A Fair and Robust Voting System by Broadcast

Dalia Khader[1], Ben Smyth[2], Peter Y. A. Ryan[1], and Feng Hao[3]

[1]Universite du Luxembourg, SnT,
Luxembourg
{dalia.khader | peter.ryan}@uni.lu

[2]Toshiba Corporation,
Kawasaki, Japan
toshiba@bensmyth.com

[3]Newcastle University
Newcastle, United Kingdom
Feng.hao@newcastle.ac.uk

Abstract: Hao, Ryan, and Zieliński (2010) propose a two-round decentralized voting protocol that is efficient in terms of rounds, computation, and bandwidth. However, the protocol has two drawbacks. First, if some voters abort then the election result cannot be announced, that is, the protocol is not robust. Secondly, the last voter can learn the election result before voting, that is, the protocol is not fair. Both drawbacks are typical of other decentralized e-voting protocols. This paper proposes a recovery round to enable the election result to be announced if voters abort, and we add a commitment round to ensure fairness. In addition, we provide a computational security proof of ballot secrecy.[1,2]

1 Introduction

Paper-based elections derive security properties from physical characteristics of the real world. For example, marking a ballot in isolation inside a polling booth and depositing the completed ballot into a locked ballot box provides privacy; the polling booth also ensures that voters cannot be influenced by other voters, and the locked ballot box prevents the announcement of early results, thereby ensuring fairness; and the transparency of the whole election process from ballot casting to tallying alongside the impossibility of altering the markings on a paper ballot sealed inside a locked ballot box gives an assurance of correctness and facilitates verifiability. Moreover, the combination of these physical constraints ensures a robust voting scheme. Replicating these attributes

[1] Smyth's work was partly done at Loria, CNRS & INRIA Nancy Grand Est, France as part of the ProSecure project, which is funded by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement n0258865, and the ANR-07-SeSur-002 AVOTE project. Khader & Ryan conducted their work as part of the SeRVTS-C09/IS/06 project, funded by the FNR.

[2] This paper has been published in Word format after conversion from Latex. We have tried to eliminate the errors introduced during this conversion process, however, we suspect some errors remain. Accordingly, we refer the reader to the LaTeX created document, which is available on the authors' web pages.

in a digital setting has proven to be difficult and, hence, the provision of secure electronic voting systems is an active research topic, first inspired by Chaum [Cha81].
Two classes of e-voting systems can be distinguished: (i) Decentralized e-voting systems, where voters run a multi-party computational protocol without any additional parties, for example [Sch99, KY02, Gro04, HRZ10] and (ii) Centralized e-voting systems, where election administrators run the election, for example [JCJ05, XSH+07, RT09]. Decentralized systems are typically designed for small-scale elections with a focus on security with minimal trust assumptions; whereas, centralized schemes are typically designed for large-scale elections and rely upon stronger trust assumptions to enable scalability, usability, and robustness. In this paper we focus on decentralized voting schemes.

Kiayias & Yung [KY02], Groth [Gro04] and Hao, Ryan, and Zieliński [HRZ10] have come to a consensus that the following properties are essential for decentralized voting schemes:

- Perfect ballot secrecy: A voter's vote is not revealed to anyone else, modulo what can be computed from the published tally.
- Self-tallying: At the end of the protocol, voters and observers can tally the election result from public information.
- Fairness: Nobody has access to partial results before the *deadline*. The precise definition of deadline varies in the literature. In this paper, we suppose fairness is satisfied if no one has access to partial results before casting their vote. (Note that our definition would permit a voter to abort the protocol after having observed partial results but could not change their vote.)
- Dispute-freeness: A scheme is dispute-free if anyone can verify that the protocol was run correctly and that each voter acted according to the rules of the protocol.

In addition, we also consider *robustness*.

- Robustness: A corrupt voter cannot prevent the election result from being announced.

Hao, Ryan, and Zieliński [HRZ10] propose an election scheme, which makes some progress toward satisfying these properties. However, their scheme is neither robust nor fair: in particular, a single voter can prevent the election result from being announced and the last voter can cast her vote with full knowledge of the election result.

1.1 Contribution

We propose a variant of the Hao, Ryan, and Zieliński [HRZ10] election scheme that ensures fairness and robustness, and we formally prove ballot secrecy using provable security techniques.

