Enabling Vote Delegation for Boardroom Voting

Oksana Kulyk[1], Stephan Neumann[1], Karola Marky[1], Melanie Volkamer[1,2]

[1] Technische Universität Darmstadt, Darmstadt, Germany
name.surname@secuso.org
[2] Karlstad University, Karlstad, Sweden

Abstract. A lot of decisions are made during boardroom meetings. After a discussion, the head of the board often asks for a quick poll. But what if you cannot join the meeting? So called boardroom voting schemes have been proposed to conduct the poll over the Internet and thereby enabling also those who are not present but available online to participant in the poll. But what if you are not available at this point in time? For important decisions you may want to delegate your vote to a present and trusted board member. In this paper, we show how to extend an existing boardroom voting scheme towards delegation functionality. The new scheme is evaluated against security requirements determined for boardroom voting and security requirements tailored to the delegation process.

1 Introduction

Boardroom voting schemes are one of the new research directions in electronic voting and a number of approaches have been proposed [2,12–14,16,20,27]. Some of these approaches provide the possibility to participate in the polls remotely. However, time and geographical restrictions often prevent absent board members from participating in the poll. Consequently, decisions are often not supported by a required quorum. For such situation, it is worth considering the possibility to delegate ones' vote to a trusted board member that is present, the so-called *proxy*.

We extend the boardroom voting approach described in [20] to enable delegation. To do so we introduce so so-called *delegation token*, that is sent by the voter who wants to delegate. The rest of the board members get a random value that is indistinguishable from the authorised delegation token. The authorised delegation token is also distributed via secret sharing among all the of board members. In this way, during the voting the board members can validate the votes cast with an authorised delegation token, without revealing the identity of the board member who received the authorised delegation token, and thus was trusted by the voter to cast a delegated ballot on her behalf.

Note that an alternative solution would be to let the voters jointly establish a a public-key encryption system used in the election before the election, so that the delegating voters could use it to encrypt their delegation tokens, and the majority of present voters could use the corresponding private key in order to

decrypt the data sent within an election. However, our delegated voting scenario assumes, that all the setup that is required for a specific election occurs within the meeting, since it is not always practical to demand that the present board members gather together in advance to conduct election setup before each election.Hence, the way for the absent voters to delegate to a trusted board member of her choice prior to the election is needed, which is achieved by our proposal. Therefore,

The paper is structured as follows. We describe the requirements on delegated voting found in the literature in Section 2 and the background for our scheme in Section 3. We describe our solution for delegating in boardroom voting setting in Section 4, followed by the security evaluation in Section 5. We describe the related work in Section 6 and conclude in Section 7.

2 Security Requirements

For the security requirements that are related to *direct voting* as opposed to delegating, we rely on the list of security requirements for boardroom voting given in [20][3].

Voter eligibility. Only the votes from eligible voters, and only one vote from each voter, should be included in the result.

Vote integrity. Each cast vote (direct or delegated) of an eligible voter should be correctly included in the tally.

Robustness. After the vote casting has finished, the election result can be computed even in case where some of the scheme components are faulty.

Vote privacy. The voting scheme should not provide any more information that enables establishing a link between the honest voter[4] and her vote (delegated or direct) aside from the information that would be output by the ideal proxy boardroom voting scheme.

Fairness. The voting scheme should not reveal partial election results before all the votes have been cast.

The security requirements on the *delegation process* are based on the available literature concerning delegated voting [18,19,30]. Note we provide informal definitions for these requirements, since no formalization of the requirements on delegation process either in form of legal or technical definitions exist.

---

[3] As opposed to [20], similar to the proposal in [5] we consider verifiability to be a part of integrity, and not a separate requirement. We furthermore consider uniqueness a part of eligibility

[4] We refer to a voter or a proxy as *honest* if she behaves according to the scheme specification.

Delegation eligibility. Only the delegated ballots on behalf of eligible voters, and only one delegated ballot per voter, should be included in the tally.

Delegation integrity. No proxy can vote on the voter's behalf unless authorised by the voter.

Delegation privacy. The voting scheme should not provide any information to the public or to the proxies themselves, that identifies whether a particular voter has delegated to a particular proxy.[5]. We further require that the voter should not be able to use her private data from the election to construct the proof for delegating to a specific proxy herself by divulging this data to an adversary.

Delegation power privacy. The voting scheme should not reveal, how many voters have delegated to a specific proxy.

3 Background

In this section we describe the background required for our scheme.

3.1 Cryptographic Primitives

The public-key *ElGamal* cryptosystem [11] is used in our scheme for encrypting the cast votes and other data that is exchanged during the election.

In order to prove the validity of statements in our scheme without revealing any information beyond that, *zero-knowledge proofs* are used. Namely, we will use the methods described in [6] in order to construct the zero-knowledge proofs of statements about discrete logairthms. In this, we use the following notation in our paper: for example, given the public values g, h, y_1, y_2, the notation $PoK\{x_1, x_2 : y_1 = g^{x_1} \wedge y_2 = g^{x_2}h^{x_1}\}$ denotes the proof of knowledge of secret values x_1, x_2 so that $y_1 = g^{x_1}$ and $y_2 = g^{x_2}h^{x_1}$ holds.

