Coercion-Resistant Internet Voting with Everlasting Privacy

Philipp Locher[1,2], Rolf Haenni[1], and Reto E. Koenig[1]

[1] Bern University of Applied Sciences, CH-2501 Biel, Switzerland
{philipp.locher,rolf.haenni,reto.koenig}@bfh.ch
[2] University of Fribourg, CH-1700 Fribourg, Switzerland
philipp.locher@unifr.ch

Abstract. The cryptographic voting protocol presented in this paper offers public verifiability, everlasting privacy, and coercion-resistance simultaneously. Voters are authenticated anonymously based on perfectly hiding commitments and zero-knowledge proofs. Their vote and participation secrecy is therefore protected independently of computational intractability assumptions or trusted authorities. Coercion-resistance is achieved based on a new mechanism for deniable vote updating. To evade coercion by submitting a final secret vote update, the voter needs not to remember the history of all precedent votes. The protocol uses two types of mix networks to guarantee that vote updating cannot be detected by the coercer. The input sizes and running times of the mix networks are quadratic with respect to the number of submitted ballots.

1 Introduction

Publishing the list of submitted ballots is a prerequisite for introducing public verifiability in electronic voting systems. Since the encrypted votes included in the ballots are protected by cryptographic techniques available today, there is no guarantee that the protection will withstand future advances in cryptanalysis and computational abilities. The secrecy of a vote submitted today is therefore not guaranteed to last forever. As a consequence, election organizers are often concerned about handing over the election data to everyone, even if this results in limiting the scope of public verifiability. Another serious concern for election organizers is the increased scalability of vote buying, bribery, or coercion attacks in an entirely digitalized environment. Providing a receipt to allow individual verifiability and not providing a receipt to disallow vote buying is a strong conflict in the design of electronic voting protocols.

1.1 Related Work

To the best of our knowledge, everlasting privacy and coercion-resistance have never been addressed together in a single cryptographic voting protocol. As each of them is a highly challenging problem on its own, offering them together seems to be nearly impossible. In the existing literature on everlasting privacy, the

adversary is assumed to possess unlimited computational power and an infinite amount of time to break the privacy of the votes. Some of the proposed solutions are designed for the traditional setting, in which ballots are cast in a private polling booth, whereas other protocols offer everlasting privacy for Internet elections with the aid of trusted authorities. In each of these proposals, a subset of colluding authorities could potentially break the privacy of the votes. The first protocol not relying on trusted authorities has been proposed recently by Locher and Haenni (LH15) [8]. They use an efficient set membership proof and a proof of knowledge of the representation of a committed value to achieve everlasting privacy. Their protocol is a direct predecessor of the protocol presented in this paper.

In the literature on coercion-resistance, there are two complementary strategies for a voter to evade coercion. In the protocol of Juels et al. (JCJ05) [7], the voter under coercion presents a fake credential to the adversary. The system is designed in a way that ballots submitted with a fake credential are silently eliminated during tallying using a quadratic number of plaintext equivalence tests (PET). The adversary model of JCJ05 allows voters to escape from adversarial control for a short moment during the voting period, which they can use for submitting a ballot using their true credentials. The second principal strategy against coercion is to let voters update their votes arbitrarily many times. In some protocols implementing this strategy, voters need to remember the history of all precedent votes when submitting the final ballot, which implies that simulation attacks cannot be prevented. In a more recent protocol by Achenbach et al. (AKLM15) [1], voters can submit a final ballot without remembering any previous votes and such that no coercer can learn whether vote updating has taken place or not. To achieve what they call *deniable vote updating*, they need trusted authorities performing jointly a quadratic number of encrypted plaintext equivalence tests (EPET). The adversary model of AKLM15 allows voters to escape from adversarial control at the end of the voting period, which is a slightly stronger assumption than in JCJ05. On the other hand, voters under coercion can follow the adversary's instructions without lying or concealing something.

1.2 Contribution

The contribution of this paper is a new cryptographic protocol for remote electronic voting. For voters not observed by an adversary before or during vote casting, it provides everlasting privacy without relying on trusted authorities or computational intractability assumptions. This means that no one will ever be able to break the secrecy of the vote or the secrecy of the voter's participation. The protocol also offers adequate protection against vote buying, bribery, and coercion attacks by polynomially bounded adversaries. As far as we know, this protocol is the first to offer everlasting privacy and coercion-resistance simultaneously.

The core of the protocol is composed of a set membership proof, a proof of known representation of a committed value, and a new tallying process that guarantees that no adversary can learn if particular votes have been updated or not. The proposed mechanism is based on two types of mix networks, which

are applied to a quadratic number of input encryptions. The shuffling destroys any link to the original list of submitted ballots, but at the same time preserves the information whether a given vote has been updated or not. The quadratic running time of the tallying procedure leads to a performance comparable to JCJ05 and AKLM15. Our approach is therefore not an efficient solution for large elections.

1.3 Paper Overview

We present our new protocol on two different levels of technical abstraction. In Section 2, we give a high-level overview of the approach by specifying the underlying adversary and trust model, and by discussing the resulting protocol properties. In Section 3, we introduce the cryptographic primitives, present the cryptographic details of the protocol, and provide a more precise discussion of the security properties. We summarize the findings of this paper in Section 4.

2 Coercion-Resistant Internet Voting with Everlasting Privacy

The approach presented in this paper is the first cryptographic voting protocol that offers verifiability, coercion-resistance, and everlasting privacy simultaneously. Three types of parties are involved in the protocol: an election administration, a group of trusted authorities, and the voters. They communicate over different communication channels. During registration, the protocol requires an authentic channel between voters and the election administration. Furthermore, a broadcast channel with memory—in the form of a robust append-only public bulletin board—is needed for collecting the election data. We assume that the election administration and the trusted authorities have their own designated areas on the bulletin board. Finally, for sending their votes to the bulletin board, voters need access to an anonymous channel. We assume that it is impossible to intercept and record the complete traffic over this channel during an election and storing the intercepted data for future use [2].