2 Preliminaries

This section presents the assumptions and cryptographic primitives that will be used to construct our scheme. We shall start with some notations and conventions used throughout the paper. Let $\mathcal{H}$ denote a hash function and (p, q, g) be cryptographic parameters, where p and q are large primes such that $q|p-1$ and g is a generator of the multiplicative subgroup of $\mathbf{Z}_p^*$ of prime order q. In some of our security proofs we rely on the assumption that the Decisional Diffie-Hellman (DDH) problem is hard, which is a logical consequence of using ElGamal-style encryption as a building block for our protocol .

Definition (Decisional Diffie-Hellman problem)

Given integers $g^a, g^b, g^c \in \mathbf{Z}_p^*$ *and* $\mathbf{a, b, c} \in \mathbb{Z}_q^*$ are chosen randomly.

The distribution $\{(g, g^a, g^b, g^{ab})\}$ *is computationally indistinguishable from* $\{(g, g^a, g^b, g^c)\}$.

Our scheme is reliant on signatures of knowledge to ensure secrecy and integrity and to ensure voters encrypt valid votes; we now recall suitable primitives.

2.1 Knowledge of Discrete Logs

Proof Statement: Proving knowledge of x, given h where $h \equiv g^x \bmod p$ [CEGP87, CEG88, Sch90][3].

Sign: Given x, select a random nonce $w \in_R \mathbf{Z}_q^*$ and compute

- Witness $g' = g^w \bmod p$
- Challenge $c = \mathcal{H}(g') \bmod q$
- Response $s = w + c \cdot x \bmod q$.

Output Signature (g', s)

Verify: Given h and signature (g', s), check $g^s \equiv g' \cdot h^c \pmod p$, where $c = \mathcal{H}(g') \bmod q$.

A valid proof asserts knowledge of x such that $x = \log_g h$, i.e., $h \equiv g^x \bmod p$.

[3] The challenge can also include the ID of the participant to prevent replay attacks such that $c = \mathcal{H}(\mathbf{ID} || g^{t_r}) \bmod q$

2.2 Equality Between Discrete Logs

Proof Statement: Proving knowledge of the discrete logarithm x to bases $f, g \in \mathbf{Z}_p^*$, given h, k where $h \equiv f^x \bmod p$ and $k \equiv g^x \bmod p$ [Ped91, CP93].

Sign: Given f, g, x, select a random nonce $w \in_R \mathbf{Z}_q^*$. Compute

- Witnesses $f' = f^w \bmod p$ and $g' = g^w \bmod p$
- Challenge $c = \mathcal{H}(f', g') \bmod q$
- Response $s = w + c \cdot x \bmod q$.

Output signature as (f', g', s)

Verify: Given f, g, h, k and signature (f', g', s, c), check $f^s \equiv f' \cdot h^c \pmod p$ and $g^s \equiv g' \cdot k^c \pmod p$, where $c = \mathcal{H}(f', g') \bmod q$.

A valid proof asserts $\log_f h = \log_g k$, i.e., there exists an x such that $h \equiv f^x \bmod p$ and $k \equiv g^x \bmod p$. This signature of knowledge scheme can be extended to a disjunctive proof of equality between discrete logs (see below.)

2.3 Disjunctive Proof of Equality Between Discrete Logs

Proof Statement: Given that $(a, b) = (g^x, g^{y-x} \cdot g^m)$ contains message m, prove that $m \in \{min, \ldots, max\}$ for some parameters $min, max \in \mathbf{N}$, where $min < \max$ [CGS97, CDS94].

Sign: Given (a, b) such that $a \equiv g^x \bmod p$ and $b \equiv h^x \cdot g^m \bmod p$ for some nonce $x \in \mathbf{Z}_q^*$, where plaintext $m \in \{min, \ldots, max\}$.

For all $i \in \{min, \ldots, m-1, m+1, \ldots, max\}$, compute challenge $c_i \in_R \mathbf{Z}_q^*$, response $s_i \in_R \mathbf{Z}_q^*$, and witnesses $a_i = \dfrac{g^{s_i}}{a^{c_i} \bmod p}$ and $b_i = \dfrac{h^{s_i}}{\left(\dfrac{b}{g^i}\right)^{c_i} \bmod p}$.