The non-interactive version for zero-knowledge proofs is computed using the strong Fiat-Shamir heuristic as described in [4] with $\mathcal{H}$ as a cryptographic hash function.

For the delegation process, our scheme relies on distribution of secret delegation tokens among multiple voters, which is done using *threshold secret sharing*. This secret can then be reconstructed only if at least the threshold of all the voters collaborates. A number of proposals have been made for this purpose, with our scheme relying on a proposal in [26][6].

---

[5] Note, that the relevancy of this requirement might be debated, since in some cases it is reasonable to assume, that other boardroom members know whom the delegating voter trusts anyway. Still, we choose to include this requirement for the case, when the voter does not wish to publicly disclose his support for a particular proxy to others, or even to the proxy herself.

[6] While there are extensions of [26] that ensure verifiability to protect against dishonest dealers (e.g. [24]), this protection is not required in our scheme, since the dealer has no incentive to cheat during secret sharing. Hence, for the sake of simplicity we chose to use a less complex variant.

Another application of threshold secret sharing techniques in our scheme is the *distributed generation of a public ElGamal key* used for encryption within the election, and a set of private key shares distributed among the voters. The distributed key generation occurs as described in [23]. The threshold t is being chosen so that the collaboration of at least t voters is required to reconstruct the secret key or decrypt a ciphertext. The secret key shares are then used to distributively decrypt the encrypted data by applying the scheme as described in [23]. A zero-knowledge proof is used to prove the decryption validity.

In order to commit to a value without revealing it, in our scheme we use *Pedersen commitments* [24], that are unconditionally hiding and computationally binding under the assumption that the discrete logarithm is hard in the chosen group.

In order to anonymize a list of ElGamal ciphertexts by removing the link between each ciphertext and its sender, a *re-encryption mix net* scheme is being applied. The mix net consists of several mix nodes, in which each mix node shuffles the ciphertexts in turn, and provides the output as an input to the next mix node. It holds that the ciphertexts are anonymized, as long as at least one mix node does not reveal its secret values used for shuffling. For the sake of preventing manipulation (i.e. that no ciphertexts have been replaced during the shuffling), various methods for proving the validity of the shuffle have been proposed, such as [3, 29].

In order to check, whether two ElGamal ciphertexts e_1, e_2 encrypt the same value, without revealing any other information about corresponding plaintexts, we use the technique described in [15], the *plaintext equivalence test (PET)*, denoted as $\mathsf{PET}(e_1, e_2)$. This test is distributively performed by the participants who hold the shares of a secret decryption key and outputs 1 in case e_1 and e_2 encrypt the same plaintext, or a random value otherwise.

3.2 Boardroom Voting Scheme from [20]

We briefly describe the boardroom voting scheme proposed in [20] and used as a basis for our proxy boardroom voting extension. The scheme in [20] is based upon the proposal in [8] and adjusted towards decentralized setting, where the voters take over the role of the trustees. The election runs as follows.

Setup. Prior to the election, the voters exchange their public signing keys in order to enable authenticated message exchange. For this purpose they conduct the decentralized key exchange as described in [22], so that the correctness of the exchanged keys is established by manual verification of the so-called short authentication strings via an out-of-band channel (e.g. phone conference or physical proximity). After exchanging the signing keys, each pair of voters runs the Diffie-Hellman key [10] exchange in order to establish the symmetric secret keys for private communication. After the public signing keys and the symmetric secret keys have been established, the voters run the distributed threshold secret sharing as described in [23] in order to generate a public election key pk and share the corresponding private election key sk.

Voting. Once the election key has been generated, one voter takes over the role of the election organizer and initilizes the voting. In order to vote, each voter encrypts her chosen voting option with the public election key pk and broadcasts it as her ballot to other voters.

Tallying. Once all the ballots have been cast, the voter jointly perform the tallying. For this, they shuffle the ballots with a verifiable re-encryption mix net [29], where each voter acts as a mix node. After the shuffling, the voters jointly run the distributed threshold decryption as described in [23] in order to reveal the election result.

4 Our Scheme

We are now ready to provide a description of our scheme for proxy voting in boardroom voting setting. We assume the existence of a trustworthy public-key infrastructure among all eligible voters, established either via decentralized key exchange as in [20] or in any other appropriate way[7]. Furthermore, the PKI is used to establish private communication channels between the voters, and a reliable broadcast channel for present voters is established (e.g. via Byzantine agreement [21]). For the sake of simplicity, we describe the tallying with the anonymization performed via mix net shuffle. However, the scheme can be easily modified for supporting homomorphic tallying.

In further descriptions we imply that every message is signed by its sender id_i with a private signing key sk_{id_i}. In order to prevent the reuse of old signatures, the signature should furthermore incorporate timestamps and/or other specific information about the election.