2.1 Adversary Model and Trust Assumptions

The general adversarial goals are to break the integrity or secrecy of the votes or to influence the election outcome via bribery or coercion. We consider active adversaries, which may interfere with the voting process at any point to reach their goals. To achieve coercion-resistance, we assume that a threshold number of authorities not colluding with the adversary is available for the tallying process. We also assume that no adversary can control the machines used during the voting process.[3]

[3] We are aware that requiring a secure platform is a strong and probably unrealistic assumption. We do not explicitly address this problem in this paper.

To discuss the aspect of everlasting privacy, we consider two types of adversaries with very different computational capabilities.

Present adversaries act before, during, or shortly after an election, i.e., within the cryptoperiod of the involved cryptographic keys. We assume present adversaries to be polynomially bounded and thus incapable of solving supposedly hard problems such as computing discrete logarithms in some large groups or breaking cryptographic primitives such as contemporary hash functions. Therefore, they cannot efficiently open computationally binding commitments or generate valid proof transcripts for zero-knowledge proofs without knowing the secret inputs. On the other hand, present adversaries may have the power and resources to bribe or coerce a large number of voters. A present adversary in our model is therefore equivalent to the adversary in JCJ05, except for the additional assumption that voters can escape adversarial control for submitting a final vote update. As discussed in AKLM15, this is a necessary pre-condition for offering coercion-resistance based on deniable vote updating.

Future adversaries may become active at any point in the future, i.e., strictly after the tallying phase of the election under attack. We assume that future adversaries cannot collude with present adversaries, for example by sharing information. On the other hand, we assume them to possess unlimited resources in terms of computational power and time. Clearly, contemporary cryptography will be completely useless in the presence of such an adversary, and any private keying material used in an election today will be revealed. However, the secrets hidden in perfectly hiding commitments or zero-knowledge proofs will never be revealed, even if they were generated today.

2.2 Protocol Overview

The protocol is a continuation of LH15. Trusted authorities are needed to guarantee fairness and to add coercion-resistance in form of deniable vote updating, but not for privacy. The same applies to computational intractability assumptions. They are only needed to prevent the creation of invalid ballots during vote casting and to allow voters to deny the submission of an updated vote, but not to protect privacy in the long run.

Like in LH15, the core of the protocol is a combination of a set membership proof and a proof of known representation of a committed value [3, 5]. When casting a vote, the voter provides a zero-knowledge proof of knowledge of the representation of one of the registered public voter credentials. The same voter may submit multiple ballots, but the tallying procedure guarantees that only the last vote counts. In this way, precedent votes can be overridden without remembering their history. To guarantee that vote updating is deniable, we use two different types of mix networks to unlink the votes of a given voter from the voter's public credential. The main challenge in this step is to detect and exclude updated votes in a verifiable way without leaking any information to a potential coercer. The entire voting procedure consists of four consecutive steps (the first two steps are identical and the third step is very similar to LH15):

Registration. The voter creates a pair of private and public credentials and sends the public credential over an authentic channel to the election administration.

Election Preparation. The election administration publishes the list of public voter credentials—one for every registered voter—on the public bulletin board.

Vote Casting. The voter creates an electronic ballot and sends it over an anonymous channel to the public bulletin board. The ballot consists of the encrypted vote, a commitment to the public credential, a homomorphic encryption of an election credential, and the above-mentioned composition of zero-knowledge proofs. The voter's public credential and the election credential are derived from the same private credential.

Tallying. The trusted authorities verify the proofs included in the submitted ballots and eliminate ballots with invalid proofs. For each remaining ballot, the authorities compute a list of ciphertexts with the following property: whenever the ballot has been updated, at least one of its plaintexts is equal to 1. The construction of this list is similar to AKLM15, but to sort out updated ballots, we first shuffle the list in a verifiable mix network. The shuffle applies under encryption a one-way function to all plaintexts different from 1. In this way, the shuffled list is unlinked from the original list, but the above property that an encryption of 1 is an indicator for an updated ballot is preserved. By attaching the encrypted vote to the resulting shuffled list, we obtain an intermediate ballot containing all necessary information to conclude the tallying process. The list of all intermediate ballots is shuffled in a verifiable re-encryption mix network to unlink them from the original ballots on the bulletin board. For each output ballot of this shuffle, the trusted authorities need to decide about including the ballot in the final tally. For this, they start decrypting the ciphertexts until a plaintext equal to 1 is revealed. If this happens, the ballot is sorted out. The encrypted votes included in the remaining ballots are decrypted and counted. To enable public verification, all steps performed by a trusted authority must be accompanied by non-interactive zero-knowledge proofs.

This protocol provides everlasting privacy for the same reasons as its predecessor protocol LH15. All the identifying information contained in a ballot is either a perfectly hiding commitment or a zero-knowledge proof. To provide coercion-resistance, a relatively complex tallying phase is necessary to sort out updated votes in a verifiable way, but such that no coercer can learn if a ballot has been updated or not. Further aspects of coercion-resistance are discussed in the next subsection. Note that the tallying procedure requires two mix networks, which are both applied to a quadratic number of input encryptions. The performance of the tallying procedure is therefore comparable to AKLM15.

2.3 Discussion of Coercion-Resistance

To protect an electronic voting system from adversaries trying to bribe or coerce voters, receipt-freeness is a necessary precondition. Intuitively, a receipt consists

of some auxiliary non-public information, which is sufficient for voters to prove towards a passive adversary how they voted. According to JCJ05, there are at least three additional coercive attacks, which receipt-freeness alone can not prevent. Voters could be forced to cast a random vote (randomization attack), to abstain from voting (forced-abstention attack), or to hand the private keying material over to the coercer (simulation attack).

