot secrecy without trusting the election talliers for this property (as is the case in all schemes descending from [3, 31]), because a future computationally unbounded adversary is equivalent to an untrusted contemporary tallier.

The latter issue, i.e. whether the trust in the talliers for ballot secrecy is justified or not, is starting to gain consideration in the literature from different perspectives. [34] argues that while it is an accepted assumption in theory, in practice it is not at all clear if the voters would trust a system that does not mathematically guarantee the privacy of their votes. In fact, the usual remedy of splitting trust among multiple talliers through distributed key generation and threshold schemes, has practical and deployment issues that might undermine this separation, as was recently shown in [63], where a flaw was found in the distributed key generation protocol used in many such schemes, if the majority of talliers is dishonest. Additionally, [63] stressed the fact that if all the 'independent' talliers use hardware or software components that originate from the same vendor, or follow the same instructions, the very essence of the assumption on trust distribution is broken, since the trust is now concentrated on the central vendor / programmer / authority.

A similar doubt is raised in [64] where anamorphic encryption [65] is applied to electronic voting. The authors consider the case of an opressive regime that conducts elections to legitimize its rule. In this case, the voters are forced to participate in the election or face dire consequences, anamorphic voting allows them to report their true preference to an independent auditor, thus challenging the legitimacy of the election. While this scenario is different from the one considered for everlasting privacy, since in the latter case the elections are free in the present but need to be protected in the future, the concern about trusting the talliers for ballot secrecy is similar in both works.

Coercion resistance and everlasting privacy. The best known schemes that provide both coercion resistance and everlasting privacy based on anonymous casting are [59, 60]. A key insight of both these works is that the anonymous channel used during casting for everlasting privacy can also help with coercion resistance by thwarting the forced abstention attack. The scheme of [59] utilizes deniable vote updating, where the voters initially follow the coercer's instructions, but deniably change their ballot later to reflect their true choice. On the other hand,

[60] works in the JCJ setting with anonymous credentials based on a novel type of verifiable blind signatures [39] and performs tallying in linear time to the total number of votes. However, it requires an explicitly employing an anonymous channel over which every voter sends two messages in different phases of the protocol in order to cast their ballot (even for practical everlasting privacy). This hurts usability as it is not expected of voters to be available in two different phases of the protocol. Furthermore, despite the theoretical linear complexity of tallying, the blind signature scheme used in that work, was impacted by the ROS attack of [66] making the scheme impractical for real-world large-scale elections. Our work, is directly comparable to these two schemes since it aims for the same properties. Its main advantage is that it is a vote-and-go scheme, where the voters can cast their ballot in a single step. It also does not require an external anonymous channel for practical everlasting privacy, since it builds one by using our proposed linkable ring signature scheme. Finally, it combines all the advantages of the fake-credential paradigm over revoting, that were previously discussed.

7 Conclusion and future work

In this paper, we proposed a voting scheme that supports coercion resistance in the JCJ framework, ballot secrecy without the need to trust the election talliers, and everlasting privacy, without sacrificing verifiability. It is based on a new linkable ring signature scheme with unconditional anonymity. Our construction has favorable security properties; an open issue is the increased complexity of the tallying phase, which is in part inherent in the JCJ architecture and in part due to the requirements for everlasting privacy. In future work, we aim to improve the efficiency of the tallying phase. One mechanism we will explore in this direction, is the batching of PETs during the identification of coerced votes, which can reduce the complexity of the tallying phase by a factor of n. However, a naive implementation might negatively affect universal verifiability. In addition, we aim to reduce the complexity of the vote casting phase to allow the increase of the size of the anonymity set. We will also explore what would be the optimal value for β by taking into consideration the trade-off between performance and privacy for particular election types. Finally, we plan to improve the usability of the casting phase by integrating a mechanism similar to panic passwords [25] to allow the voters to easily generate their secret credentials. This must be done in a way that does not affect everlasting privacy.

Acknowledgements The authors would like to thank the anonymous reviewers of previous versions of this paper for their comments and suggestions which greatly improved this work.