4.1 Pre-Election

A list of all the eligible voters $id_1, ..., id_N$ is made available [8], with a list of their public signing keys pk_{id_i} (the corresponding private signing keys sk_{id_i} are possessed only by the voters). Furthermore, each voter broadcasts a pair of keys (g_i, h_i) with $x_i = \log_{g_i} h_i$ known only to the voter id_i. The list of voters that are about to be present at the meeting is known in advance, so that the majority of them are actually present.

4.2 Delegation

The delegation can occur before as well as during the election, prior to the voting. We define $V_d \subset \{id_1, ..., id_N\}$ as a set of voters who delegate, and $V_p =$

---

[7] Note that as this PKI can used independently of any specific election, it can be prepared well in advance and reused subsequently.

[8] This list, for example, could be a list of board members who have a right to participate in the meeting.

$\{id_1, ..., id_N\} \backslash V_d$ as the voters who decide to vote directly (referred to as present voters, or as proxies).

The threshold t is defined as $\lfloor N_p/2 \rfloor + 1$, with $N_p = |V_p|$ as the number of present voters. If a voter $id_i \in V_d$ decides to delegate, following steps are required:

The voter id_i selects a random value $m_i \in \mathbb{Z}_q$, which serves as her delegation token. She then shares $g_i^{m_i}$ among present voters as follows:

- Compute the shares of m using Shamirs secret share scheme: select a random polynomial $f_i(x) \in \mathbb{Z}_q[x]$ with degree $t - 1$ and $f_i(0) = m_i$. For each voter $id_j \in V_p$, compute secret share $m_{i,j} = f_i(j)$.
- For each voter $id_j \in V_p$, furthermore compute commitments $c_{i,j} = (c_{i,j}^{(1)}, c_{i,j}^{(2)})$ with $c_{i,j}^{(1)} = g_i^{r_{i,j}} h_i^{u_{i,j}}$, $c_{i,j}^{(2)} = g_i^{m_{i,j}} h_i^{r_{i,j}}$ for random $r_{i,j}, u_{i,j} \in \mathbb{Z}_q$, and a digital signature on $c_{i,j}$, $s_{i,j} = \mathsf{Sign}(\mathsf{sk}_{id_i}, c_{i,j})$.
- For each voter $id_j \in V_p$, set $m'_{i,j}$ to m_i if the voter id_j is chosen as a proxy, and a random value in $\mathbb{Z}_q$ otherwise. If the voter does not want to choose a proxy and wants to abstain instead, she sets $m'_{i,j}$ to a random value in $\mathbb{Z}_q$ for each voter.

The tuple $(g_i^{m_{i,j}}, m'_{i,j}, s_{i,j}, r_{i,j}, u_{i,j})$ is being sent to each voter $id_j \in V_p$ over a private channel. Note that id_j can compute $c_{i,j}^{(1)}, c_{i,j}^{(2)}$ herself.

4.3 Setup

At this point, any voter id_i who delegated her voting right can change her mind and attend the meeting; in that case, id_i is excluded from V_d and added to V_p prior to voting.

During the election, the distributed threshold secret sharing is being executed by the present voters $id_j \in V_p$ to establish the public election key $\mathsf{pk}_v = (g_v, h_v)$ and the corresponding private election key sk_v with $h_v = g_v^{\mathsf{sk}_v}$. At this point the list of valid voting options is being made available, as $\mathbb{V} = \{v_1, ..., v_L\} \subset \mathbb{Z}_q^L$.

Furthermore, for all the delegating voters $id_i \in V_d$ an encryption of the delegation token m_i with pk_v is jointly calculated, whereby each voter $id_j \in V_p$ performs the following steps, given the tuple $(g_i^{m_{i,j}}, m'_{i,j}, c_{i,j}, r_{i,j}, u_{i,j})$ as received during the delegation:

- Encrypt her share of $g_i^{m_i}$ resulting in $e_{i,j}^{(d)} = \mathsf{Enc}(\mathsf{pk}_v, g_i^{m_{i,j}})$,
- Compute the proof of knowledge $\chi_{i,j}$, which is constructed using the technique in [7] and proves that $e_{i,j}^{(d)}$ encrypts the same value that is committed in $c_{i,j} = (c_{i,j}^{(1)}, c_{i,j}^{(2)})$ (i.e. $\chi_{i,j} = PoK\{r_{i,j}, u_{i,j}, r'_{i,j} : a_{i,j} = g_v^{r'_{i,j}} \wedge b_{i,j}/c_{i,j}^{(2)} = h_v^{r'_{i,j}} h_i^{-r_{i,j}} \wedge c_{i,j}^{(1)} = g_i^{r_{i,j}} h_i^{u_{i,j}}\}$ for $e_{i,j}^{(d)} = (a_{i,j}, b_{i,j})$).
- Broadcast the tuple $(id_i, e_{i,j}^{(d)}, c_{i,j}, \chi_{i,j})$.

Given that for each i, at least t of the values of $e_{i,j}^{(d)}$ with valid proofs, $j \in Q_i \subset \{1, ..., N\}$, $|Q_i| \geq t$ are broadcast, these values are combined as

$$e_i^{(d)} = \prod_{j \in Q_i} (e_{i,j}^{(d)})^{\lambda_{i,j}}$$

with $\lambda_{i,j} := \sum_{k \in Q_i, k \neq j} \frac{j}{j-k}$. The resulting value of $e_i^{(d)}$ thus corresponds to the encryption of $g_i^{m_i} = g_i^{\sum_{j \in Q_i} m_{i,j} \lambda_{i,j}}$ with the public signing key pk_v.