Deniable vote updating as implemented in our protocol is an adequate countermeasure to coercion in general. Whatever a present adversary forces the voter into, the voter can extinguish the demands of the adversary by submitting secretly a final vote. Other than JCJ05, deniable vote updating is convincing by the fact that a voter can act exactly as demanded by the coercer without lying or pretending. In addition, as casting the last vote is independent of the history of votes submitted previously, the voter must not memorize any state. In other words, submitting a final vote will always erase any previous votes, even if they had been cast by the adversary. Erasing votes in this way remains undetected by the adversary, because the election credential added to the ballot is encrypted and obfuscated during the tallying phase. As a result, a present adversary will never succeed with a randomization, forced-abstention, or simulation attack.

A general problem of coercion-resistant systems such as JCJ, which are based the voter's ability to lie about some secret credential in the presence of the coercer, is the unintended use of a wrong credential. The resulting ballot will appear on the bulletin board and the voter can check its inclusion, but the vote will not be taken into account in the final tally. Since the system cannot respond with a warning in such a case, voters are unable to detect using a wrong credential. In a protocol based on deniable vote updating, the system can issue such a warning when votes are cast with a wrong credential. This is a remarkable difference when considering individual verifiability.

The everlasting privacy property of our protocol even prevents an additional coercive attack not discussed in JCJ05. A future adversary may try to coerce a voter by claiming to know how the voter has voted in the past and by threatening the voter with making it public (*"I know how you voted and I am going to tell everyone, unless..."*). In a protocol that offers everlasting privacy, this claim cannot be justified whatsoever.

3 Detailed Cryptographic Protocol

In this section, we present the cryptographic details of our new coercion-resistant protocol for electronic elections with everlasting privacy. We start with a short discussion of cryptographic preliminaries. Then we provide a detailed formal description of the protocol and analyse its security properties.

3.1 Cryptographic Preliminaries

Let $\mathcal{G}_p$ be a multiplicative cyclic group of prime order p, for which the DL assumption is believed to hold. Furthermore, let $\mathbb{G}_q \subset \mathbb{Z}_p^*$, be a large prime-order

subgroup of the group of integers modulo p. Finally, suppose that independent generators $g_0, g_1 \in \mathcal{G}_p$ and $h, h_0, h_1, ... \in \mathbb{G}_q$ are publicly known. Independence with respect to generators of a cyclic group means that their relative discrete logarithms are not known to anyone.

Homomorphic Commitments and Encryptions. In our protocol, we use two instances of the perfectly hiding Pedersen commitment scheme, one over $\mathcal{G}_p$ and one over $\mathbb{G}_q$. We distinguish them by $\text{com}_p(u, r) = g_0^r g_1^u$ for a commitment to $u \in \mathbb{Z}_p$ with randomization $r \in \mathbb{Z}_p$ and $\text{com}_q(v, s) = h_0^s h_1^v$ for a commitment to $v \in \mathbb{Z}_q$ with randomization $s \in \mathbb{Z}_q$. In the case of $\mathbb{G}_q$, we write $\text{com}_q(v_1, ..., v_n, s) = h_0^s h_1^{v_1} \cdots h_n^{v_n}$ for a commitment to n values $v_1, ..., v_n \in \mathbb{Z}_q$.

The protocol also requires an instance of an ElGamal encryption scheme over $\mathbb{G}_q$, where $x \in \mathbb{Z}_q$ is a shared private key and $y = h^x \in \mathbb{G}_q$ a public key. We write $E = \text{enc}_y(m, r) = (h^r, my^r) \in \mathbb{G}_q \times \mathbb{G}_q$ for encrypting a message $m \in \mathbb{G}_q$ with randomization $r \in \mathbb{Z}_q$ and $m = \text{dec}_x(E) = ba^{-x}$ for decrypting a ciphertext $E = (a, b)$ in a distributed way using the private key shares of x. We write $\mathbf{M} = \text{dec}_x(\mathbf{E}) = (m_1, ..., m_n)$ for decrypting a list of ciphertexts $\mathbf{E} = (E_1, ..., E_n)$. To re-encrypt a ciphertext E with a new randomization $r' \in \mathbb{Z}_q$, we use the standard procedure $E' = \text{reEnc}_y(E, r') = E \cdot \text{enc}_y(1, r')$ of multiplying E with an encryption of 1. We write $\mathbf{E}' = \text{reEnc}_y(\mathbf{E}, \mathbf{r}') = (E'_1, ..., E'_n)$ to re-encrypt a list of ciphertexts $\mathbf{E} = (E_1, ..., E_n)$ with new randomizations $\mathbf{r}' = (r'_1, ..., r'_n)$.

Zero-Knowledge Proofs. Our protocol relies strongly on various non-interactive zero-knowledge proofs of knowledge. A fundamental proof is the preimage proof $\mathit{NIZKP}[(a) : b = \phi(a)]$ for a one-way group homomorphism $\phi : X \to Y$, where $a = \phi^{-1}(b) \in X$ is the secret preimage of a public value $b \in Y$. Examples of such preimage proofs result from the above homomorphic commitment and encryption schemes, for example $\mathit{NIZKP}[(u, r) : C = \text{com}_p(u, r)]$ for proving knowledge of the opening of a Pedersen commitment, $\mathit{NIZKP}[(m, r) : E = \text{enc}_y(m, r)]$ for proving knowledge of the plaintext and randomization of an ElGamal ciphertext, or $\mathit{NIZKP}[(x) : \mathbf{M} = \text{dec}_x(\mathbf{E}) \wedge y = h^x]$ for proving knowledge of the private key used in the decryption of a list of ciphertexts.

The most common construction of a non-interactive preimage proof is the Σ-protocol in combination with the Fiat-Shamir heuristic. Proofs constructed in this way are perfect zero-knowledge in the random oracle model. Their transcript consists of one or multiple commitments and one or multiple responses to a challenge obtained from querying the random oracle with the public inputs and the commitments. In practice, the random oracle is implemented with a cryptographic hash function. In the protocol description, we will write $\pi = \mathit{NIZKP}[\cdot]$ for the transcripts of non-interactive proofs.