This work has been supported by project MIS 5154714 of the National Recovery and Resilience Plan Greece 2.0 funded by the European Union under the NextGenerationEU Program and by the Special Account for Research Funding of National Technical University of Athens.

Work	Coercion Resistance		Everlasting Privacy			Tally complexity	Notes
	method	trust		method	trust		
[7]	fake credentials	t out of n_{TA}	-			$\mathcal{O}(mn)$	
[48, 49]	fake credentials	t out of n_{TA}	-			$\mathcal{O}(n)$	Bilinear maps
[50, 51]	fake credentials	t out of n_{TA}	-			$\mathcal{O}(m \log m)$	Heavy encrypted bitwise operations and communication for MPC.
[52]	fake credentials	EA	-			$\mathcal{O}(m \cdot n)$	
[25]	fake credentials	t out of n_{TA}	-			$\mathcal{O}(\beta m)$	Panic Passwords
[67]	revoting	t out of n_{TA}	-			$\mathcal{O}\big(m^2 + m \cdot n\big)$	
[59]	revoting	t out of n_{TA}	✓	perfectly hiding commitments	none	$\mathcal{O}\big(m^2 + m \cdot n\big)$	
[60]	fake credentials	t out of n_{TA}	✓	publicly auditable conditional blind signatures	none	$\mathcal{O}(m)$	Requires two-phase voter interaction. Limited Practical scalability due to the ROS attack.
[53]	revoting	EA	-			$\mathcal{O}(m \log m)$	Requires trust to EA for verifiability
[55]	revoting	EA	✓	anonymous credentials	EA	$\mathcal{O}(m)$	Strict CR requires trust to EA for verifiability
Our work	fake credentials	t out of n_{TA}	✓	linkable ring signatures with unconditional anonymity	none	$\mathcal{O}(\beta mn)$	Vote-and-go

Table 1. Overview of voting schemes that achieve coercion resistance and/or everlasting privacy. We denote with n the number of voters, with m the number of ballots on the BB with β the size of the anonymity set and with n_{TA} the number of talliers of TA.

References

[1] V. Cortier, D. Galindo, R. Küsters, J. Müller, and T. Truderung, "Sok: Verifiability notions for e-voting protocols," in *IEEE Symposium on Security and Privacy*, IEEE Computer Society, 2016, pp. 779–798. DOI: 10.1109/SP.2016.52.

[2] D. Bernhard, V. Cortier, D. Galindo, O. Pereira, and B. Warinschi, "Sok: A comprehensive analysis of game-based ballot privacy definitions," in *IEEE Symposium on Security and Privacy*, IEEE Computer Society, 2015, pp. 499–516. DOI: 10.1109/SP.2015.37.

[3] R. Cramer, R. Gennaro, and B. Schoenmakers, "A secure and optimally efficient multi-authority election scheme," in *EUROCRYPT*, 1997, pp. 103–118.

[4] D. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," *Commun. ACM*, pp. 84–88, 1981.

[5] T. Moran and M. Naor, "Receipt-free universally-verifiable voting with everlasting privacy," in *CRYPTO*, ser. LNCS, vol. 4117, 2006, pp. 373–392. DOI: 10.1007/11818175_22.

[6] J. Benaloh and D. Tuinstra, "Receipt-free secret-ballot elections (extended abstract)," in *STOC*, ACM, 1994, pp. 544–553. DOI: 10.1145/195058.195407.

[7] A. Juels, D. Catalano, and M. Jakobsson, "Coercion-resistant electronic elections," in *WPES 2005*, ACM, 2005, pp. 61–70. DOI: 10.1145/1102199.1102213.

[8] J. K. Liu, M. H. Au, W. Susilo, and J. Zhou, "Linkable ring signature with unconditional anonymity," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 157–165, 2014.

[9] J. K. Liu, V. K. Wei, and D. S. Wong, "Linkable spontaneous anonymous group signature for ad hoc groups (extended abstract)," in *ACISP*, ser. LNCS, vol. 3108, 2004, pp. 325–335. DOI: 10.1007/978-3-540-27800-9_28.