4.4 Voting

The voters who are present in the meeting (i. e. $id_j \in V_p$) cast their ballots directly by submitting $E_j^{(p)} = \mathsf{Enc}(\mathsf{pk}_v, v_j)$ with v_j signifying their choice, and the accompanying well-formedness proof σ_j that proves the knowledge of v_j and, in case of anonymization via homomorphic tallying, that $v_j \in \mathbb{V}$. Furthermore, for each delegating voter $id_i \in V_d$, each present voter $id_j \in V_p$ calculates a value $\hat{e}_{i,j}^{(d)} = \mathsf{Enc}(\mathsf{pk}_v, g_i^{m'_{i,j}})$. Note that $\hat{e}_{i,j}^{(d)}$ encrypts m_i only in case that the voter id_j is in possession of a token m_i (i.e. $m'_{i,j} = m_i$). For the sake of ensuring soundness, the voter further calculates $\pi_{i,j}$ as a proof of knowledge of plaintext discrete logarithm for $m'_{i,j}$ constructed using the technique described in [7] (i.e. $\pi_{i,j} = PoK\{w_{i,j}, m_{i,j} : a_{i,j}^{(d)} = g_v^{w_{i,j}}, b_{i,j}^{(d)} = g_i^{m_{i,j}} h_v^{w_{i,j}}\}$ with $\hat{e}_{i,j}^{(d)} = (a_{i,j}^{(d)}, b_{i,j}^{(d)})$), calculates $E_{i,j}^{(d)}$ as $\mathsf{Enc}(\mathsf{pk}_v, v_{i,j}^{(d)})$ with $v_{i,j}^{(d)}$ as her chosen option to cast on behalf of the delegating voter id_i, $\sigma_{i,j}$ as the proof of plaintext knowledge for $E_{i,j}^{(d)}$, and broadcasts the tuple $(\hat{e}_{i,j}^{(d)}, E_{i,j}^{(d)}, \pi_{i,j}, \sigma_{i,j})$.

4.5 Tallying - Weeding Duplicates and Invalid Delegations

In the next stage, the delegated ballots are jointly processed by the present voters. First, the delegated ballots with invalid proofs of knowledge $\pi_{i,j}, \sigma_{i,j}$ are removed. Then, the vote updating policy is applied. Namely, the given two ballots cast as direct ballots by the same voter, or two delegated ballots cast on behalf of the same voter by the same proxy, either all but the last (if vote updating is allowed) or all by the first (if vote updating is not allowed) cast ballot are excluded from further processing.

The next step removes the delegated ballots if they have canceled by the voter, i.e. if the voter cast a direct ballot instead. Namely, out of all the delegated ballots tuples $(\hat{e}_{i,j}^{(d)}, E_{i,j}^{(d)}, \pi_{i,j})$, the ballots with $id_i \in V_p$ are removed.

The remaining delegated ballots are being anonymized via verifiable re-encryption mix net with each present voter acting as a mix node, resulting in an anonymized list $V = \{(\hat{e'}_{i,j}^{(d)}, E'_{i,j}^{(d)})\}_{id_i \in V_d, id_j \in V_p}$. The values $e_i^{(d)}$ that encrypt the voters delegation tokens m_i are also processed through the mix net resulting in an anonymized list $V' = \{e'_i^{(d)}\}_{id_i \in V_d}$. The next step removes the delegated ballots cast with an invalid delegation token. For this, the following procedure is performed for each anonymized tuple $(\hat{e'}_{i,j}^{(d)}, E'_{i,j}^{(d)}) \in V$:

- Calculate $\mathsf{PET}(\hat{e'}_{i,j}^{(d)}, e'_i^{(d)})$ for each $e'_i^{(d)} \in V'$.
- If the PET is positive for some $e'_i^{(d)}$, add $E'_{i,j}^{(d)}$ to the list V'' for further tallying and remove $e'_i^{(d)}$ from V'.

4.6 Tallying - Mixing and Decrypting

After that, the list of ciphertexts $\{E_j^{(p)}\}_{id_j \in V_p} \cup \{E'_i^{(d)}\} \in V''$ is being anonymized with another mix net shuffle. The anonymized result is being decrypted via distributed decryption.

5 Security

We now conduct an informal security evaluation of the proposed scheme. Namely, we argue that the security requirements outlined in Section 2 are fulfilled under the following assumptions[9]:

(A1) Out of N_p present voters, at least $N_p - t + 1$ are honest and do not divulge their private information to the adversary.
(A2) The devices of honest voters are trustworthy.
(A3) At least t of present voters are available, capable to communicate with each other, and produce valid output during the election.
(A4) The PKI is trustworthy.
(A5) The adversary is computationally restricted, the decisional Diffie-Hellman problem is hard in the selected group, and the signature scheme used in the PKI is reliable. The random oracle is instantiated by the hash function $\mathcal{H}$.
(A6) No coercion takes place.

