HAL open science

Coercion-Resistant Proxy Voting

Oksana Kulyk, Stephan Neumann, Karola Marky, Jurlind Budurushi, Melanie Volkamer

To cite this version:

Oksana Kulyk, Stephan Neumann, Karola Marky, Jurlind Budurushi, Melanie Volkamer. Coercion-Resistant Proxy Voting. 31st IFIP International Information Security and Privacy Conference (SEC), May 2016, Ghent, Belgium. pp.3-16, 10.1007/978-3-319-33630-5_1 . hal-01369533

HAL Id: hal-01369533

https://inria.hal.science/hal-01369533v1

Submitted on 21 Sep 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Coercion-resistant Proxy Voting

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

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

Abstract. In general, most elections follow the principle of equality, or as it came to be known, the principle of "one man – one vote". However, this principle might pose difficulties for voters, who are not well informed regarding the particular matter that is voted on. In order to address this issue, a new form of voting has been proposed, namely *proxy voting*. In proxy voting, each voter has the possibility to delegate her voting right to another voter, so called *proxy*, that she considers a trusted expert on the matter. In this paper we propose an end-to-end verifiable Internet voting scheme, which to the best of our knowledge is the first scheme to address voter coercion in the proxy voting setting.

1 Introduction

Democratic elections represent an important value in many modern states. This is not limited to governments, also companies and other organizations are constituted in democratic elections. In general, most democratic elections follow the principle of equal elections, meaning that one person's vote should be worth as much as another's, i.e. one man – one vote [14]. However, this principle often represents a challenge to voters, who are not sufficiently informed regarding the particular matter that is voted on. In order to address this issue, a new form of voting has been proposed, the so called *proxy voting*. In proxy voting, each eligible voter has the possibility to delegate her voting right to another eligible voter, so called *proxy*, that she considers a trusted expert on the matter.

There already exist few proxy voting implementations, provided by different organizations. Two widely known implementations are LiquidFeedback[3] and Adhocracy[4]. Further proxy voting proposals are the approaches proposed in [19] and [21].

However, all existing proxy voting proposals fail to address the issue of voter coercion: namely, the case when the adversary threatens the voter to vote in a particular way, or to abstain from voting. This issue has been commonly considered for non-proxy Internet voting, and a number of Internet voting schemes have been proposed, that address the problem of coercion, e.g. by providing coercion resistance [12] or coercion evidence [8]. In this paper, we build upon [6,12]

[3] http://liquidfeedback.org/, last accessed January, 7, 2016.
[4] https://adhocracy.de/, last accessed January, 7, 2016.

and an extension proposed by Spycher *et al.* [18] to propose a coercion resistant end-to-end verifiable Internet proxy voting scheme.

This paper is structured as follows: In section 2 we identify and derive security requirements that are relevant for proxy voting. Section 3 introduces the fundamentals used for our proposal, which we present in section 4. In section 5 we evaluate the security of our proposal with respect to the requirements. Section 6 summarizes our contributions and provides directions for future research.

2 Requirements for Proxy Voting

The following **functional requirements** should be provided by a proxy voting system:

Delegation cancellation. If the voter for any reasons decides to vote herself, she should be able to cancel the delegation at any point of time before the tallying.

Delegation back-up. The voter can assign up to T priorities to her proxies. Only the vote from the proxy having highest priority will be included in the vote count. This functionality is useful if the voter wants to have a "back-up" delegation, in case her first choice of a proxy abstains from the election.

The **security requirements** in Internet voting have been thoroughly investigated in the literature, and both formal [7] and informal [13,16] definitions have been proposed. In this work, we aim to address the following security requirements for the election.

Vote integrity. All votes cast by eligible voters should be included in the result.

Availability. It should be possible to compute election result even if some of the involved entities are faulty.

Vote secrecy for voters. The adversary should not be able to learn how a voter has voted.

We aim at achieving the following security requirements for the delegation process:

Delegation integrity. Only the proxy having a valid permit from the voter should be able to cast a vote on this voter's behalf. The proxy should not be able to alter the priority given to them by the voter.

Delegation availability. A proxy should not be selectively prevented from having the votes delegated to her.

Vote secrecy for proxies. The adversary should not be able to learn how a proxy has voted.

Delegation privacy. There should not be a link between a voter's identity and her selected proxy. Furthermore, it should not be possible to reveal whether a voter has delegated a vote or cast it herself.