[10] T. E. Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *CRYPTO*, ser. LNCS, vol. 196, 1984, pp. 10–18. DOI: 10.1007/3-540-39568-7_2.

[11] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *CRYPTO*, ser. LNCS, vol. 263, 1986, pp. 186–194. DOI: 10.1007/3-540-47721-7_12.

[12] D. Bernhard, O. Pereira, and B. Warinschi, "How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios," in *ASIACRYPT*, ser. LNCS, vol. 7658, 2012, pp. 626–643. DOI: 10.1007/978-3-642-34961-4_38.

[13] E. McMurtry, O. Pereira, and V. Teague, "When is a test not a proof?" In *ESORICS*, ser. LNCS, vol. 12309, 2020, pp. 23–41. DOI: 10.1007/978-3-030-59013-0_2.

[14] C. Schnorr, "Efficient identification and signatures for smart cards (abstract)," in *EUROCRYPT*, ser. LNCS, vol. 434, 1989, pp. 688–689. DOI: 10.1007/3-540-46885-4_68.

[15] D. Chaum and T. P. Pedersen, "Wallet databases with observers," in *CRYPTO*, ser. LNCS, vol. 740, 1992, pp. 89–105. DOI: 10.1007/3-540-48071-4_7.

[16] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *CRYPTO*, ser. LNCS, vol. 839, 1994, pp. 174–187. DOI: 10.1007/3-540-48658-5_19.

[17] J. Groth, "Non-interactive zero-knowledge arguments for voting," in *ACNS*, ser. LNCS, 2005, pp. 467–482. DOI: 10.1007/11496137_32.

[18] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," in *EUROCRYPT*, ser. LNCS, vol. 1070, 1996, pp. 143–154. DOI: 10.1007/3-540-68339-9_13.

[19] M. Hirt and K. Sako, "Efficient receipt-free voting based on homomorphic encryption," in *EUROCRYPT*, ser. LNCS, vol. 1807, 2000, pp. 539–556. DOI: 10.1007/3-540-45539-6_38.

[20] M. Jakobsson and A. Juels, "Mix and match: Secure function evaluation via ciphertexts," in *ASIACRYPT*, ser. LNCS, vol. 1976, 2000, pp. 162–177. DOI: 10.1007/3-540-44448-3_13.

[21] M. R. Clarkson, S. Chong, and A. C. Myers, "Civitas: Toward a secure voting system.," in *IEEE Security and Privacy Symposium*, May 19, 2008.

[22] M. Backes, N. Döttling, L. Hanzlik, K. Kluczniak, and J. Schneider, "Ring signatures: Logarithmic-size, no setup—from standard assumptions," in *EUROCRYPT*, Springer International Publishing, 2019, pp. 281–311, ISBN: 978-3-030-17659-4.

[23] X. Bultel and C. Olivier-Anclin, "On the anonymity of linkable ring signatures," in *Cryptology and Network Security*, Springer Nature Singapore, 2025, pp. 212–235, ISBN: 978-981-97-8013-6.

[24] P. Behrouz, P. Grontas, V. Konstantakatos, A. Pagourtzis, *et al.*, "Designated-verifier linkable ring signatures," in *24th International Conference on Information Security and Cryptology - ICISC 2021*, ser. LNCS, vol. 13218, 2022, pp. 51–70. DOI: https://doi.org/10.1007/978-3-031-08896-4_3.

[25] J. Clark and U. Hengartner, "Selections: Internet voting with over-the-shoulder coercion-resistance," in *Financial Cryptography and Data Security, FC 2011*, ser. LNCS, vol. 7035, 2011, pp. 47–61. DOI: 10.1007/978-3-642-27576-0_4.

[26] J. Benaloh, "Simple verifiable elections," in *USENIX/ACCURATE Electronic Voting Technology Workshop (EVT)*, Vancouver, B.C.: USENIX Association, Aug. 2006. [Online]. Available: https://www.usenix.org/conference/evt-06/simple-verifiable-elections.