We start off with evaluating the security requirement related to direct voting.

Voter eligibility. This requirement is ensured as long as the PKI used to authenticate the voters is trustworthy (A4). Furthermore, the duplicate votes submitted by voters are removed throughout the tallying phase. Unless the PKI is not trustworthy (A4), dishonest voters cannot cast multiple votes. If a voter delegates her right to vote and additionally casts a vote personally, then the voter's delegation is invalidated throughout the tallying phase if the PKI is trustworthy (A4).

Vote integrity. For direct votes, this requirement can be violated by replacing a cast vote with another ciphertext at the time of vote casting. Alternatively, the adversary could drop the messages with cast votes from particular voters at vote casting, thus excluding these votes from the tally. However, given a trustworthy voting device, such a manipulation will be detected by the voter, since her result would not fit with the result of other voters (A2). Another way to manipulate

---

[9] Note, that these assumptions are common within e-voting systems, e.g. Helios [1].

the tally would be to replace the ballots during the shuffling or to produce an incorrect decryption result. Both possibilities are prevented by the soundness of the zero knowledge proofs of shuffle validity and decryption validity (A5).

We now consider the integrity of delegated votes. Note, in case the voter has delegated her vote to multiple proxies, only a vote from one of them is included into the tallying. Hence, in this way excluding the votes of other proxies from being included in the tallying is not considered a violation of delegation integrity for proxies. Similarly, excluding the votes cast on behalf of dishonest voters does not violate the requirement.

A dishonest majority of present voters might prevent the delegated ballot on behalf of the particular voter from being included in the tally by refusing to publish their values $e_{i,j}^{(d)}$ and preventing the reconstruction of $e_i^{(d)}$. While this is prevented by the assumption that more than half of all the present voters are honest (A5), we still do not consider it to be a violation of vote integrity, since the misbehaviour of dishonest voters would be detected.

On the other hand, publishing the invalid values $e_{i,j}^{(d)}$, so that the reconstructed $e_i^{(d)}$ does not encrypt the value of $g_i^{m_i}$ for a valid delegation token m_i, would indeed be a violation of vote integrity, if undetected. However, the soundness of zero-knowledge proof $\chi_{i,j}$ that accompanies $\hat{e}_{i,j}^{(d)}$ and the computational binding property of the commitment $c_{i,j}$ (A5) that holds unless the secret x_i is leaked (A2) ensure, that each $e_{i,j}^{(d)}$ encrypts the value $g^{m_{i,j}}$ contained in $c_{i,j}$. Since $c_{i,j}$ is signed by the voter (and a lack of a valid signature would be noticeable to the honest present voters, as well as to the delegating voters who verify the election data), the unforgeability of the signature (A5) ensures that $c_{i,j}$ was sent by the voter herself, hence, it contains the valid value of $g_i^{m_{i,j}}$. Hence, the reconstructed value $e_i^{(d)}$ encrypts the same $g_i^{m_i}$ that is shared by the voter id_i.

Another way to prevent the delegated votes from an honest proxy to be included in the tally is to ensure that the result of $\mathsf{PET}(\hat{e'}_{i,j}^{(d)}, e'_i^{(d)})$ outputs some value other than 1. This is prevented due to the soundness of the zero-knowledge proofs accompanying the PETs. Furthermore, analogously to the case of direct ballots, the soundness of zero knowledge proofs regarding shuffle validity and decryption validity prevent the manipulation of cast ballots (A5).

Robustness. Violating robustness would mean, that either the mixing, the weeding of invalid delegations or the decryption has failed to output a valid output. This is prevented if at least t present voters are available and provide the required output during the tallying (A3).

Vote privacy. This requirement is violated if the adversary corrupts voting devices, which then leak the choices made by the voters. This is prevented as long as the honest voters' devices are not compromised (A2). Furthermore, the voters themselves do not leak the randomness used by encrypting the vote (A2).

Another way to violate vote privacy of honest voters is to decrypt the encrypted votes prior to their anonymization (i.e. before mixing). This, however,

requires breaking the encryption of the votes (A5), or obtaining at least t shares of a private election key from the present voters (A1,A2).

Furthermore, vote privacy can be violated by revealing the secret permutation used by each voter during the mixing. However, as long as at least one voter keeps this permutation secret (A1), the permutation between the resulting output and the input ciphertexts remains secret as well.

Fairness. As the cast ballots are attached to the voter's identities until the tallying, violating fairness would also imply violating vote privacy. Hence, fairness is ensured under the same assumptions as vote privacy: namely, that the voting devices of honest voters are trustworthy (A2), at least $N_p - t + 1$ of N_p present voters are honest (A1), and the underlying encryption cannot be broken (A5).

We further evaluate the security requirements related to delegated voting.

Delegation eligibility. Casting a delegated ballot on behalf of a non-eligible voter would require forging the signatures on the commitments $c_{i,j}$ sent to the present voters (prevented by the assumptions (A4) and (A5)). Furthermore, multiple delegated ballots on behalf of the same voter are dismissed during tallying.