Delegation unprovability. The proxy should not be able to prove to anyone how many votes have been delegated to her. Moreover, the proxy can not gain any knowledge about the actual number the incoming delegations.

Note, that as we want to ensure coercion resistance, we require that vote secrecy for both voters and proxies should be ensured also for the case when the adversary is capable of communicating with the voter or proxy.

3 Background

In this section we introduce the fundamentals used to design our coercion-resistant verifiable proxy voting scheme.

3.1 Cryptographic Primitives

In the following we describe the cryptographic primitives our scheme relies on. Hereby, $\mathbb{G}_q$ denotes a cyclic multiplicative group with order q and $\mathbb{Z}_q$ denotes the cyclic additive group of integers modulo q.

Zero-knowledge proofs. In order to prove the correctness of statements within the voting scheme without revealing anything beyond the correctness *zero-knowledge proofs* are employed. For this sake, techniques such as proving the knowledge of discrete logarithm [17] or discrete logarithm equality [5] are being used. These proofs can be made non-interactive by employing the strong version of the Fiat-Shamir heuristic suggested in [3]. An important extension of such proofs are *designated-verifier proofs* described in [11]. Given the verifier's public key, these proofs convince only the designated verifier about the correctness, rather than the general public.

Linear Encryption. In some parts of our scheme, we use a modified encryption scheme suggested in [4] (further denoted as *LE-ElGamal*). This scheme is semantically secure under the DLIN assumption which is implied in groups where the DDH assumption holds. Namely, let $pk = (g_1, g_2, h) \in \mathbb{G}_q^3$ be the public keys of the encryption and $(x_1, x_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$ the private keys with $g_1^{x_1} = g_2^{x_2} = h$. If the keys are jointly generated by multiple parties with x_1, x_2 as threshold-shared secrets, then, according to [2] at least 2/3 of the parties have to be honest.

The message $m \in \mathbb{G}_q$ is encrypted as follows: two random values $(r_1, r_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$ are chosen and then the encryption - denoted as $\{\{m\}\}_{pk} \in \mathbb{G}_q^3$ - is calculated as $(c_1, c_2, c_3) = (g_1^{r_1}, g_2^{r_2}, m \cdot h^{r_1+r_2})$. The decryption then proceeds as $m = c_3 \cdot c_1^{-x_1} \cdot c_2^{-x_2}$. Ciphertexts can then be reencrypted by multiplying a ciphertext by an encryption of 1 using a random value $(r_1', r_2') \in \mathbb{Z}_q \times \mathbb{Z}_q$.

Further operations used together with the ElGamal encryption, such as mix net shuffle, well-formedness proofs and plaintext equivalence tests can be adjusted for LE-ElGamal as well.

Plaintext Equivalence Tests. In order to check whether a given ciphertext c encrypts the same plaintext as another ciphertext c' without revealing any additional information about plaintexts, *plaintext-equivalence tests* [10] can be employed. This can be performed by the holders of an encryption secret key and consists of jointly decrypting the value $(c/c')^r$ given a secretly shared random value $r \in \mathbb{Z}_q$ which results in 1 in case the plaintexts are equal or in random value otherwise.

Proxy Reencryption. Let $\{m\}_{pk_1}$ be a ciphertext encrypting message m with ElGamal public key $pk_1 = (g_1, h_1)$. Given the knowledge of corresponding secret key $x_1 = \log_{g_1} h_1$ which can also be a shared secret between several participants the method described in [9] allows for computing a new ciphertext $\{m\}_{pk_2}$, that encrypts the same message using a different ElGamal public key pk_2.

Mix nets. Important components in electronic voting systems are *mix net schemes* which are used for anonymizing lists of ciphertexts $e_1, ..., e_N$. In addition to ensure the integrity of the shuffle a number of zero-knowledge proofs have been proposed in the literature. The most efficient schemes - up to now - are presented in [20] and [1]. A modification of such proofs can be used to mix tuples of ciphertexts $(\bar{e_1}, ..., \bar{e_N})$ with $\bar{e_i} = (e_{i,1}, ..., e_{i,k})$ while preserving the ordering within the tuple.

3.2 JCJ/Civitas Scheme

For our goal to provide a scheme for coercion resistant proxy voting we chose the JCJ/Civitas voting scheme proposed in [12] and then extended and implemented in [6] as basis. The coercion resistance of the scheme is based on the application of voting credentials. These credentials are used to authorize votes from eligible voters. In case a coercer demands the credential from a voter she can simply provide a fake credential which could not be distinguished from the real one by the coercer. We briefly outline the scheme below.