Set Membership Proof. Let $U = \{u_1 \ ..., u_N\}$ be a finite set of values $u_i \in \mathbb{Z}_p$ and $C = \text{com}_p(u, r)$ a commitment to an element $u \in U$. Both U and C are publicly known. With a *set membership proof*, denoted by

$$\mathit{NIZKP}[(u, r) : C = \text{com}_p(u, r) \wedge u \in U],$$

the prover demonstrates knowledge of corresponding values $u \in U$ and $r \in \mathbb{Z}_p$. A general way of constructing a set membership proof is to demonstrate that $P(u) = 0$ for the polynomial $P(X) = \prod_{i=1}^{N}(X - u_i)$. This proof, denoted by

$$NIZKP[(u, r) : C = \mathrm{com}_p(u, r) \wedge P(u) = 0],$$

is a particular case of a *polynomial evaluation proof*. In a recent publication, Bayer and Groth proposed a polynomial evaluation proof with a logarithmic size, which is the current state-of-the-art [5].

Proof of Known Representation. In a cyclic group such as $\mathbb{G}_q$ with generators $h_1, \ldots, h_n$, a tuple $(v_1, \ldots, v_n) \in \mathbb{Z}_q^n$ is called *DL-representation* (or simply *representation*) of $u \in \mathbb{G}_q$, if $u = h_1^{v_1} \cdots h_n^{v_n}$ [6]. For such a value $u \in \mathbb{G}_q \subset \mathbb{Z}_p$, let $C = \mathrm{com}_p(u, r)$ and $D = \mathrm{com}_q(v_1, \ldots, v_n, s)$ be publicly known commitments. Following Au et al. [3], a proof of known *representation of a committed value* (or simply *representation proof*), denoted by

$$\begin{aligned} NIZKP[(u, r, v_1, \ldots, v_n, s) : C = \mathrm{com}_p(u, r) \wedge \\ D = \mathrm{com}_q(v_1, \ldots, v_n, s) \wedge u = h_1^{v_1} \cdots h_n^{v_n}], \end{aligned}$$

demonstrates that the tuple of committed values in D is a DL-representation of the committed value in C.

Cryptographic Shuffle. The input of a cryptographic shuffle is a list $\mathbf{Z} = (z_1, \ldots, z_n)$ of input values $z_i \in Z$. The mixer applies a keyed one-way function $f : Z \times K \to Z$ to each input value z_i and permutes the results by picking a random permutation $\phi : [1, n] \to [1, n]$ from the set Φ_n of permutations of length n. The output of a cryptographic shuffle is therefore a list $\mathbf{Z'} = (z'_1, \ldots, z'_n)$ of values $z'_j = f(z_i, k_i)$ for indices $j = \phi(i)$ and keys $k_i \in K$. Additionally, the mixer proves the correctness of the shuffle using one of the existing techniques [4, 9]. We denote the two steps of this procedure by

$$(\mathbf{Z'}, \pi_{\mathbf{Z}}) = \mathrm{shuffle}_f^{\phi}(\mathbf{Z}, k_1, \ldots, k_n),$$

where $\pi_{\mathbf{Z}}$ is the transcript of the non-interactive zero-knowledge proof. To prevent that a single mixer must be fully trusted, the shuffling needs to be performed by multiple independent mixers in a mix network. The unlinkability between input and output is guaranteed as long as at least one permutation remains secret.

In our protocol, we need two instances of a cryptographic shuffle. In the first case, the input is a list $\mathbf{E} = (E_1, \ldots, E_n)$ of ElGamal ciphertexts $E_i \in \mathbb{G}_q \times \mathbb{G}_q$. For random values $\gamma_i \in_R \mathbb{Z}_q \setminus \{0\}$, the function $\exp(E_i, \gamma_i) = E_i^{\gamma_i}$ is applied to each input ciphertext E_i, which gives us $(\mathbf{E'}, \pi_{\mathbf{E}}) = \mathrm{shuffle}_{\exp}^{\phi}(\mathbf{E}, \gamma_1, \ldots, \gamma_n)$.[4] In this particular shuffle, both the ciphertexts and the plaintexts are unlinked from

[4] Note that $\gamma_i \neq 0$ is a crucial pre-condition to avoid trivial output ciphertexts $(1, 1)$. The verifier of $\pi_{\mathbf{E}}$ must therefore check $E_i \neq (1, 1)$ for every $E_i \in \mathbf{E}$ and reject the proof if one of the checks fails.

their original values in $\mathbf{E}$. There is only one exception: an encryption of 1 remains an encryption of 1.

In the second case, the input list $\mathbf{EE} = (\mathbf{E}_1, \dots, \mathbf{E}_n)$ contains n individual lists $\mathbf{E}_i = \{E_{i,1}, \dots, E_{i,n}\}$ of ElGamal ciphertexts, i.e., $\mathbf{EE}$ contains a total of n^2 ciphertexts $E_{i,j} \in \mathbb{G}_q \times \mathbb{G}_q$. For random values $\mathbf{r}'_i = (r'_{i,1}, \dots, r'_{i,n}) \in_R \mathbb{Z}_q^n$, the function $\text{reEnc}_y(\mathbf{E}_i, \mathbf{r}'_i)$ is applied to each input list $\mathbf{E}_i$. In other words, $(\mathbf{EE}', \pi_{\mathbf{EE}}) = \text{shuffle}^{\phi}_{\text{reEnc}_y}(\mathbf{EE}, \mathbf{r}'_1, \dots, \mathbf{r}'_n)$ re-encrypts all n^2 ciphertexts, but only the rows of the input $\mathbf{EE}$ are permuted, not the columns.