Delegation integrity. One way to violate this requirement would be to cast delegated votes on behalf of non-eligible voters. Given the fact that delegations are accompanied by signed values $c_{i,j}$, this attack strategy is prevented unless the underlying PKI is not trustworthy (A4). Furthermore, reusing old signatures on $c_{i,j}$ would be prevented, since the election information and the timestamp are incorporated in the signature.

Another way to violate this requirement for a proxy id_j who wants to vote on behalf of the voter id_i without being authorised, is to find out the value of m_i, shared by id_i to the present voters during the delegation. This would require either corrupting the voting device of id_i (A2) or eavesdropping on the communication between id_i and a proxy chosen by her (prevented due to private communication channels, i.e. the trustworthiness of the PKI (A4)). Note that even if the adversary succeeds in obtaining at least t shares of $g_i^{m_{i,j}}$ from the present voters, she would still require to compute the discrete logarithm $m_{i,j}$ (A5).

Alternatively, an adversary can attempt manipulating the computation of $e_i^{(d)}$, so that it encrypts a plaintext $g^{m_i'}$ chosen by her. As shown in the evaluation of vote integrity, however, the assumptions (A4, A5) ensure than $e_i^{(d)}$ encrypts the same value $g_i^{m_{i,j}}$ sent by the voter.

Finally, delegation integrity can be violated, if the proxy id_j submits a value $\hat{e}_{i,j}^{(d)}$ which is accepted during the weeding of invalid delegations. The soundness of the proof of knowledge of plaintext discrete logarithm $\pi_{i,j}$ ensures (A5), that the proxy knows the discrete log m_i of the plaintext $g_i^{m_i}$ encrypted in $\hat{e}_{i,j}^{(d)}$. As shown above, the assumptions (A4) and (A5) ensure that the reconstructed values $e_i^{(d)}$ encrypt the delegation tokens submitted by the voters to their chosen

proxies. The soundness of the proof of shuffle ensures (A5), that the anonymized encrypted delegation tokens $e'^{(d)}_i$ encrypt the same values as $e^{(d)}_i$.

Delegation privacy. The delegation privacy requirement would be violated if it is revealed which proxy possesses the value m_i that was shared by the voter id_i among other present voters. This can be achieved either by corrupting the voting device of id_i that stores m_i (A2), coercing the present voters into disclosing all the shares $m_{i,j}$ with the adversary (A6), getting access to at least t shares of $g_i^{m_i}$ (i.e. corrupting at least t present voters (A1), their voting devices (A2) or the communication channels between the present and the delegated voters (A4)), or decrypting $e^{(d)}_i$ and the values of $\hat{e}^{(d)}_{i,j}$ (i.e. either breaking encryption (A5) or obtaining at least t shares of a secret key sk_v by corrupting at least t present voters (A1) or their voting devices (A2)).

Furthermore, the delegating voter herself cannot construct a proof that she delegated to a specific proxy, even if she provides all the shares $g_i^{m_{i,j}}$ and the value of m_i to the adversary. Namely, given that the voter knows the discrete logarithm $x_i = \log_{g_i} h_i$, she can provide fake values of $g_i^{m_{i,j}}$, m_i instead. As such, for every values $m_{i,j}, r_{i,j}$ and $u_{i,j}$ (thus, for every pair of commitments $c^{(1)}_{i,j}, c^{(2)}_{i,j}$) and every value $m'_{i,j} \neq m_{i,j}$ the voter can find $r'_{i,j}, u'_{i,j}$ so that $x_i r_{i,j} + m_{i,j} = x_i r'_{i,j} + m'_{i,j}$ and $x_i u_{i,j} + r_{i,j} = x_i u'_{i,j} + r'_{i,j}$ (thus, $c^{(1)}_{i,j} = g_i^{r'_{i,j}} h_i^{u'_{i,j}}$ and $c^{(2)}_{i,j} = g_i^{m'_{i,j}} h_i^{r'_{i,j}}$). She can then fake the receipt by sending a random value m'_i and a set of shares $m'_{i,j}$ that reconstruct to m'_i together with the corresponding values of $r'_{i,j}, u'_{i,j}$ to the present voter who requests such a receipt. Given t as threshold and N_p as the total amount of present voters among which g^{m_i} is shared, the voter would have to fake at least $N_p - t + 1$ shares $m_{i,j}$. Hence, as long as at least $N_p - t + 1$ present voters are honest, and that the delegating voter knows the identities of the honest present voters, the adversary would not be able to distinguish between the fake values $g_i^{m_{i,j}}$, m_i that from the real ones.