Setup and Registration. Prior to the election the election supervisor announces the different election authorities, namely the *registrar*, *registration tellers* and *tabulation tellers* and publishes their respective public keys on the bulletin board. The registrar publishes the electoral register which contains the identity, the public registration key, and the public designation key of each voter. Building upon a homomorphic cryptosystem the tabulation tellers generate an election key pair in a distributed manner and publish the respective public key pk on the bulletin board.

For each voter V_i each registration teller $j = 1, ..., N$ generates a credential share $c_{i,j}$ and publishes its encryption next to the respective voter's identity on the bulletin board from which the encryption of the voting credential

$E_i = \{c_i\}_{pk} = \prod_{j=1}^{N} \{c_{i,j}\}_{pk}$ can be computed by multiplying all the individual credential shares. The shares $c_{i,j}$ in plaintext together with corresponding designated-verifier correctness proofs are then being forwarded to the voter. Now the voter can use them to compute their secret voting credential $c_i = \prod_{i=1}^{N} c_{i,j}$. Finally, the encrypted credentials E_i are being shuffled via mix net.

Voting. The voters use anonymous channels for vote casting. As her vote, the voter casts a tuple

$$\langle A_i = \{o\}_{pk}, C_i = \{c_i\}_{pk}, \sigma \rangle$$

with o as a chosen voting option and σ as well-formedness proof for A_i and proof of plaintext knowledge for C_i. The tuple is sent to one of the available *ballot boxes* which stores the votes. In case the voter is forced to reveal her credential to a coercer she can give a fake credential c' instead while the coercer is not able to distinguish it from a real one.

Tallying. The votes with invalid proofs are excluded and the plaintext-equivalence tests are used for identifying the votes with duplicated credentials which are handled according to the rules concerning vote updating. The remaining tuples $\langle A_i, C_i \rangle$ are being shuffled and of votes are being anonymized with mix net shuffling. Afterwards, plaintext-equivalence tests are applied for checking the validity of the voting credential by each C_i with each authorized credential from the shuffled list E_i. For the votes with valid credentials the voting options A_i are being decrypted.

Security assumptions that JCJ/Civitas relies on:

1. The adversary is not capable of simulating the voters during the registration process. The adversary is also not present the registration phase.
2. At least one registration teller is trusted and the communication channel between this teller and the voter is untappable[5].
3. The voting client is trusted.
4. At least k out of n tabulation tellers do not fail during decryption.
5. At least $n - k + 1$ out of n tabulation tellers are trusted not to reveal their key shares.
6. The channels between voters and voting system used for vote casting are anonymous.
7. The DDH assumption and the RSA assumption hold an a random oracle is implemented via cryptographic hash function.
8. At least one ballot box to which the cast votes are submitted is correct.

In addition to the security requirements, it is assumed that the voters are capable of handling the credentials, e.g. by using some kind of secure credential management.

[5] That is, the adversary is incapable of reading the messages sent over the channel.

4 Proposed Proxy Voting Scheme

To tailor our JCJ/Civitas extension towards proxy voting we introduce a new kind of credentials, so called *delegation credentials*. In addition to a unique voter credential in JCJ/Civitas, each voter i obtains a list of prioritized *delegation credentials*. To delegate a vote with a certain priority j the voter selects the j-th credential from her list and forwards it to the intended proxy. Voters are allowed to forward different credentials with different priorities to different proxies. Throughout the tallying phase for each voter only the vote cast with the highest priority is counted. Due to the fact that delegation credentials are generated on the same principles as the voting credentials in the original scheme the security of our extension also relies on the fact that the delegating credentials can be faked by the voter in case of coercion.

4.1 Necessary Modifications

We describe the modifications to the JCJ/Civitas scheme that are needed for implementing the delegation while ensuring the requirements listed in Section 2.