3.2 Protocol Description

As outlined in Section 2.2, the protocol consists of four consecutive phases. We will now present the details of each phase using the cryptographic primitives and formal notation introduced in the previous section. Summaries of all phases are included in corresponding figures at the end of each subsection. Note that the registration and election preparation phase are identical to the predecessor protocol in LH15, and vote casting is very similar. To achieve coercion-resistance, complexity has been added mainly to the tallying phase.

Registration. The first step of the protocol is the registration of voters before an election. To register, the voter picks a *private credential* $(\alpha, \beta) \in_R \mathbb{Z}_q \times \mathbb{Z}_q$ at random and computes the *public credential* $u = h_1^{\alpha} h_2^{\beta} \in \mathbb{G}_q$. Note that the private credential is a DL-representation of the public credential. Finally, the voter sends u over an authentic channel to the election administration.

Registration (Voter):

1. Pick private credential $(\alpha, \beta) \in_R \mathbb{Z}_q \times \mathbb{Z}_q$.
2. Compute public credential $u = h_1^{\alpha} h_2^{\beta} \in \mathbb{G}_q$.
3. Send u over an authentic channel to the election administration.

Fig. 1: Summary of the registration phase.

Election Preparation. After the registration phase, the election administration defines the list $\mathbf{U} = ((V_1, u_1), \dots, (V_N, u_N))$ based on the electoral roll. Each pair $(V_i, u_i) \in \mathbf{U}$ links a public credential u_i to the corresponding voter identity V_i. Next, the list $\mathbf{A} = (a_0, \dots, a_N)$ of coefficients $a_i \in \mathbb{Z}_p$ of the polynomial $P(X) = \prod_{i=1}^{N}(X - u_i) \in \mathbb{Z}_p[X]$ is computed to allow voters the creation of the set membership proof during vote casting.[5] Finally, an independent *election*

[5] As the computation of the coefficients is quite expensive ($\frac{1}{2}N^2$ multiplications in $\mathbb{Z}_p$), it is performed by the election administration, possibly already during the registration phase in an incremental way. Note that the coefficients can be re-computed and verified by anyone, and voters can efficiently verify the inclusion of their public credential u by checking $P(u) = 0$.

generator $\hat{h} \in \mathbb{G}_q$ is defined in some publicly reproducible way and $(\mathbf{U}, \mathbf{A}, \hat{h})$ is posted into the administration's designated area of the public bulletin board.

Election Preparation (Election Administration):

1. Define $\mathbf{U} = ((V_1, u_1), \ldots, (V_N, u_N))$ based on the electoral roll.
2. Compute coefficients $\mathbf{A} = (a_0, \ldots, a_N)$ of $P(X) = \prod_{i=1}^{N}(X - u_i) \in \mathbb{Z}_p[X]$.
3. Define election generator $\hat{h} \in \mathbb{G}_q$.
4. Post $(\mathbf{U}, \mathbf{A}, \hat{h})$ into the designated area of the bulletin board.

Fig. 2: Summary of the election preparation phase.

Vote Casting. During the election, voters select their vote by choosing their preferred election options and encoding them by an element of the set $\mathbb{V} \subset \mathbb{G}_q$ of valid votes. We assume that the election options, their encoding in $\mathbb{V}$, and the public key y of the trusted authorities are publicly known.

To cast a vote, the voter computes a commitment $C = \mathrm{com}_p(u, r)$ of the public credential and a commitment $D = \mathrm{com}_q(\alpha, \beta, s)$ of the private credentials. Next, the voter computes an encryption $E = \mathrm{enc}_y(\hat{h}^\beta, \rho)$ of the *election credential* $\hat{h}^\beta \in \mathbb{G}_q$ and an encryption $F = \mathrm{enc}_y(v, \sigma)$ of the encoded vote $v \in \mathbb{V}$. Finally, the voter generates three non-interactive zero-knowledge proofs. The first proof,

$$\pi_1 = \mathit{NIZKP}[(u, r) : C = \mathrm{com}_p(u, r) \wedge P(u) = 0],$$

is a set membership proof proving that C is indeed a commitment to the public credential of one of the eligible voters listed in $\mathbf{U}$. The second proof,

$$\pi_2 = \mathit{NIZKP}[(u, r, \alpha, \beta, s) : C = \mathrm{com}_p(u, r) \wedge D = \mathrm{com}_q(\alpha, \beta, s) \wedge u = h_1^\alpha h_2^\beta],$$

is a proof of known representation of the committed value in C. It prevents voters from taking someone else's credential from $\mathbf{U}$. Finally, the third proof, $\pi_3 =$

$$\mathit{NIZKP}[(\alpha, \beta, s, \rho, v, \sigma) : D = \mathrm{com}_q(\alpha, \beta, s) \wedge E = \mathrm{enc}_y(\hat{h}^\beta, \rho) \wedge F = \mathrm{enc}_y(v, \sigma)],$$

demonstrates that D and E have been generated using the same value β and that the vote contained in F is known to the voter. The two commitments, the two ciphertexts, and the three proofs form the ballot $B = (C, D, E, F, \pi_1, \pi_2, \pi_3)$, which is posted to the bulletin board over an anonymous channel. The voter may submit multiple such ballots during the election period. If multiple identical copies of the same ballot are posted to the bulletin board, we assume that only one of them is stored.[6]

[6] The bulletin board could also accept multiple copies of the same ballot, which then need to be eliminated in the tallying phase. But this makes preventing replay and board flooding attacks more complicated.