Select a random nonce $w \in_R \mathbf{Z}_q^*$. Compute witnesses $a_m = g^w \bmod p$ and $b_m = h^w \bmod p$,

challenge

$$c_m = \mathcal{H}(a, b, a_{min}, b_{min}, \ldots, a_{max}, b_{max}) - \sum_{i \in \{min, \ldots, m-1, m+1, \ldots, max\}} c_i \pmod{q}$$

and response $s_m = w + x \cdot c_m \bmod q$.

To summarize, we have

- Witnesses $(a_{min}, b_{min}), \ldots, (a_{max}, b_{max})$
- Challenge $c_{min}, \ldots, c_{max}$
- Response $s_{min}, \ldots, s_{max}$

Output signature of knowledge (a_i, b_i, c_i, s_i) for all $i \in \{min, \ldots, max\}$.

Verify: Given (a, b) and $(a_{min}, b_{min}, c_{min}, s_{min}, \ldots, a_{max}, b_{max}, c_{max}, s_{max})$, for each $min \leq i \leq \max$ check $g^{s_i} \equiv a_i \cdot a^{c_i} \pmod{p}$ and $h^{s_i} \equiv b_i \cdot \left(\frac{b}{g^i}\right)^{c_i} \pmod{p}$.

Finally, check. $\mathcal{H}(a, b, a_{min}, b_{min}, \ldots, a_{max}, b_{max}) \equiv \sum_{min \leq i \leq max} c_i \pmod{q}$.

A valid proof asserts that (a, b) contains the message m such that $m \in \{min, \ldots, max\}$.

3 Voting Scheme

In this section, we present a variant of the Hao, Ryan, and Zieliński [HRZ10] election scheme, which guarantees fairness without any computational overhead and, moreover, we introduce a recovery procedure to ensure robustness.

In [HRZ10, Gro04, KY02] the authors assume authenticated public channels to prevent a participant from voting multiple times and to ensure eligibility of voters: we adopt the same assumption.

3.1 Toward Fairness

In this section, we extend the Hao, Ryan, and Zieliński [HRZ10] protocol to include an additional *Commitment Round* to ensure fairness.

Given a number of voters $n \in \mathbb{N}$, the scheme proceeds as follows:

Setup Round: Each voter $i \in n$ selects a private key $x_i \in_R \mathbb{Z}_q^*$ and computes the corresponding public key $a_i = g^{x_i} \bmod p$. Each voter has to prove that a_i has been constructed correctly by proving knowledge of x_i (§2.1).

Commitment Round: Each voter $i \in n$ computes h_i as follows.

$$h_i = g^{(x_1+\cdots+x_{i-1})-(x_{i+1}+\cdots+x_n)} = \frac{\prod_{j=1}^{i-1} a_j}{\prod_{j=i+1}^{n} a_j}$$

The voter constructs $b_i = h_i^{x_i} \cdot g^{v_i}$, where $v_i \in \{0,1\}$ is the voter's vote.
A disjunctive proof of equality between discrete logarithms $\log_T (a_i / g^{v_i}) = \log_T (h_i / g^{1-v_i})$ and $\log_T (a_i / g^{v_i}) = \log_T (h_i / g^{1-v_i}) / g$ is computed to prove that $v_i \in \{0,1\}$ (§2.3). Note that the signature includes challenge c_{v_i}, which acts as a computationally binding commitment to values a_i and b_i. Furthermore, the value b_i is not published in this round.

Voting Round: Each voter publishes b_i.

In the above protocol description, the pair (a_i, b_i) is an ElGamal-style encryption of the voter's vote, where v_i is the plaintext, x_i is a nonce, and h_i is the public encryption key; ballot secrecy is ensured because no coalition can recover a voter's vote.

As an alternative to the above commitment round, a voter could publish a hash of the values output during the voting round in [HRZ10], however, we have observed that the signature of a knowledge scheme has a computationally binding and computationally hiding commitment to the vote v_i since the value b_i is hashed among the other elements of the signature of knowledge. Thus, a hash of the values output in the voting round in [HRZ10] is not necessary.
In [HRZ10] the last voter can vote having complete knowledge of the election result. This limitation is avoided in our scheme with an additional round, more precisely, the commitment round and the voting round correspond to a single voting round in [HRZ10]. The separation of rounds exploits the result by Cramer *et al.* [CFSY96] (Lemma 1). Namely, no partial results are available during the commitment round in order to ensure Fairness.

