E2E Verifiable Electronic Voting System for Shareholders

Samiran Bag
University of Warwick
United Kingdom
Email: samiran.bag@warwick.ac.uk

Feng Hao
University of Warwick
United Kingdom
Email: feng.hao@warwick.ac.uk

***Abstract*—In this paper we propose an end-to-end (E2E) verifiable online shareholder voting system. Our system allows different voters to have different weights associated with their votes. These weights are dependent upon the number of shares owned by voters in an organization. In our system, the voters cast their votes over the Internet using a personal computing device, say a smart-phone. The voting client interacts with an online voting server which generates encrypted ballots for the chosen candidates with a receipt. Every encrypted ballot comes with a non-interactive zero-knowledge proof to prove the well-formedness of the ciphertext. In addition, the voting system allows the voters to verify their votes are cast as intended through voter initiated auditing. All the encrypted ballots and non-interactive proofs are made available on a publicly readable bulletin board. By checking the receipt against the bulletin board, voters are assured that their votes as recorded as cast. Finally, this e-voting scheme allows everyone including third-party observers to verify all votes are tallied as recorded without involving any tallying authorities. Once the polling concludes, the tallying result is available immediately on the bulletin board with publicly verifiable audit data to allow everyone including third-party observers to verify the tallying integrity of the entire election.**

***Index Terms*—Electronic voting, Shareholder voting, End-to-end verifiability, Zero knowledge proofs, Decisional Diffie-Hellman assumption, Security proof.**

I. Introduction

In a publicly traded company, shareholders have certain rights pertaining to their equity investment. These include the right to vote on certain corporate matters related to the functioning of that company. For example, shareholders may vote to elect the board of directors or approve proposed motions, such as making tie-ups with external organizations or making a new acquisition. Thus, shareholders play a crucial role in the management of a company and execution of certain policies.

Typically companies grant stockholders one vote per share, thus giving those shareholders with a greater investment in the company a greater say in the corporate decision-making. Alternatively, each shareholder may have one vote, regardless of how many shares they own. Shareholders can exercise their voting rights in person at the corporation's annual general meeting (AGM) or remotely through postal voting. However, postal voting can be a tedious process for ordinary voters especially when they need to voter frequently. According to the UK Shareholders Association, only 6% of individual shareholders in the UK actually participate in voting.

An electronic voting system has the potential to make the voting process more convenient by using a touch-screen Direct Recording Electronic (DRE) machine for on-site voting or using a smart phone for remote voting. In spite of the perceived convenience with e-voting systems, they are often susceptible to external attacks. An e-voting system is prone to many forms of malicious attacks, e.g., intrusion, software alteration, eavesdropping etc. These attacks pose a serious threat to the security of any e-voting system, in particular the integrity of the election result. If a hacker can modify the electronic votes or tallies without being detected, the assurance on the tallying integrity will be completely lost. The integrity of democracy will be compromised as well.

To address the security of an e-voting system, researchers have come up with various privacy and verifiability notions. Different definitions of verifiability have been proposed in the literature. These can be be found in [1], [6], [10], [22], [23], [30], [31], [36], [37]. Similarly, there are many notions of privacy in the literature as well. These could be found in [8]–[15], [18], [25], [35].

One of the most essential properties of a secure e-voting system is end-to-end verifiability, which covers the notions of both individual and universal verifiabilities. With the individual verifiability, every voter is able to verify that their vote is cast as intended and recorded as cast. With the universal verifiability, every observer from the public is able to verify that all the votes are tallied as recorded. However, a major difficulty with deploying many E2E verifiable e-voting systems is that they require a set of trustworthy tallying authorities, who are computing and cryptographic experts tasked to perform complex cryptographic decryption and tallying operations. In reality, finding and managing such authorities in an e-voting environment has proved to be rather difficult [2].

In this paper we propose an E2E verifiable e-voting system that does not require any tallying authorities and supports weighted tallying. Our system is called "Share-Holder E-voting" (SHE). The SHE system uses the DRE-ip protocol proposed in [34] as a basic building block, but adds the support for weighted tallying which is required for shareholder voting applications. Our system allows shareholders to vote over the internet with the facility to verify that their votes are cast, recorded and tallied correctly (the same system can also be easily adapted to use for onsite voting, e.g., during the AGM meetings). Further, we will show that our scheme is secure according to the privacy model proposed in [10].

The rest of the paper is organized as follows. In Section III we discuss NIZK proof systems and the DDH assumption. In Section IV, we present our new shareholder voting scheme. In Section V we provide a comprehensive analysis of the scheme. In Section VI we discuss the efficiency of our e-voting scheme. Finally, we conclude the paper in Section VII.

II. Related Work

Research on verifiable e-voting started with the seminal work by Chaum. He proposed a voting system called Votegrity based on visual cryptography [20]. In this scheme, an e-voting machine called Direct Recording Electronic (DRE) issues two sheets of transparencies for every cast vote. Each of the two sheets does not reveal the choice of the voter on its own. But if they are superposed by putting one on top of another, the voter's choice becomes visible. The voter is free to choose one of the sheets as a receipt and the other one is returned to the custom machine which then stores a digital copy of the same before shredding it. Once the polling has finished, all the saved voter receipts are published on the bulletin board, so that the voters can verify that their ballots are not discarded. Chaum's solution highlights the important notion of *end-to-end (E2E) verifibility*, which encompasses the following requirements: individual voters are able to verify their votes are *cast as intended*, and *recorded as cast*, and any public observers (universal verifiers) are able to verify that all votes are *tallied as recorded*.