Vote Casting (Voter):

1. Select vote $v \in \mathbb{V}$.
2. Pick $r \in_R \mathbb{Z}_p$ and compute $C = \mathrm{com}_p(u, r) \in \mathcal{G}_p$.
3. Pick $s \in_R \mathbb{Z}_q$ and compute $D = \mathrm{com}_q(\alpha, \beta, s) \in \mathbb{G}_q$.
4. Pick $\rho \in_R \mathbb{Z}_q$ and compute $E = \mathrm{enc}_y(\hat{h}^\beta, \rho) \in \mathbb{G}_q \times \mathbb{G}_q$.
5. Pick $\sigma \in_R \mathbb{Z}_q$ and compute $F = \mathrm{enc}_y(v, \sigma) \in \mathbb{G}_q \times \mathbb{G}_q$.
6. Compute non-interactive proofs:

$$\pi_1 = \mathit{NIZKP}[(u, r) : C = \mathrm{com}_p(u, r) \wedge P(u) = 0],$$
$$\pi_2 = \mathit{NIZKP}[(u, r, \alpha, \beta, s) : C = \mathrm{com}_p(u, r) \wedge D = \mathrm{com}_q(\alpha, \beta, s) \wedge u = h_1^\alpha h_2^\beta],$$
$$\pi_3 = \mathit{NIZKP}[(\alpha, \beta, s, \rho, v, \sigma) : D = \mathrm{com}_q(\alpha, \beta, s) \wedge E = \mathrm{enc}_y(\hat{h}^\beta, \rho) \wedge F = \mathrm{enc}_y(v, \sigma)].$$

7. Post $B = (C, D, E, F, \pi_1, \pi_2, \pi_3)$ to the bulletin board over an anonymous channel.

Fig. 3: Summary of the vote casting phase.

Tallying. At the end of the election period, the ballots submitted to the bulletin board need to be processed by the trusted authorities. We present this process by looking at the group of trusted authorities as a single entity performing the necessary shuffling and decryption tasks jointly. In reality, different trusted authorities will perform respective tasks using their own secret inputs and random values. The cryptographic shuffling is a serial and the distributed decryption (usually) a parallel process.

To initiate the tallying process, the trusted authority retrieves the list $\mathbf{B}$ of all ballots from the bulletin board. We assume that the ballots in $\mathbf{B}$ are ordered according to their submission. The authority verifies the non-interactive proofs π_1, π_2, π_3 for each ballot $(C, D, E, F, \pi_1, \pi_2, \pi_3) \in \mathbf{B}$, and ballots with invalid proofs are eliminated. From all ballots with valid proofs, the two ciphertexts (E, F) are selected. We denote the resulting ordered list of such pairs by $\mathbf{E} = ((E_1, F_1), \ldots, (E_n, F_n))$ and assume that $\mathbf{E}$ is ordered according to $\mathbf{B}$. This implies for all $j > i$ that (E_j, F_j) has been cast after (E_i, F_i). Furthermore, the validity of the proofs guarantees that each $(E_i, F_i) \in \mathbf{E}$ originates from a person in possession of valid private credentials. Finally, we know that two distinct pairs $(E_i, F_i), (E_j, F_j) \in \mathbf{E}$ belong to the same private credentials, whenever E_i and E_j contain the same plaintext.

In the next step, the trusted authority computes for each E_i a list $\mathbf{E}_i = (E_{i,1}, \ldots, \ldots, E_{i,n-1})$ of ciphertexts

$$E_{i,j} = \begin{cases} E_j & \text{for } j < i, \\ E_{j+1}/E_i & \text{for } j \geq i. \end{cases}$$

Note that $\mathbf{E}_i$ may contain one or multiple encryptions of 1, but only if some $E_j \in \{E_{i+1}, \dots, E_n\}$ contain the same plaintext as E_i. If this is the case, then (E_i, F_i) has been updated and needs to drop out at some point. To determine the updated votes without decrypting $\mathbf{E}_i$, the authority first performs a cryptographic shuffle

$$(\mathbf{E}'_i, \pi_{\mathbf{E}_i}) = \text{shuffle}^{\phi_i}_{\text{exp}}(\mathbf{E}_i, \gamma_{i,1}, \dots, \gamma_{i,n-1})$$

on each $\mathbf{E}_i$, where $\phi_i \in_R \Phi_{n-1}$ is a random permutation and $\gamma_{i,j} \in_R \mathbb{Z}_q \setminus \{0\}$ are random exponents. The goal of this shuffle is to conceal any plaintext different from 1. Let $\mathbf{E}'_i = (E'_{i,1}, \dots, E'_{i,n-1})$ be the result of this shuffle and $\mathbf{F}_i = (F_i, E'_{i,1}, \dots, E'_{i,n-1})$ the extension of this list by inserting F_i at the front. For $\mathbf{FF} = (\mathbf{F}_1, \dots, \mathbf{F}_n)$, the authority performs an additional cryptographic shuffle

$$(\mathbf{FF}', \pi_{\mathbf{FF}}) = \text{shuffle}^{\phi}_{\text{reEnc}_{y,}}(\mathbf{FF}, \mathbf{r}'_1, \dots, \mathbf{r}'_n),$$

for a random permutation $\phi \in_R \Phi_n$ and re-encryption randomizations $\mathbf{r}'_i = (r'_{i,1}, \dots, r'_{i,n}) \in \mathbb{Z}_q^n$. The purpose of this shuffle is to remove the link to the original ballots. Let $\mathbf{FF}' = (\mathbf{F}'_1, \dots, \mathbf{F}'_n)$ be the result of this shuffle and $\mathbf{F}'_i = (F'_i, E''_{i,1}, \dots, E''_{i,n-1})$ a single entry of $\mathbf{FF}'$. To determine whether F'_i must be excluded from the final tally, the authority checks if $\text{dec}_x(E''_{i,j}) = 1$ holds for some $j \in [1, n-1]$. Let $U \subseteq [1, n]$ be the subset of indices i for which this is the case, and $V = [1, n-1] \setminus U$ the subset of indices for which this is not the case.[7] For every $i \in U$, the authority selects from $\mathbf{F}'_i = (F'_i, E''_{i,1}, \dots, E''_{i,n-1})$ one of the encryptions $E''_{i,j}$ containing 1 as plaintext and computes a non-interactive proof

$$\tilde{\pi}_i = \mathit{NIZKP}[(x) : 1 = \text{dec}_x(E''_{i,j}) \wedge y = h^x].$$