Lemma 1: The signature of knowledge produced during the commitment round demonstrates $v \in \{0,1\}$ without releasing the actual value of v .

Once all voters have completed the protocol, the self-tallying property allows the election result to be derived by observers and voters.

Self-Tallying: Given some protocol output such that all the signatures of knowledge hold the result $v \log_g V$, where V is defined below:

$$V = \prod_{i=1}^{n} b_i = \prod_{i=1}^{n} h_i^{x_i} \cdot g^{v_i} = g^{\sum_{i=1}^{n} v_i}$$

In our scheme, the result v is the sum of the votes for 1; the votes for 0 can be trivially derived as $n - v$.

Formally, the computation $v \log_g V$ follows from Proposition 2, as shown by Hao, Ryan, and Zieliński. Although the computation of the discrete logarithm is hard in general, we know that the election result v is such that $1 \leq v \leq n$ and, therefore, the search for the value v is feasible with complexity of $O(n)$ by linear search or $O(\sqrt{n})$ using the Pollard-Lambda [Pol00] or baby-step giant-step algorithm [Sha71] (see also [LL90,3.1]).

Proposition 2:

Given integer $n \in \mathbf{N}$, we have for all $x_i \in \mathbf{Z}_q^*$ and $y_i = (x_1 + \cdots + x_{i-1}) - (x_{i+1} + \cdots + x_n)$ the $\sum_{i=1}^{n} x_i \cdot y_i = 0$.

3.2 Robustness

In the protocol by Hao, Ryan, and Zieliński a voter can prevent the election result from being announced by aborting. In this section, we introduce an efficient *recovery round* to enable the election result to be announced even if voters abort. Moreover, our recovery round maintains the security of the scheme; in particular, no votes can be modified or revealed during the recovery round.

Let us suppose $\mathcal{L}$ is the set of voters that submitted valid ballots in the voting round, where $|\mathcal{L}| < n$, that is, a subset of voters either did not vote or submitted an invalid signature of knowledge. A recovery round can be executed as follows to allow the election result to be announced:

Recovery Round: Each voter $i \in \mathcal{L}$ computes $\widehat{h}_i$ as follows:

$$\widehat{h}_i = \frac{\prod_{j \in \{i+1,\ldots,n\} \setminus \mathcal{L}} a_j}{\prod_{j \in \{1,\ldots,i-1\} \setminus \mathcal{L}} a_j}$$

Each voter publishes $\widehat{h}_i^{x_i}$ together with a signature of knowledge asserting $\log_{\top g} [\![a_i]\!] = \log_{\top}(\hat{h}_i) [\![\hat{h}_i^{\top}(x_i)]\!]$ (§2.2).

In the recovery round, the outputs $\left\{ \widehat{h}_i^{x_i} \middle| i \in \mathcal{L} \right\}$ act as cancellation tokens during tallying to eliminate the need for private keys of voters whom did not participant in the voting round (see Table 1 for a simple illustration).

No	First round	Second round	Third round	Recovery
1	g^{x_1}	commitment	$g^{x_1 y_1} = g^{x_1(-x_2 - x_3 - x_4 - x_5)}$	$\widehat{h}_1^{x_1} = g^{x_1(x_2 + x_4)}$
2	g^{x_2}	commitment	Abort	--
3	g^{x_3}	commitment	$g^{x_3 y_3} = g^{x_3(x_1 + x_2 - x_4 - x_5)}$	$\widehat{h}_3^{x_3} = g^{x_3(x_4 - x_2)}$
4	g^{x_4}	commitment	Abort	--
5	g^{x_5}	commitment	$g^{x_5 y_5} = g^{x_5(x_1 + x_2 + x_3 + x_4)}$	$\widehat{h}_5^{x_5} = g^{x_5(-x_2 - x_4)}$

Table 1. Example of recovery: With no loss of generality, we assume $n = 5$ and all participating voters send ``no" votes. Also, we have omitted the mention of ZKPs, as it is not needed for this illustration. Notice that data sent in the recovery round cancel out the effects of the drop-outs from the final tallying.