[27] R. Kusters, T. Truderung, and A. Vogt, "Clash attacks on the verifiability of e-voting systems," in *IEEE Symposium on Security and Privacy*, 2012, pp. 395–409. DOI: 10.1109/SP.2012.32.

[28] S. G. Weber, R. Araujo, and J. Buchmann, "On coercion-resistant electronic elections with linear work.," in *ARES*, IEEE, 2007, pp. 908–916. [Online]. Available: http://dblp.uni-trier.de/db/conf/IEEEares/ares2007.html#WeberAB07.

[29] V. Cortier, D. Galindo, S. Glondu, and M. Izabachène, "Election verifiability for helios under weaker trust assumptions," in *ESORICS*, Cham, 2014, pp. 327–344.

[30] B. Smyth, S. Frink, and M. R. Clarkson, *Election verifiability: Cryptographic definitions and an analysis of helios, helios-c, and jcj*, Cryptology ePrint Archive, Paper 2015/233, 2015.

[31] B. Adida, "Helios: web-based open-audit voting," in *Proceedings of the 17th conference on Security symposium*, USENIX, 2008, pp. 335–348.

[32] V. Cortier and B. Smyth, "Attacking and fixing helios: An analysis of ballot secrecy," in *24th Computer Security Foundations Symposium*, 2011, pp. 297–311. DOI: 10.1109/CSF.2011.27.

[33] M. Arapinis, V. Cortier, S. Kremer, and M. Ryan, "Practical everlasting privacy," in *Principles of Security and Trust*, vol. 7796 LNCS, 2013, pp. 21–40. DOI: 10.1007/978-3-642-36830-1_2.

[34] P. Grontas and A. Pagourtzis, "Anonymity and everlasting privacy in electronic voting," *Int. J. Inf. Sec.*, vol. 22, no. 4, pp. 819–832, 2023. DOI: 10.1007/S10207-023-00666-2.

[35] D. Pointcheval, "Efficient universally-verifiable electronic voting with everlasting privacy," in *SCN*, Cham: Springer Nature Switzerland, 2024, pp. 323–344, ISBN: 978-3-031-71070-4. DOI: 10.1007/978-3-031-71070-4_15.

[36] A. Fujioka, T. Okamoto, and K. Ohta, "A practical secret voting scheme for large scale elections," in *AUSCRYPT*, ser. LNCS, vol. 718, 1992, pp. 244–251. DOI: 10.1007/3-540-57220-1_66.

[37] P. Grontas, A. Pagourtzis, and A. Zacharakis, "Security models for everlasting privacy," *E-Vote-ID*, p. 140, 2019.

[38] D. Chaum, "Blind signatures for untraceable payments," in *Advances in Cryptology*, D. Chaum, R. L. Rivest, and A. T. Sherman, Eds., Boston, MA: Springer US, 1983, pp. 199–203, ISBN: 978-1-4757-0602-4.

[39] P. Grontas, A. Pagourtzis, A. Zacharakis, and B. Zhang, "Publicly auditable conditional blind signatures," *J. Comput. Secur.*, vol. 29, no. 2, 229–271, Jan. 2021, ISSN: 0926-227X. DOI: 10.3233/JCS-181270. [Online]. Available: https://doi.org/10.3233/JCS-181270.

[40] P. P. Tsang and V. K. Wei, "Short linkable ring signatures for e-voting, e-cash and attestation," in *Information Security Practice and Experience*, R. H. Deng, F. Bao, H. Pang, and J. Zhou, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 48–60, ISBN: 978-3-540-31979-5.

[41] A. Russo, A. F. Anta, M. I. G. Vasco, and S. P. Romano, "Chirotonia: A scalable and secure e-voting framework based on blockchains and linkable ring signatures," in *2021 IEEE International Conference on Blockchain (Blockchain)*, 2021, pp. 417–424. DOI: 10.1109/Blockchain53845.2021.00065.