Ballot Clustering. Within the JCJ/Civitas scheme coercion-resistance is achieved by breaking the link between a voter's identity and votes cast in her name, both real and fake votes. The introduction of prioritized delegation credentials requires a relation between different credentials being maintained throughout the vote tallying phase. Retaining such a relation might however cause vulnerabilities with regard to coercion. To address these challenges we build upon proposals from scientific literature. Spycher *et al.* [18] present a JCJ extension towards linear tallying time. Therefore, the authors propose to assign identifiers to cast (real and fake) votes. During vote tallying after anonymization cast votes are only compared against the public credential assigned to their respective identifier. This reduces the tallying complexity from quadratic to linear regarding the number of cast votes. We build upon this approach: votes cast by the voter or delegated to different proxies share the same identifier such that within the set of votes sharing the same identifier the vote with the highest priority is tallied.

Delegation Server. Forwarding voting credentials to proxies results in a coercion vulnerability: The adversary might coerce a proxy to forward all received voting credentials. In order to test whether the proxy complies the adversary could anonymously delegate a credential to her and check whether this credential is being forwarded back to her. We address this problem by introducing a new entity - possibly implemented in a distributed way - that functions as *delegation server* (DS). The underlying idea is that proxies do not receive delegated credentials directly from the voter. Instead the voter blinds her credential and sends it to the DS (over an anonymous and untappable channel) which forwards an anonymization of the blinded credential to the proxy. The unblinding value is sent to the proxy over the private and anonymous channel.

Inclusion of Linear Encryption. To prevent unauthorized usage of voting credentials the JCJ/Civitas scheme forces the voter to include the plaintext knowledge proof for the ciphertext encrypting the credential. This solution, however, is inapplicable to the delegation process since the proxy does not get to know the value of the credential as outlined above. We address this challenge by publishing voting credentials (in the registration and voting phases) encrypted with linear encryption rather than ElGamal encryption. On the other hand for delegating her vote the voter encrypts their delegating credential with standard ElGamal using (g_2, h) as an encryption key. The resulting tuple $\{c\}_{pk} = (a = g_2^r, b = ch^r)$ together with other necessary information (see Section 3.2) is being forwarded to the proxy. If the proxy wants to cast the vote she chooses a random value of s and computes (g_1^s, a, bh^s) which is an LE-ElGamal encryption of c with randomness values r, s for which the proxy also can prove the knowledge of s as $\log_{g_1} g_1^s$.

4.2 Scheme Description

In this section we provide a detailed description of our proposed scheme.

Preliminary Stage. During this stage the keys used for encrypting votes and/or credentials are generated. The DS generates ElGamal keys with $pk_D = (g_D, h_D)$ as public key. The tabulation tellers distributively generate LE-ElGamal keys $pk_T = (g_1, g_2, h_T)$. We further denote $\{m\}_{pk_T}$ as ElGamal encryption with (g_2, h_T) as corresponding key and $\{\{m\}\}_{pk_T}$ as LE-ElGamal encryption.

The list of proxies $D_1, ..., D_d$ is being made public together with their public keys used for signing and designated-verifier proofs. For the sake of simplicity we assume that each proxy is eligible to vote herself as well. Furthermore, anonymous channels that enable communication between the proxies and the delegation servers as well as between proxies and the rest of the voters are established.

Setup Phase. Opposed to the standard JCJ/Civitas scheme, each registration teller generates T credential shares at random for each voter. Analogously to the JCJ/Civitas scheme, the encrypted credentials are publicly assigned to the respective voters whereby the order of the credential shares is of central importance to the delegation process. A public identifier, e.g. the position of the respective voter in the electoral roll is assigned to each voter. After the setup phase the bulletin board contains T credentials for every voter $V_1, ..., V_k$ (see Table 1) as well as individual credential shares from each of N registration tellers for each priority $1, ..., T$. We consider the lower number to denote the higher priority.

Registration. The registration phase remains identical to the standard JCJ/Civitas scheme except the fact that each registration teller releases T ordered credential shares to the voter. The voter can then verify whether the received shares from the tellers for a credential $c_i^{(j)}$ correspond to the encrypted shares published on the bulletin board near id_i and priority j.

ID	Priority	Credential shares
id_1	1	$\{\{c_1^{1,1}\}\}_{pk_T}, \ldots, \{\{c_1^{1,N}\}\}_{pk_T}$
$\vdots$	$\vdots$	$\vdots$
id_k	T	$\{\{c_n^{T,1}\}\}_{pk_T}, \ldots, \{\{c_n^{T,N}\}\}_{pk_T}$

Table 1. Content of the bulletin board after the setup phase of the extended scheme.