Note, however, that in case one of the voters $id_j \in V_p$ (i.e. who received delegations) is not available in the meeting, our scheme reveals the number of delegating voters who either abstained (but still participated in the delegation by issuing invalid delegation tokens $m'_{i,j} \neq m_i$ to all the voters in V_p) or delegated to id_j. We do not consider such a case to be a violation of delegation privacy, since, as shown above, the scheme does not reveals the identities of the voters who either issued invalid delegation tokens or delegated to id_j and does not make it possible to tell whether a given voter issued a valid token to id_j or not (under the assumptions (A1, (A2), (A4), (A5), and (A6)). At the same time, in order to reduce the information leakage in our scheme, we would suggest actively encouraging that the voters in V_d who decide to abstain still participate in the delegation phase of the election by issuing invalid delegation tokens $m'_{i,j} \neq m_i$ to all the voters in V_p. Furthermore, the voters in V_p can be encouraged to re-delegate by forwarding their delegation token to another trusted present voter, if they think they would not be able to participate in the meeting.

Delegation power privacy. Given $N_d = N - N_p$ delegating voters, each present voter should posses N_d delegation tokens. Violating delegation power privacy would mean estimating, possibly with the help of the proxy herself who tries to prove her delegation power, how many of those tokens are valid. However, given that the delegation privacy requirement is fulfilled, a proxy herself does not know which ones of the delegation tokens she received are valid. Hence, under the assumptions that at least $N_p - t + 1$ of the present voters are honest (A1), the PKI is trustworthy (A4), the voting devices of the delegating voters and honest proxies are trustworthy (A2), the voters do not collaborate with the proxy to prove that they delegated their voting right to her (A6) and the encryption is not broken (A5), delegation power privacy is ensured.

Note that as already mentioned in the evaluation of delegation privacy, if a proxy $id_j \in V_p$ does not participate in the election, our scheme could reveal the number of voters N_j who either delegated to id_j or issued invalid delegation tokens $m_{i,j}$ to all the proxies. However, since the scheme does not reveal, how many voters out of N_j abstained, delegation power privacy is not violated, especially if the voters who want to abstain are encouraged to issue invalid delegation tokens instead of not participating at all.

6 Related Work

A number of proposals considered decentralised elections, i.e. the boardroom voting setting. The first proposal was made in [9] using decryption mix net. Several proposals focused on self-tallying approach, based upon self-dissolving commitments [12–14, 16, 17, 27]. Other approaches to boardroom voting have extended the decentralised tallying scheme proposed in [8] and partially implemented in the Helios voting system [1]. The variant of this approach using homomorphic tallying has been proposed in [25], and the variant using mix net and decentralised PKI establishment has been proposed and implemented in [20]. A boardroom voting system described and evaluated in [2] implements a boardroom voting scheme that does not rely on cryptography.

Several schemes with delegated voting functionality have been proposed in the literature. The proposal by Kulyk *et al.* [19] addresses coercion resistance in delegated voting by extending the well-known coercion-resistant JCJ/Civitas theme for electronic voting towards delegated voting. A further proposal in [18] extends the Helios voting system with delegated voting functionality. Their approach, however, only allows to delegate after the election setup has been conducted (i.e. after the election key has been generated) which is not suitable to the boardroom voting setting that we consider in this work. Tchorbadjiiski [28] introduces hash chains to compute "connected" credentials on the client side to enable a transitive and revocable delegation process, and the proposal by Zwattendorfer *et al.* [30] uses blind signatures to enable delegation privacy.

7   Conclusion

We proposed an electronic voting scheme that facilitates delegated voting in the boardroom voting setting. The scheme enforces both general security requirements on electronic voting, such as vote privacy and integrity of the election, and security requirements that are specifically tailored to the delegation process. As such, the scheme ensures delegation privacy by hiding the link between the voter and the identity of her chosen proxy, delegation integrity by ensuring that a proxy can only cast a ballot on some voter's behalf if authorised by the voter, delegation eligibility to ensure that only the delegated ballots on behalf of the eligible voters are included in the tally and delegation power privacy to ensure, that the scheme does not reveal how many voters have delegated to a particular proxy.

In the future, we plan to extend our security evaluation and provide formal security proofs for the proposed delegated boardroom voting scheme. For this, the formal definitions for security requirements specific to delegated voting should be established.

Another direction of future work would be to address the attack vectors on vote secrecy, delegation privacy and delegation power privacy that exploit the information revealed by the election result. Due to the relatively small number of voters in boardroom voting, these attacks might have more impact than they have in large scale elections. Hence, we plan to consider ways to minimize information revealed by the result during the course of the delegation.

Finally, a direction of future work would be focusing on the efficiency of the scheme. While the scheme is designed for small-scale elections , it nonetheless requires a relatively high level of interaction among the present voters (the bottleneck being the weeding of invalid delegated ballots during the tallying). Therefore, methods for improving the efficiency by reducing the required communications and computations are required.