[42] G. Chen, C. Wu, W. Han, X. Chen, H. Lee, *et al.*, "A new receipt-free voting scheme based on linkable ring signature for designated verifiers," in *2008 International Conference on Embedded Software and Systems Symposia*, 2008, pp. 18–23. DOI: 10.1109/ICESS.Symposia.2008.54.

[43] T. Haines and X. Boyen, "VOTOR: conceptually simple remote voting against tiny tyrants," in *Proceedings of the Australasian Computer Science Week Multiconference, Canberra, Australia, February 2-5, 2016*, ACM, 2016, p. 32.

[44] T. H. Yuen, M. F. Esgin, J. K. Liu, M. H. Au, and Z. Ding, "Dualring: Generic construction of ring signatures with efficient instantiations," in *Annual International Cryptology Conference*, Springer, 2021, pp. 251–281.

[45] J. Bootle, K. Elkhiyaoui, J. Hesse, and Y. Manevich, "Dualdory: Logarithmic-verifier linkable ring signatures through preprocessing," in *European Symposium on Research in Computer Security*, Springer, 2022, pp. 427–446.

[46] X. Hui and S. C.-K. Chau, "Llring: Logarithmic linkable ring signatures with transparent setup," in *European Symposium on Research in Computer Security*, Springer, 2024, pp. 299–319.

[47] D. Balla and P. Chaidos, "On the (strong) linkability of linkable ring signatures," *Cryptology ePrint Archive*, 2025.

[48] R. Araújo, S. Foulle, and J. Traoré, "A practical and secure coercion-resistant scheme for internet voting," in *Towards Trustworthy Elections: New Directions in Electronic Voting*, D. Chaum, M. Jakobsson, R. L. Rivest, P. Y. A. Ryan, J. Benaloh, *et al.*, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 330–342, ISBN: 978-3-642-12980-3. DOI: 10.1007/978-3-642-12980-3_20. [Online]. Available: https://doi.org/10.1007/978-3-642-12980-3_20.

[49] R. Araújo and J. Traoré, "A practical coercion resistant voting scheme revisited," in *E-Voting and Identify*, J. Heather, S. Schneider, and V. Teague, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 193–209, ISBN: 978-3-642-39185-9.

[50] V. Cortier, P. Gaudry, and Q. Yang, "Is the JCJ voting system really coercion-resistant?" In *CSF 2024*, ser. CSF 2024, This is the long version of the paper published at CSF 2024., Enschede, Netherlands: IEEE, 2024. [Online]. Available: https://inria.hal.science/hal-03629587.

[51] D. F. Aranha, M. Battagliola, and L. Roy, *Faster coercion-resistant e-voting by encrypted sorting*, Cryptology ePrint Archive, Paper 2023/837, 2023. [Online]. Available: https://eprint.iacr.org/2023/837.

[52] R. Giustolisi and M. S. Garjan, "Efficient cleansing in coercion-resistant voting," in *Electronic Voting*, D. Duenas-Cid, P. Roenne, M. Volkamer,

J. Budurushi, M. Blom, *et al.*, Eds., Cham: Springer Nature Switzerland, 2025, pp. 72–88, ISBN: 978-3-031-72244-8.

[53] W. Lueks, I. Querejeta-Azurmendi, and C. Troncoso, "Voteagain: A scalable coercion-resistant voting system," in *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, S. Capkun and F. Roesner, Eds., USENIX Association, 2020, pp. 1553–1570.

[54] T. Haines and J. Mueller, *How not to VoteAgain: Pitfalls of scalable coercion-resistant e-voting*, Cryptology ePrint Archive, Paper 2020/1406, 2020. [Online]. Available: https://eprint.iacr.org/2020/1406.

[55] I. Querejeta-Azurmendi, D. Arroyo Guardeño, J. L. Hernández-Ardieta, and L. Hernández Encinas, "Netvote: A strict-coercion resistance re-voting based internet voting scheme with linear filtering," *Mathematics*, vol. 8, no. 9, 2020, ISSN: 2227-7390. DOI: 10.3390/math8091618. [Online]. Available: https://www.mdpi.com/2227-7390/8/9/1618.