Before the voting, the voter merges her $N \cdot T$ credential shares as follows:

$$\begin{aligned} c_i^{(1)} &= c_i^{1,1} \cdot c_i^{1,2} \cdot \ldots \cdot c_i^{1,N} \\ \vdots \quad & \vdots \\ c_i^{(T)} &= c_i^{T,1} \cdot c_i^{T,2} \cdot \ldots \cdot c_i^{T,N} \end{aligned}$$

Voting. To cast a vote (without considering delegation) for voting option o voter i prepares the following tuple:

$$\langle \{\{id_i\}\}_{pk_T}, \{\{c_i^{(j)}\}\}_{pk_T}, \{\{o\}\}_{pk_T}, \sigma \rangle$$

Here σ signifies both the well-formedness proofs for o as well as proof of randomness knowledge for $\{\{c_i^{(j)}\}\}_{pk_T}$: namely, given $\{\{c_i^{(j)}\}\}_{pk_T} = (c_1, c_2, c_3) = (g_1^{r_1}, g_2^{r_2}, c_i^{(j)} h^{r_1+r_2})$ the voter proves the knowledge of randomness r_1 as $\log_{g_1} c_1$. The value of j is chosen depending on the voter's delegations where we distinguish the following cases:

1. If the voter does not intend to delegate her vote at a later point in time she sets $j = 1$.
2. If the voter might intend to delegate her vote at a later point in time she sets j as the lowest available priority: that is $j = T$ in case she did not delegate any vote yet or $j = j_d - 1$ if j_d is the highest priority that was already delegated.

Additionally, the voter i casts her identifier id_i in an encrypted manner which later serves for clustering ballots from the same voter with different credentials.

Delegating. To delegate her vote with priority $j = 2, \ldots, T$ to the proxy D_k the following protocol (see Figure 1) is executed.

1. The voter i chooses a random value x and sends the following tuple to one or more of the DS:

   $$\langle \{\{id_i\}\}_{pk_T}, \{(c_i^{(j)})^x\}_{pk_D}, \sigma, id_{D_k} \rangle$$

   Here $c_i^{(j)}$ is the j-th credential from her credential list $(c_i^{(1)}, \ldots, c_i^{(T)})$, id_i is the voter's index, σ is the proof of plaintext knowledge for $\{(c_i^{(j)})^x\}_{pk_D}$ and id_{D_k} is the identifier, e.g. the public key, of the chosen proxy. The voter also sends $x, \{\{id_i\}\}_{pk_T}$ to D_k over a private channel.

2. The DS computes $\{(c_i^{(j)})^x\}_{pk_T}$ from $\{(c_i^{(j)})^x\}_{pk_D}$ using proxy reencryption scheme and a designated-verifier proof using the public designated-verifier key of D_k that both $\{(c_i^{(j)})^x\}_{pk_T}$ and $\{(c_i^{(j)})^x\}_{pk_D}$ encrypt the same plaintext. The proof and the values of $\{(c_i^{(j)})^x\}_{pk_T}$, $\{(c_i^{(j)})^x\}_{pk_D}$ together with the voter's index $\{\{id_i\}\}_{pk_T}$ are being forwarded to D_k.
3. The proxy D_k verifies the proof and sends the signed value of $\{(c_i^{(j)})^x\}_{pk_D}$ back to the voter as confirmation.[6]
   She further computes $\{c_i^{(j)}\}_{pk_T}$ as $\{(c_i^{(j)})^x\}_{pk_T}^{1/x}$.

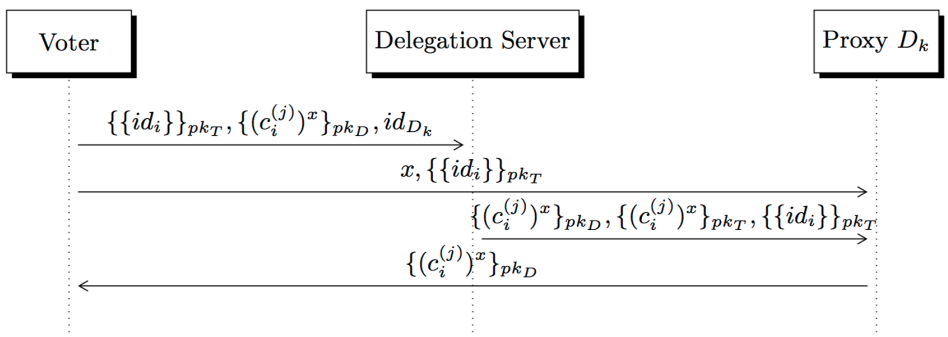

Fig. 1. Delegation of the voter id_i to the proxy D_k, with zero-knowledge proofs omitted.