Suppose $\mathcal{L}'$ is the set of voters that broadcast valid values in the recovery round such that $\mathcal{L}' = \mathcal{L}$, then the self-tallying property allows the election result to be derived by observers and voters; otherwise, another recovery round is required by voters $\mathcal{L}'$.
Given the output of the recovery round for all voters $\mathcal{L}$, such that all the signatures of knowledge hold, the result is $v = \log_g V$, where V is defined below:

$$V = g^{\sum_{i \in \mathcal{L}} v_i} = \prod_{i \in \mathcal{L}} \hat{h}_i^{x_i} \cdot h_i^{x_i} \cdot g^{v_i} = \prod_{i \in \mathcal{L}} \hat{h}_i^{x_i} \cdot b_i$$

Once again, the result v is the sum of the votes for 1.
Formally, the computation $v = \log_g V$ follows from Proposition 3.3.

Proposition 3.3:
Given the integer $n \in \mathbb{N}$ *and set* $\mathcal{L} \subset \{1, \ldots, n\}$,
we have for all $x_i \in_R \mathbb{Z}_q^*$, $y_i = (x_1 + \cdots + x_{i-1}) - (x_{i+1} + \cdots + x_n)$

and $\hat{y}_i = \sum_{j \in \{i+1, \ldots, n\} \setminus \mathcal{L}} x_j - \sum_{j \in \{1, \ldots, i-1\} \setminus \mathcal{L}} x_i$

that $\sum_{j \in \mathcal{L}} (x_j \cdot y_j) + (x_j \cdot \hat{y}_j) = 0$.

Proof:
We have

$$\sum_{j \in \mathcal{L}} (x_j \cdot y_j) + (x_j \cdot \hat{y}_j) = \sum_{j \in \mathcal{L}} x_j \cdot (y_j + \hat{y}_j)$$

and

$$y_j + \hat{y}_j = \sum_{k \in \{1, \ldots, j-1\} \cap \mathcal{L}} x_k - \sum_{k \in \{j+1, \ldots, n\} \cap \mathcal{L}} x_i .$$

Note that if a voter decides $|\mathcal{L}|$ is too small to maintain privacy (e.g., when $|\mathcal{L}| = 2$), then she can decide not to join the recovery round and abort; in this case, the voter obtains an assurance of ballot secrecy (under the DDH assumption), but her vote is not included in the tallying procedure, i.e., her vote is discarded.

Discussion: Re-running an Election is not Equivalent to Recovery.
Critics may argue that the recovery round is not necessary because elections can be efficiently re-run. However, two runs of an election protocol do not gaurantee the same result and this may lead to attacks. For example, suppose there is a referendum to decide whether electronic voting should be adopted.I In this setting, opponents of electronic voting could force a re-run of the referendum in the hope that the system's failure to announce the election result in the first run will sway the electorate's opinion in a re-run. This can occur in [HRZ10]. For example, all voters behave honestly except Mallory, who forces a re-run and thus has the opportunity to influence the opinion of the electorate; moreover, Mallory can plausibly deny that she is malicious, for example, by claiming that she dropped her laptop and lost her key.

3.3 Multi-Candidate Voting Scheme

We adopt the technique used in [HRZ10] to extend our scheme to multi-candidate elections. Assuming we have n voters and k candidates. A value m is chosen such that it is the smallest integer where $2^m > n$. The main modification to handle multi-candidate elections is during the voting round: the voter's choice is $v_i \in \left\{2^{0}, 2^{(k-1)}, \ldots, 2^{(k-1)m}\right\}$.

The setup and recovery rounds are unchanged. The commitment round uses a signature of knowledge (§2.3) where $min = 2^0$ and $max = 2^{(k-1)m}$.