References

1. Adida, B.: Helios: Web-based open-audit voting. In: SS 2008: 17th Conference on Security Symposium. pp. 335–348. USENIX (Jul 2008)

2. Arnaud, M., Cortier, V., Wiedling, C.: Analysis of an electronic boardroom voting system. In: VoteID 2013: 4th International Conference on E-Voting and Identify. pp. 109–126. Springer (Jul 2013)
3. Bayer, S., Groth, J.: Efficient zero-knowledge argument for correctness of a shuffle. In: EUROCRYPT 2012: Advances in Cryptology. pp. 263–280. Springer (2012)
4. Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to Helios. In: ASIACRYPT 2012: 18th International Conference on the Theory and Application of Cryptology and Information Security. pp. 626–643. Springer (Dec 2012)
5. Budurushi, J., Neumann, S., Olembo, M.M., Volkamer, M.: Pretty understandable democracy-a secure and understandable internet voting scheme. In: ARES 2013: 8th International Conference on Availability, Reliability and Security. pp. 198–207. IEEE (2013)
6. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups. In: CRYPTO 1997: Advances in Cryptology, pp. 410–424. Springer (1997)
7. Camenisch, J., Stadler, M.: Proof systems for general statements about discrete logarithms. Tech. rep., Citeseer (1997)
8. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. European transactions on Telecommunications 8(5), 481–490 (1997)
9. DeMillo, R.A., Lynch, N.A., Merritt, M.J.: Cryptographic protocols. In: STOC 1982: 14th annual ACM symposium on Theory of computing. pp. 383–400. ACM (1982)
10. Diffie, W., Hellman, M.: New directions in cryptography. IEEE transactions on Information Theory 22(6), 644–654 (1976)
11. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: CRYPTO 1984: Workshop on the Theory and Application of Cryptographic Techniques. pp. 10–18. Springer (1984)
12. Giustolisi, R., Iovino, V., Rønne, P.: On the possibility of non-interactive e-voting in the public-key setting. In: FC 2016: International Workshops, BITCOIN, VOTING, and WAHC. Springer (Feb 2016)
13. Groth, J.: Efficient maximal privacy in boardroom voting and anonymous broadcast. In: FC 2004: 8th International Conference on Financial Cryptography. pp. 90–104. Springer (Feb 2004)
14. Hao, F., Ryan, P.Y., Zielinski, P.: Anonymous voting by two-round public discussion. IET Information Security 4(2), 62–67 (2010)
15. Jakobsson, M., Juels, A.: Mix and match: Secure function evaluation via ciphertexts. In: Okamoto, T. (ed.) ASIACRYPT 2000: 6th International Conference on the Theory and Application of Cryptology and Information Security. pp. 162–177. Springer (Dec 2000)
16. Khader, D., Smyth, B., Ryan, P.Y., Hao, F.: A fair and robust voting system by broadcast. In: EVOTE 2012: 5th International Conference on Electronic Voting. vol. 205, pp. 285–299. Gesellschaft für Informatik (2012)
17. Kiayias, A., Yung, M.: Self-tallying elections and perfect ballot secrecy. In: PKC 2002: International Workshop on Public Key Cryptography. pp. 141–158. Springer (2002)
18. Kulyk, O., Marky, K., Neumann, S., Volkamer, M.: Introducing proxy voting to Helios. In: ARES 2016: 11th International Conference on Availability, Reliability and Security. pp. 98–106. IEEE (Sep 2016)

19. Kulyk, O., Neumann, S., Marky, K., Budurushi, J., Volkamer, M.: Coercion-resistant proxy voting. In: IFIP SEC 2016: 31st International Conference on ICT Systems Security and Privacy Protection. pp. 3–16. Springer (Jun 2016)
20. Kulyk, O., Neumann, S., Volkamer, M., Feier, C., Koster, T.: Electronic voting with fully distributed trust and maximized flexibility regarding ballot design. In: EVOTE 2014: 6th International Conference on Electronic Voting, Verifying the Vote. pp. 1–10. IEEE (2014)
21. Lamport, L., Shostak, R., Pease, M.: The Byzantine generals problem. TOPLAS 1982: ACM Transactions on Programming Languages and Systems 4(3), 382–401 (1982)
22. Nguyen, L.H., Roscoe, A.W.: Efficient group authentication protocol based on human interaction. In: FCS-ARSPA 2006: Workshop on Foundation of Computer Security and Automated Reasoning Protocol Security Analysis. pp. 9–33 (Aug 2006)
23. Pedersen, T.P.: Distributed provers and verifiable secret sharing based on the discrete logarithm problem. DAIMI Report Series 21(388) (1992)
24. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: CRYPTO 1992: Advances in Cryptology. pp. 129–140. Springer (1992)
25. Ritter, J.: Decentralized e-voting on android devices using homomorphic tallying. Master's thesis, Bern University of Applied Sciences, Biel, Switzerland (2014)
26. Shamir, A.: How to share a secret. Communications of the ACM 22(11), 612–613 (1979)
27. Szepieniec, A., Preneel, B.: New techniques for electronic voting. JETS 2015: USENIX Journal of Election Technology and Systems pp. 46–69 (2015)
28. Tchorbadjiiski, A.: Liquid democracy diploma thesis. RWTH AACHEN University, Germany (2012)
29. Terelius, B., Wikström, D.: Proofs of restricted shuffles. In: AFRICACRYPT 2010: 3rd International Conference on Cryptology in Africa. pp. 100–113. Springer (May 2010)
30. Zwattendorfer, B., Hillebold, C., Teufl, P.: Secure and privacy-preserving proxy voting system. In: ICEBE 2013: IEEE 10th International Conference on e-Business Engineering. pp. 472–477. IEEE (Sep 2013)