Casting a delegated vote. To cast a delegated vote for a (unknown) voter X with (unknown) priority Y the proxy first calculates $\{\{c_X^{(Y)}\}\}_{pk_T}$. For this - given an encryption $\{c_X^{(Y)}\}_{pk_T} = (c_1, c_2)$ - she chooses a random value s and computes $\{\{c_X^{(Y)}\}\}_{pk_T} = (g_1^s, c_1, c_2 h^s)$. Then She encrypts her voting option o and prepares her ballot according to the voting process outlined above.

Cancelling a delegation. If the voter intends to withdraw one or several vote delegations (but not excluding delegation in general), she issues the highest prioritized unused credential, overriding all previously cast votes on her behalf.

Obfuscating the number of cast votes. The fake votes intended to hide the number of votes cast by a specific voter or her proxy are added accordingly to Spycher *et al.*. For each identifier id_i the tabulation tellers cast a random number of votes of the following form:

$$\langle \{\{id_i\}\}_{pk_T}, \{\{r\}\}_{pk_T}, \{\{o\}\}_{pk_T}, \sigma \rangle$$

[6] This can be done via a two-way anonymous channel or by publishing the signature on $\{(c_i^{(j)})^x\}_{pk_D}$ on the bulletin board.

Here r denotes a fake credential randomly drawn for each fake vote, a random valid voting option o, and the respective zero knowledge proofs σ. The number of fake votes for each voter is secret and is distributed as outlined in [18].

First anonymization. After all the votes have been cast, votes with invalid proofs are removed and excluded from further tallying. Thereafter, tuples of the form $\langle \{\{id\}\}_{pk_T}, \{\{c\}\}_{pk_T}, \{\{o\}\}_{pk_T} \rangle$ are anonymized by the application of a mix net.

Ballot clustering. After anonymizing the tuples the values of id are decrypted. For any index id_i appearing within at least one anonymized tuple the respective ordered list of credentials $(\{\{c_1^{(1)}\}\}_{pk_T}, \ldots, \{\{c_1^{(T)}\}\}_{pk_T})$ is obtained from the bulletin board. All tuples sharing the same id_i are clustered. The ordered of credentials is attached to each cluster, and the value id_i is removed from all tuples.

Second anonymization. All lists of resulting tuples together with the respective list of attached credentials are anonymized by the application of a mix net. Note that the order of ciphertexts within the tuples is preserved so that the priority order of delegation credentials remains intact.

Extracting votes to be counted. In addition to weeding out the votes with invalid credentials our goal is to tally only the vote with the highest priority in case delegation took place. In order to do this for any element of the cluster - namely a list of tuples and public credentials - the list of tuples is matched on the basis of plaintext equivalence tests (PET) in decreasing order against the list of public credentials. Starting with the highest priority credential c_1 PETs are executed between all credentials of submitted tuples until a match is found. If a match is found the value $\{\{o\}\}_{pk_T}$ is extracted from the matching tuple. Once the process has been terminated for all tuples a list of encrypted voting options $(\{\{o_1\}\}_{pk_T}, \ldots, \{\{o_n\}\}_{pk_T})$ has been extracted.

Third anonymization and result calculation. The list of encrypted voting options is anonymized by the application of a mix net. Eventually, the anonymized encrypted voting options are decrypted and the result is determined.

5 Security of the Proposed Scheme

In this section we consider the security evaluation of our scheme. We first summarize the security assumptions that need to be made, and then provide an informal security argument regarding the requirements given in Section 2.

5.1 Security Assumptions

We summarize the security assumptions specific to our scheme in the list below:

1. More than 2/3 of all tabulation tellers are trusted not to disclose private key shares to the adversary.
2. At least 1/3 of all tabulation tellers does not fail during decryption.
3. At least one of DS does not fail to forward the delegated votes to the proxy.
4. The DS does not disclose private information to the adversary.
5. The public key infrastructure for the proxies is trustworthy.
6. The private signing and designated-verifier keys of the proxy is not leaked.
7. Communication channels between the voters and the proxies are private.