[56] G. S. Grewal, M. D. Ryan, S. Bursuc, and P. Y. A. Ryan, "Caveat coercitor: Coercion-evidence in electronic voting," in *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, IEEE Computer Society, 2013, pp. 367–381.

[57] D. Chaum, R. Carback, M. Yaksetig, J. Clark, M. Nejadgholi, *et al.*, *Votexx: Extreme coercion resistance*, Cryptology ePrint Archive, Paper 2024/1354, 2024. [Online]. Available: https://eprint.iacr.org/2024/1354.

[58] D. Chaum, R. Carback, J. Clark, C. Liu, M. Nejadgholi, *et al.*, "Revisiting silent coercion," in *Electronic Voting - 10th International Joint Conference, E-Vote-ID 2025, Nancy, France, October 1-3, 2025, Proceedings*, ser. Lecture Notes in Computer Science, vol. 16028, Springer, 2025, pp. 38–54.

[59] P. Locher, R. Haenni, and R. E. Koenig, "Coercion-resistant internet voting with everlasting privacy," in *FC'16 Workshops*, 2016. DOI: 10.1007/978-3-662-53357-4_11.

[60] P. Grontas, A. Pagourtzis, A. Zacharakis, and B. Zhang, "Towards everlasting privacy and efficient coercion resistance in remote electronic voting," in *FC'18 Workshops*, ser. LNCS, vol. 10958, 2018, pp. 210–231.

[61] E. Cuvelier, O. Pereira, and T. Peters, "Election verifiability or ballot privacy: Do we need to choose?" In *ESORICS*, 2013, pp. 481–498.

[62] T. Haines, R. Mosaheb, J. Müller, and I. Pryvalov, "Sok: Secure e-voting with everlasting privacy," *Proc. Priv. Enhancing Technol.*, no. 1, pp. 279–293, 2023. DOI: 10.56553/POPETS-2023-0017.

[63] J. Benaloh, M. Naehrig, and O. Pereira, "Reactive: Rethinking effective approaches concerning trustees in verifiable elections," in *Electronic Voting*, D. Duenas-Cid, P. Roenne, M. Volkamer, M. Blom, P. Gaudry, *et al.*, Eds., Cham: Springer Nature Switzerland, 2026, pp. 17–37, ISBN: 978-3-032-05036-6.

[64] R. Giustolisi, M. Rakeei, and G. Lenzini, *Anamorphic voting: Ballot freedom against dishonest authorities*, Cryptology ePrint Archive, Paper 2025/647, 2025. [Online]. Available: https://eprint.iacr.org/2025/647.

[65] G. Persiano, D. H. Phan, and M. Yung, "Anamorphic encryption: Private communication against a dictator," in *Advances in Cryptology – EUROCRYPT 2022: 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 – June 3, 2022, Proceedings, Part II*, Berlin, Heidelberg: Springer-Verlag, 2022, 34–63, ISBN: 978-3-031-07084-6. DOI: 10.1007/978-3-031-07085-3_2.

[66] F. Benhamouda, T. Lepoint, J. Loss, M. Orrù, and M. Raykova, "On the (in)security of ROS," in *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, A. Canteaut and F. Standaert, Eds., ser. Lecture Notes in Computer Science, vol. 12696, Springer, 2021, pp. 33–53. DOI: 10.1007/978-3-030-77870-5_2. [Online]. Available: https://doi.org/10.1007/978-3-030-77870-5_2.

[67] D. Achenbach, C. Kempka, B. Lowe, and J. Muller-Quade, "Improved Coercion-Resistant electronic elections through deniable Re-Voting," *USENIX Journal of Election Technology and Systems (JETS)*, vol. 3, no. 2, pp. 26–45, Aug. 2015, ISSN: 2328-2797. [Online]. Available: https://www.usenix.org/jets/issues/0302/achenbach.