For every $i \in V$, the authority computes $\mathbf{V}_i = \text{dec}_x(\mathbf{F}'_i)$ along with a non-interactive proof

$$\hat{\pi}_i = \mathit{NIZKP}[(x) : \mathbf{V}_i = \text{dec}_x(\mathbf{F}'_i) \wedge y = h^x].$$

The final tally is obtained by checking if the plaintext votes at the first position in every $\mathbf{V}_i$ are elements of $\mathbb{V}$ and by summing them up if this is the case. To complete the tallying process, the trusted authority posts

$$(\mathbf{E}, \{\mathbf{E}_i, \mathbf{E}'_i, \pi_{\mathbf{E}_i}\}_{i=1}^n, \mathbf{FF}, \mathbf{FF}', \pi_{\mathbf{FF}}, U, \{E''_{i,j}, \tilde{\pi}_i\}_{i \in U}, V, \{\mathbf{V}_i, \hat{\pi}_i\}_{i \in V})$$

to the designated area of the public bulletin board.

3.3 Security Properties

We will now look at our protocol from the perspective of its security properties. We provide an informal discussion of how correctness, everlasting privacy, and coercion-resistance are achieved. Fairness is achieved in a trivial way by submitting votes encrypted.

[7] Think of U and V as the indices of the *updated* and *valid* votes, respectively.

Tallying (Trusted Authority):

1. Retrieve the list $\mathbf{B}$ of all ballots from the bulletin board.
2. For each $(C, D, E, F, \pi_1, \pi_2, \pi_3) \in \mathbf{B}$, verify π_1, π_2, π_3. Select the pairs (E, F) from ballots with valid proofs. Let $\mathbf{E} = ((E_1, F_1), \dots, (E_n, F_n))$ denote the list of such pairs.
3. Compute $\mathbf{FF} = (\mathbf{F}_1, \dots, \mathbf{F}_n)$ by applying to following steps to $1 \leq i \leq n$:
   (a) Compute $\mathbf{E}_i = (E_1, \dots, E_{i-1}, E_{i+1}/E_i, \dots, E_n/E_i)$.
   (b) Pick $\phi_i \in_R \Phi_{n-1}$ and $\gamma_{i,j} \in_R \mathbb{Z}_q \setminus \{0\}$.
   (c) Compute $(\mathbf{E}'_i, \pi_{\mathbf{E}_i}) = \text{shuffle}^{\phi_i}_{\text{exp}}(\mathbf{E}_i, \gamma_{i,1}, \dots, \gamma_{i,n-1})$.
   (d) For $\mathbf{E}'_i = (E'_{i,1}, \dots, E'_{i,n-1})$, let $\mathbf{F}_i = (F_i, E'_{i,1}, \dots, E'_{i,n-1})$.
4. Pick $\phi \in_R \Phi_e$ and $\mathbf{r}'_i = (r'_{i,1}, \dots, r'_{i,n}) \in_R \mathbb{Z}_q^n$.
5. Compute $(\mathbf{FF}', \pi_{\mathbf{FF}}) = \text{shuffle}^{\phi}_{\text{reEnc}_y}(\mathbf{FF}, \mathbf{r}'_1, \dots, \mathbf{r}'_n)$.
6. Let $U = \{i \in [1, n] : \exists j \in [1, n-1] \text{ s.t. } \text{dec}_x(E''_{i,j}) = 1\}$.
7. For every $i \in U$:
   (a) Select $E''_{i,j}$ from $\mathbf{F}'_i = (F'_i, E''_{i,1}, \dots, E''_{i,n-1})$ such that $\text{dec}_x(E''_{i,j}) = 1$.
   (b) Compute $\tilde{\pi}_i = \mathit{NIZKP}[(x) : 1 = \text{dec}_x(E''_{i,j}) \wedge y = h^x]$.
8. For every $i \in V = [1, n] \setminus U$:
   (a) Compute $\mathbf{V}_i = \text{dec}_x(\mathbf{F}'_i)$.
   (b) Compute $\hat{\pi}_i = \mathit{NIZKP}[(x) : \mathbf{V}_i = \text{dec}_x(\mathbf{F}'_i) \wedge y = h^x]$.
9. Post $(\mathbf{E}, \{\mathbf{E}_i, \mathbf{E}'_i, \pi_{\mathbf{E}_i}\}_{i=1}^n, \mathbf{FF}, \mathbf{FF}', \pi_{\mathbf{FF}}, U, \{E''_{i,j}, \tilde{\pi}_i\}_{i \in U}, V, \{\mathbf{V}_i, \hat{\pi}_i\}_{i \in V})$ into the designated area of the bulletin board.

Fig. 4: Summary of the tallying phase with a single trusted authority.

Correctness. For a present adversary not colluding with any of the trusted authorities and not in possession of a private credential, there are two principle ways of creating a ballot that will be accepted in the final tally. First, the adversary may try to find (α', β') such that $u = h_1^{\alpha'} h_2^{\beta'}$ for some u in U, which is equivalent to solving the discrete logarithm problem. Second, the adversary may try to fake a proof transcript without knowing such a pair (α', β'), but this is prevented by the computational soundness of π_1, π_2, and π_3.

If the present adversary is an eligible voter in possession of a valid private credential, then using it for submitting more than one ballot is explicitly allowed by the protocol, but only the last ballot is considered in the final tally. The malicious voter could try to submit ballots with different election credentials, but the soundness of π_3 does not allow this. Without using the private credential, the voter is not more powerful than any other present adversary.