5.2 Security Evaluation

We hold on to the assumptions of the original scheme which we provide in Section 3.2. The security properties of our scheme and the additional assumptions that are needed for them can then be evaluated as follows:

Vote integrity. The assumptions on integrity for voters remain the same as in the original scheme.

Availability. The election results can be computed under the assumption that at least $k = \frac{1}{3}n$ tabulation tellers participate in the decryption.

Vote secrecy for voters. The system provides probabilistic coercion resistance, as there are two scenarios where the coercer can tell whether the voter has obeyed. The first scenario occurs if the number of fake votes for some voter equals the known minimal number. The probability of this can be controlled with the choice of an appropriate distribution function for fake votes. In the second attack, the coercer requests all the delegation credentials from the voter, and casts a vote with priority j that is unknown to the voter. In that case, unless there is a vote cast with the same priority from another voter, the coercer knows whether the credential given to her was real. The success probability of such an attack can be reduced with a smaller value of T. For example, with $T = 3$ the voter can either vote herself, delegate once or choose one back-up proxy, which we assume to be sufficient functionality for most of the elections.

Outside of these scenarios, the system provides vote secrecy, and with it coercion resistance for the voters under the same assumptions as the original scheme with $k \leq \frac{1}{3}n$. Since the delegating credentials are generated and distributed in the same way as the voting credentials the voter can cheat the coercer who acts as a proxy by providing a fake credential instead. Even if the coercer is watching the voter directly during the delegation, the voter can input a random value as her delegating credential so that the coercer cannot distinguish it from the real one.

Delegation integrity. Casting a delegated vote without voter's permission or a delegated vote with a higher than given priority would require the knowledge of the corresponding credential $c_i^{(j)}$ for the voter i and priority j. Given that each one of those credentials is generated in the same way as the voter credentials in

the original scheme it holds that they are not leaked under the same assumptions. Another way to break the delegation integrity would be to intercept the value x used for blinding the credential forwarded to the proxy which requires control over the communication channel between voter and proxy. Furthermore, it must be assumed that the public key infrastructure for the proxies is trustworthy so that impersonation of a proxy to a voter is infeasible.

Delegation availability. The proxy receives the credential from the DS accompanied by the designated verifier proof. In case no credential is sent the voter gets no confirmation and thus is able to repeat the delegation by choosing another DS. For this it must be assumed that the proxy's private key is not leaked and the confirmation cannot be sent by someone else. However, the DS is capable of submitting an invalid credential if it can fake the designated verifier proof. Therefore, it must be assumed that the private designated-verifier key of the proxy is not being leaked.

Vote secrecy for proxies. Given an anonymous channel between the proxies and the bulletin board the secrecy of votes cast by proxies corresponds to the vote secrecy for the voters. An additional assumption is required that the DS is not collaborating with the adversary. In this case using a designated-verifier proof the proxy can fake the credentials resulting from the delegation process and present them to the coercer if forced to do so.

Delegation privacy. The proxy could be capable of learning the identity of the delegating voter in following ways: 1) by identifying the message's origin via network, 2) by learning the voter's ID, 3) by being able to assign the given delegating credential to the voter's identity. The communication channels between voters and proxies are anonymous, the voter ID is sent encrypted and only decrypted after anonymization. The delegating credentials are not otherwise leaked which is similar to the voter's credentials in the original scheme. This implies that the proxy is incapable of determining the identity of voters delegating to her. In case of a coerced proxy the coercer would not have a possibility of knowing whether the credential passed to them is valid and whether the voter has cast another vote herself. This corresponds with the coercion-resistance properties of the original scheme.

Delegation unprovability. Given the anonymous communication channels between the DS and the voters a proxy cannot prove for given credentials that they come from actual voters and are not simulated by the proxy herself. Moreover, due to the coercion resistance properties of the original scheme and its credential generation process the proxy herself cannot verify whether the credential she received is valid. This implies that the proxy is incapable of constructing a convincing proof of possessing any amount of delegated votes.

6 Conclusion

Proxy voting constitutes a new voting mode in which voters are able to delegate their right to vote on issues beyond their expertise. At the same time, it also opens new attack vectors as there is a legitimate possibility to transfer a vote to another person who could be a coercer. To address these issues we created an Internet proxy voting scheme which focuses on coercion resistance and is based on the well-known JCJ/Civitas scheme. It provides functionality that enables vote delegation while at the same time ensuring the security of the delegation.