The tallying will cause $V = g^{\sum_{i=1}^{n} v_i} = g^v$, however $v = 2^0 c_0 + 2^{(k-1)} c_1 + 2^{(k-1)2} c_2 \ldots + 2^{(k-1)m} c_{k-1}$, where c_j is the number of votes that went for candidate j for any $j \in \{0, \ldots, k-1\}$. The value $v \leq 2^{(k-1)m} n$ can be efficiently computed (the maximum value is if all voters vote for the last candidate) using a baby-step giant-step algorithm (this is possible because the values of k tend to be small), and $c_1, \ldots, c_k$ can be recovered using the super-increasing nature of the encoding with the help of algorithms such as the knapsnack algorithm.

4 Security and Performance Analysis

This section presents a computational security proof of ballot secrecy (§1) and compares our scheme with existing decentralized voting protocols in the literature (§2).

4.1 Ballot Secrecy

Hao, Ryan & Zieliński [HRZ10] provide strong arguments to show that ballot secrecy is satisfied in their scheme under the DDH assumption.
In this work we add a formal proof of Ballot Secrecy using provable security techniques and game models, assuming honest-but-curious voters. This implies participants are honestly creating the input of the protocol but curious to know the others' inputs. This assumption is a common practice [Gro04]. Under this assumption, the signatures of knowledge can be dropped from the game model. This game model is for proving ballot secrecy. Since these signatures of knowledge reveal minimum information, the first signature reveals one bit proving knowledge of x_i; the signatures of knowledge in the commitment and voting round reveal that v_i belongs to a set of values (the adversary already knows this set); and the last signature reveals another bit proving equality of x_i to the bases $g, \hat{h}_i$. None of the information revealed by the signatures of knowledge is related to the final value of the vote in an interesting manner. In our game model, we allow the adversary to query an oracle $CrptVoter(i)$ where the challenger responds with x_i.

Ballot Secrecy (BS-Security): We say a decentralized voting scheme is BS-Secure, if no polynomially-bounded adversary $\mathbb{A}$ has a non-negligible advantage against the challenger $\mathbb{C}$ in the following ballot secrecy game:

- Set-up Round: $\mathbb{C}$ chooses all x_i and publishes all g^{x_i}, for $i \in \{1, \dots, n\}$
- Challenge: The adversary chooses voters j and k that have not been queried in $CrptVoter$. The challenger randomly chooses one of j, k to have voted as 1 and the other as 0. We refer to the voter who voted 1 as pv. The challenger randomly chooses $pv \in \{j, k\}$ to vote 1 and the remaining voter to vote 0.
- Voting Round: The adversary can call for the voting round to start. The adversary gets to vote on behalf of the corrupted voters. Furthermore, the adversary gets to abort certain voters causing the need for a recovery round to be executed; he can select the voters to abort.
- Recovery Round: If a voter aborts, then the recovery round is executed. The adversary is permitted to select voters to abort during the recovery round, forcing the recovery round to be re-run.
- Guess Phase: The adversary outputs a $guess \in \{j, k\}$.

The adversary $\mathbb{A}$ may query the oracle $CrptVoter(i)$, with the restriction that $i \in \mathbb{Z}\{j,k\}$just after the game is setup and until the guess phase.

To win the game the adversary must select $guess \in \{j,k\}$ such that $guess = pv$ with a probability greater than guessing, we say that ballot secrecy is satisfied when this is not the case.

Definition 1, *(Ballot Secrecy Security):*

The voting scheme is BS-Secure if for all polynomial time adversaries, the $Pr|guess = pv| - \frac{1}{2} \leq \epsilon$ „ ϵ is negligible.

Now we show that if an adversary who can win the game above exists, then there exists a simulator that can break the DDH Problem. We shall prove the following theorem via contradiction.

Theorem 2: If there exist an adversary that wins the BS model above, then there exist a simulator that can solve the DDH problem.

Proof.:

Assume we have a tuple g^a, g^b, g^c where $c \in \{ab, random\}$. The simulator assumes $a = x_k$ and $b = x_j$. For the setup round the values $g^{x_k} = g^a$ and $g^{x_j} = g^b$ are submitted. Simulating the vote round is done as follows:

- For (v_k, v_j): The simulator tosses a fair coin of $\{0,1\}$, v_k is equal to the output of the coin and v_j is the opposite value.
- For (x_k): Simulator needs to compute $g^{x_k y_k} g^{v_k}$. The value g^{v_k} is simple to compute given the previous coin toss. Compute:

$$g^{x_k y_k} = g^{a y_k} = g^{a\left(\left(x_1 + \cdots + x_{k-1}\right) - \left(x_{k+1} + \cdots + x_n\right)\right)}.$$
$$g^{x_k y_k} = (g^{a x_1} . g^{a x_2} \ldots g^{a x_{k-1}} . g^{-a x_{k+1}} \ldots g^{-a x_n}).$$