A present adversary colluding with one or several trusted authorities—or even the authorities themselves—may try to delete, modify, or add votes in the mixing or decryption steps of the protocol, but this is prevented by the computational soundness of $\pi_{\mathbf{E}_i}$, $\pi_{\mathbf{FF}}$, $\tilde{\pi}_i$, and $\hat{\pi}_i$. Their correctness can be verified by anyone.

Everlasting Privacy. A ballot posted over an anonymous channel to the bulletin board contains no information for identifying the voter. Clearly, the future

adversary will be able to determine the private key x, use it to decrypt E_i into $\hat{h}^\beta$, and finally obtain β. As u can be regarded as a perfectly hiding commitment to β, a suitable value α' can be found for every credential u' in $\mathbf{U}$ such that $u' = h_1^{\alpha'} h_2^\beta$. Thus, knowing β and x does not link $E_i = \mathrm{enc}_y(\hat{h}^\beta, \rho)$ to u. Since the proofs π_1, π_2, and π_3 are zero-knowledge and therefore of no additional help, even a future adversary is unable to break vote or participation secrecy.

Coercion-Resistance. A voter—either voluntarily or under coercion—may prove the authorship of a ballot by disclosing the randomizations used in the encryptions E_i and F_i. From this, the coercer learns the values $\hat{h}^\beta$ and v of a submitted ballot. To issue a conclusive receipt, the voter must also prove that the ballot is indeed included in the final tally, for example by proving that every subsequent ballot has been cast by somebody else. But this is impossible as the voter cannot prove *not* to know corresponding randomizations. Alternatively, the voter may try to show that $\{E_{i+1}, \ldots, E_n\}$ does not contain an encryption of $\hat{h}^\beta$ (or equivalently that $\mathbf{E}_i$ does not contain an encryption of 1) or to establish a link between $\mathbf{F}_i \in \mathbf{FF}$ and $\mathbf{F}'_{\phi(i)} \in \mathbf{FF}'$. Both tasks either require corrupting a majority of trusted authorities or solving the DDH or DL problem. Hence, the protocol is receipt-free under ordinary trust or computational intractability assumptions.

Attacks by an active coercer can be countered by the fact that voters cannot be urged to prove or disprove having cast a final vote in privacy (see reasoning above). A voter under a randomization attack will therefore follow the coercer's instructions and cast a random vote, but then the voter will submit a final vote in privacy and deny the vote update towards the coercer. For a voter under a forced-abstention attack, who will simply submit a final vote in privacy, everlasting participation secrecy is a perfect protection towards the coercer trying to check the voter's compliance. Finally, even if the private credentials α and β are handed over to the coercer in a simulation attack, the voter will always be able use the credentials for submitting a final vote in privacy and deny it towards the coercer. The coercer may try to check if vote updating using the same credentials has taken place, but this is impossible for the reasons explained above and because the commitments are perfectly hiding.

4 Conclusion

In this paper, we introduced the first cryptographic voting protocol offering everlasting privacy and coercion-resistance simultaneously. Everlasting privacy is realized with perfectly hiding commitments and zero-knowledge proofs of knowledge, and hence does not depend on trusted authorities or computational intractability assumptions. To achieve coercion-resistance, we propose a new deniable updating mechanism based on a combination of cryptographic mixing procedures. Computational intractability assumptions are obviously required for cryptographic mixing, but this is only problematical if the extent of a coercion attack exceeds the cryptoperiod of the chosen cryptographic setting. Attacks against vote or participation secrecy will always remain impossible.

The main drawback of our protocol is the quadratic running time of the tallying procedure. Compared to LH15, which requires $O(n \log N)$ exponentiations and $O(nN)$ multiplications in $\mathcal{G}_p$ to verify all submitted ballots, we require $O(n^2)$ additional exponentiations in $\mathbb{G}_q$. The performance is therefore similar to AKLM15, i.e., the applicability of our approach is restricted to relatively small electorates. Implementing our new protocol and analysing its performance to estimate the maximal possible electorate size is subject of further research.

Some problems remain unsolved in the current version of our protocol. An open issue is the problem of flooding the bulletin board with a very large number of valid ballots. This problem is a direct consequence of deniable vote updating. Another open issue is the problem of a malicious voting platform.

Acknowledgments. We thank the anonymous reviewers for their thorough reviews and appreciate their comments and suggestions. This research has been supported by the Swiss National Science Foundation (project No. 200021L_140650).

References

1. Achenbach, D., Kempka, C., Löwe, B., Müller-Quade, J.: Improved coercion-resistant electronic elections through deniable re-voting. USENIX Journal of Election Technology and Systems (JETS) (2), 26–45 (2015)
2. Arapinis, M., Cortier, V., Kremer, S., Ryan, M.: Practical everlasting privacy. POST'13, 2nd Conference on Principles of Security and Trust. pp. 21–40 (2013)
3. Au, M.H., Susilo, W., Mu, Y.: Proof-of-knowledge of representation of committed value and its applications. ACISP'10, 15th Australasian Conference on Information Security and Privacy. pp. 352–369 (2010)
4. Bayer, S., Groth, J.: Efficient zero-knowledge argument for correctness of a shuffle. EUROCRYPT'12, 31st Annual International Conference on Theory and Applications of Cryptographic Techniques. pp. 263–280 (2012)
5. Bayer, S., Groth, J.: Zero-knowledge argument for polynomial evaluation with application to blacklists. EUROCRYPT'13, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 646–663 (2013)
6. Brands, S.: Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press (2000)
7. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. WPES'05, 4th Workshop on Privacy in the Electronic Society. pp. 61–70 (2005)
8. Locher, P., Haenni, R.: Verifiable Internet elections with everlasting privacy and minimal trust. VoteID'15, 5th International Conference on E-Voting and Identity. pp. 74–91 (2015)
9. Terelius, B., Wikström, D.: Proofs of restricted shuffles. AFRICACRYPT'10, 3rd International Conference on Cryptology in Africa. pp. 100–113 (2010)