As our extension introduces additional credentials for delegation, which might overwhelm voters, an important part of future work would be to improve usability of the scheme. In the future, we will consider the proposal by Neumann and Volkamer [15] to address the credential handling in coercion-resistant proxy voting.

Acknowledgements

This project (HA project no. 435/14-25) is funded in the framework of Hessen ModellProjekte, financed with funds of LOEWE – Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz, Förderlinie 3: KMU-Verbundvorhaben (State Offensive for the Development of Scientific and Economic Excellence).

This paper has been developed within the project 'VALID' - Verifiable Liquid Democracy - which is funded by the Polyas GmbH.

References

1. Bayer, S., Groth, J.: Efficient zero-knowledge argument for correctness of a shuffle. In: Advances in Cryptology–EUROCRYPT 2012, pp. 263–280. Springer (2012)
2. Bernhard, D., Neumann, S., Volkamer, M.: Towards a practical cryptographic voting scheme based on malleable proofs. In: E-Voting and Identify, pp. 176–192. Springer (2013)
3. Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In: Advances in Cryptology–ASIACRYPT 2012, pp. 626–643. Springer (2012)
4. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Advances in Cryptology–CRYPTO 2004. pp. 41–55. Springer (2004)
5. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Advances in Cryptology—CRYPTO'92. pp. 89–105. Springer (1993)
6. Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: Toward a secure voting system. Tech. rep. (2008)
7. Delaune, S., Kremer, S., Ryan, M.: Verifying privacy-type properties of electronic voting protocols: A taster. In: Towards Trustworthy Elections, pp. 289–309. Springer (2010)
8. Grewal, G.S., Ryan, M.D., Bursuc, S., Ryan, P.Y.: Caveat coercitor: Coercion-evidence in electronic voting. In: Security and Privacy (SP), 2013 IEEE Symposium on. pp. 367–381. IEEE (2013)

9. Jakobsson, M.: On quorum controlled asymmetric proxy re-encryption. In: Public Key Cryptography. pp. 112–121. Springer (1999)
10. Jakobsson, M., Juels, A.: Mix and match: Secure function evaluation via ciphertexts. In: Advances in Cryptology—ASIACRYPT 2000, pp. 162–177. Springer (2000)
11. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated verifier proofs and their applications. In: Advances in Cryptology—EUROCRYPT'96. pp. 143–154. Springer (1996)
12. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Proceedings of the 2005 ACM workshop on Privacy in the electronic society. pp. 61–70. ACM (2005)
13. Langer, L., Schmidt, A., Buchmann, J., Volkamer, M.: A taxonomy refining the security requirements for electronic voting: analyzing helios as a proof of concept. In: Availability, Reliability, and Security, 2010. ARES'10 International Conference on. pp. 475–480. IEEE (2010)
14. Neumann, S., Kahlert, A., Henning, M., Richter, P., Jonker, H., Volkamer, M.: Modeling the german legal latitude principles. In: 5th International Conference on eParticipation (ePart 2013). Lecture Notes in Computer Science, vol. 8075, pp. 49–56. Springer (Sep 2013)
15. Neumann, S., Volkamer, M.: Civitas and the real world: Problems and solutions from a practical point of view. In: Availability, Reliability and Security (ARES), 2012 Seventh International Conference on. pp. 180–185. IEEE (2012)
16. Neumann, S., Volkamer, M.: A holistic framework for the evaluation of internet voting systems. Design, Development, and Use of Secure Electronic Voting Systems pp. 76–91 (2014)
17. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of cryptology 4(3), 161–174 (1991)
18. Spycher, O., Koenig, R., Haenni, R., Schläpfer, M.: A new approach towards coercion-resistant remote e-voting in linear time. In: Proceedings of the 15th international conference on Financial Cryptography and Data Security. pp. 182–189. Springer-Verlag (2011)
19. Tchorbadjiiski, A.: Liquid democracy diploma thesis. RWTH AACHEN University, Germany (2012)
20. Terelius, B., Wikström, D.: Proofs of restricted shuffles. In: Progress in Cryptology–AFRICACRYPT 2010, pp. 100–113. Springer (2010)
21. Zwattendorfer, B., Hillebold, C., Teufl, P.: Secure and privacy-preserving proxy voting system. In: e-Business Engineering (ICEBE), 2013 IEEE 10th International Conference on. pp. 472–477. IEEE (2013)