  Note that all values of x_i are known to the challenger except x_j , and the simulator replaces the term $g^{a x_j} = g^c$. This becomes a valid input in the voting round if and only if $c = ab$. The same technique can be used to run the recovery round. If $c = ab$, then the round would be simulating the real protocol, regardless of the number of times the round is executed.
- For (x_j): Simulator performs the same computations as for x_k and replaces the term $g^{a x_k} = g^c$.

If $c = ab$ and, given the assumption that there an adversary that wins the privacy game exists, then the adversary will definitely return the right value among $\{j, k\}$ and the simulator will guess that $c = ab$, but if the adversary of the privacy game aborts, then $c = random$.

Note that the same proof can be extended to hold for multi-candidate schemes

4.2 Performance Comparison

We compare our scheme with existing decentralized voting protocols (Table 1). It is immediately apparent that our scheme provides better performance than [KY02] and [Gro04], and we add an additional round in comparison with [HRZ10], this additional round is introduced to achieve fairness.

Protocol	[KY02]	[Gro04]	[HRZ10]	Our scheme
Rounds	3	n+1	2	3
Exponentials	2n + 2	4	2	2
Knowledge of d.logs	n + 1	2	1	1
Equality of d.logs	n	1	0	0
Disjunctive equality of d.logs	1	1	1	1

Table 2: Performance summary per voter

Performance of Recovery: We omit the cost of the recovery round from Table 2 since the other schemes are not robust. The additional costs associated with recovery are as follows: one additional exponential and one additional equality of d.logs, per voter, per round.

Performance of Multi-Candidates: The scalability of the schemes in Table 2 to multi-candidate elections are all similar. In our scheme, the additional computation during the commitment round is linear to the number of candidates and self-tallying requires execution of the Knapsnack algorithm.

Optimisations: We highlight two optimizations**:**

1. In [HRZ10, Gro04, KY02] the authors assume that each voter has a one-way authenticated broadcast channel. This assumption was made for two reasons: to detect a voter who is casting more than one vote and to ensure that only eligible voters can vote. One might be able to relax this assumption: authenticated channels are only needed in the first round. Under this assumption, the signatures of knowledge can be used to ensure that security is preserved in later rounds, in particular, witness that the value a_i (implicitly implying x_i) has been used in every round of the protocol and also during tallying; it should follow that authentication of a_i is sufficient for security. This could be achieved by authenticating the first round only. We therefore think the assumption that all communication must use authenticated channels might be relaxed in our protocol and in the protocol proposed in [HRZ10]. The savings associated with this weaker assumption are dependent upon the implementation of an authenticated channel and studying this optimization remains as a possibility for future work.
2. Let us consider a variant of our scheme with two rounds: the voter sends the ballot during the commitment round. If all voters participate in two rounds, then we have the original scheme [HRZ10]; in this case, fairness is not provided. However, if one voter completes three rounds, then fairness is provided, as we shall now argue: Let $\{x_1, \ldots, x_n\}$ be the private keys of voters. Suppose voters publish $b_1, \ldots, b_{k-1}, b_{k+1}, \ldots, b_n$ during the commitment round (as per the original scheme [HRZ10]) and the remaining voter only publishes her signature of knowledge. Self-tallying the published ballots produces the following:

$$V = \prod_{i=1, i=\not k}^{n} b_i = \prod_{i=1, i=\not k}^{n} h_i^{x_i} \cdot g^{v_i} = {b_k}^{-1} g^{\sum_{i=1}^{n} v_i} = h_k^{-x_k} \cdot g^{-v_k} g^{\sum_{i=1}^{n} v_i}$$

Witness that no partial election result can be derived from V without b_k, hence fairness is achieved assuming one voter completes three rounds of the protocol.

5 Conclusion

We present a fair and robust variant of the decentralized electronic voting protocol proposed by Ryan & Zieliński [HRZ10], and prove that our scheme satisfies perfect ballot secrecy under the DDH assumption. Moreover, our scheme is self-tallying and dispute-free. Furthermore, we have shown that our scheme is efficient when compared to existing decentralized voting schemes from the literature.

Bibliography

[CDS94] Ronald Cramer, Ivan Damga˚rd, and Berry Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In CRYPTO'94, volume 839 of LNCS, pages 174–187. Springer, 1994.

[CEG88] David Chaum, Jan-Hendrik Evertse, and Jeroen van de Graaf. An Improved Protocol for Demonstrating Possession of Discrete Logarithms and Some Generalizations. In EUROCRYPT'87, volume 304 of LNCS, pages 127–141. Springer, 1988.

[CEGP87] David Chaum, Jan-Hendrik Evertse, Jeroen van de Graaf, and Rene´ Peralta. Demonstrat- ing Possession of a Discrete Logarithm Without Revealing It. In CRYPTO'86, volume 263 of LNCS, pages 200–212. Springer, 1987.

[CFSY96] Ronald Cramer, Matthew K. Franklin, Berry Schoenmakers, and Moti Yung. Multi-Autority Secret-Ballot Elections with Linear Work. In EUROCRYPT'96, volume 1070 of LNCS, pages 72–83. Springer, 1996.

[CGS97] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. In Eurocrypt, pages 103–118. Springer- Verlag, 1997.

[Cha81] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM, 24:84–90, February 1981.

[CP93] David Chaum and Torben P. Pedersen. Wallet Databases with Observers. In CRYPTO'92, volume 740 of LNCS, pages 89–105. Springer, 1993.

[Gro04] Jens Groth. Efficient Maximal Privacy in Boardroom Voting and Anonymous Broadcast. In FC'04, volume 3110 of LNCS, pages 90–104. Springer, 2004.

[HRZ10] Fao Hao, Peter Y. A. Ryan, and Piotr Zielin´ ski. Anonymous voting by two-round public discussion. Journal of Information Security, 4(2):62 – 67, 2010.

[JCJ05] A. Juels, D. Catalano, and M. Jakobsson. Coercion-Resistant Electronic Elections. In Proc. of Workshop on Privacy in the Electronic Society (WPES05), Alexandria, VA, USA - November 7, 2005, pages 61–70, 2005.

[KY02] Aggelos Kiayias and Moti Yung. Self-tallying Elections and Perfect Ballot Secrecy. In PKC'02, volume 2274 of LNCS, pages 141–158. Springer, 2002.

[LL90] Arjen K. Lenstra and Hendrik W. Lenstra Jr. Algorithms in Number Theory. In Jan van Leeuwen, editor, Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity, chapter 12, pages 673–716. MIT Press, 1990.

[Ped91] Torben P. Pedersen. A Threshold Cryptosystem without a Trusted Party. In EURO-CRYPT'91, number 547 in LNCS, pages 522–526. Springer, 1991.

[Pol00] John M. Pollard. Kangaroos, Monopoly and Discrete Logarithms. J. Cryptology, 13(4):437–447, 2000.

[RT09] Peter Y. A. Ryan and Vanessa Teague. Pretty Good Democracy. In Proc. of the 17th Security Protocols Workshop, Cambridge, UK, 2009, LNCS. Springer, 2009.

[Sch90] Claus-Peter Schnorr. Efficient Identification and Signatures for Smart Cards.In CRYPTO'89, volume 435 of LNCS, pages 239–252. Springer, 1990.

[Sch99] Berry Schoenmakers. A Simple Publicly Verifiable Secret Sharing Scheme and its Ap- plication to Electronic Voting. In CRYPTO'99, volume 1666 of LNCS, pages 148–164. Springer, 1999.

[Sha71] Daniel Shanks. Class number, a theory of factorization and genera. In Number Thory Institute, volume 20 of Symposia in Pure Mathematics, pages 415–440. American Mathematical Society, 1971

[XSH+ 07] Zhe Xia, Steve Schneider, James Heather, Peter Y. A. Ryan, David Lundin, Roger Peel, and Philip Howard. Preˆt a` Voter: All-In-One. In Proc. of the IAVoSS Workshop On Trustworthy Elections (WOTE 2007), June 20-21, 2007