Ryan proposed Prêt à Voter in [33]. This scheme is a variant of Votegrity. Prêt à Voter uses paper ballots with randomized candidate ordering. The ballot can be split into two halves along a perforation in the middle. The left half contains a list of candidates in random order. The right half has marking spaces corresponding to each candidate on the left. The voter simply puts a mark on the right corresponding to her preferred candidate. Then she can detach the two halves. The right half that contains a mark is then scanned by the voter and the left half is destroyed. The voter can take the right half as her receipt. The right half contains a code that specifies the candidate order for that ballot. This code is used later by the tallying authorities (also called trustees) to decrypt the vote and subsequently compute the tally. The right half does not reveal any information about the voter's choice to anyone other than the trustees. Again, the trustees cannot learn which ballot was issued to a particular voter. Hence, the privacy of the vote is preserved except in the event that the content of the voter's receipt gets decrypted by an over-the-threshold number of colluding trustees.

Other E2E verifiable e-voting systems are MarkPledge [32], Punchscan [26], Scantegrity [19], Scantegrity II [21], scratch & vote [4], STAR-Vote [7], Adder [29], and Helios [5]. These systems use either mix-net [20] or homomorphic encryption [3], but they all involve a set of trustworthy tallying authorities (TAs) to perform the decryption and tallying process in a publicly verifiable way. The TAs should be selected from different parties so they are unlikely to collude. They should also be able to manage the cryptographic keys and perform cryptographic operations independently and competently. However, finding and managing such TAs in practice has proved to be particularly difficult as shown in a campus election using Helios [2].

In 2014, Hao et al. [28] proposed to address the above issue by removing the need for TAs while retaining the E2E verifiability. Their e-voting system is called DRE-i. In this scheme, encrypted ballots are pre-computed before the election. During voting, voters are given an encrypted ballot corresponding to their choices of candidates. This encrypted ballot does not reveal the actual vote and can safely be posted on the publicly available bulletin board. Once all receipts are published, a simple algorithm can extract the final tally from the public data available on the bulletin board. However, decryption of individual ballots is not possible given the information available on the bulletin board. The bulletin board also contains zero knowledge proofs to prove well-formedness for each of the encrypted ballots. This scheme also enables a voter to perform voter-initiated audits before casting the final ballot, so the voter is ensured that their vote is cast as intended.

In the DRE-i system [28], when a voter demands an audit against a particular ballot, the DRE machine is required to reveal the ballot corresponding to the opposite choice of the candidate. Given the pair of ballots it is trivial to find the one that corresponds to the particular choice of the voter. Thus, the voter can check whether or not her ballot corresponded to her exact choice of the candidate. A voter can cast her final ballot after performing a number of audits with the confidence that the system correctly captures her vote cast as intended.

However, the pre-computation approach adopted in DRE-i requires secure storage of the pre-computed balots, or the privacy of the votes may be compromised. To address this issue, in 2016, Shahandashti and Hao proposed an alternative design of the DRE-i system called "DRE with integrity and enhanced privacy" (DRE-ip) [34]. The DRE-ip system provides the same E2E verifiability as DRE-i without requiring TAs. However, DRE-ip, in contrast to DRE-i, does not employ pre-computation of encrypted ballots. Instad, it adopts real-time computation of ballots during the voting process. As a result, DRE-ip is more suitable for polling station voting than DRE-i which requires secure storage of the pre-computed ballots. The system removes the need for TAs by keeping an aggregated form of the random factors in the memory, which will only be published at the end of the election to enable public verification of the tallying integrity without involving any TAs. The system provides a strong privacy guarantee that when the DRE is completely compromised, an attacker will only be able to learn the partial tally at the time of compromise, which is the minimum information leakage.

End-to-end verifiable voting systems like DRE-i and DRE-ip are called "self-enforcing e-voting" [28], [34] as they do not require any tallying authorities. These systems significantly

$Exp_{\mathcal{B}}^{REAL}(\lambda)$	$Exp_{\mathcal{B},S}^{SIM}(\lambda)$
$\sigma \leftarrow K(1^\lambda)$	$(\sigma,\kappa) \leftarrow S_1(1^\lambda)$
$(x,w,\tau) \leftarrow \mathcal{B}_1(\sigma)$	$(x,w,\tau) \leftarrow \mathcal{B}_1(\sigma)$
$\pi \leftarrow P(\sigma,x,w)$	$\pi \leftarrow S_2(x,\kappa)$
Return $\mathcal{B}_2(\pi,\tau)$	Return $\mathcal{B}_2(\pi,\tau)$

simplify the election management by removing the TAs. However, existing SEEV systems only support simple non-ranked voting methods. For example, the basic versions of DRE-i and DRE-ip are designed only for a single candidate election with "Yes" and "No" choices, which can be extended to support multiple candidates as explained in [28] and [34]. However, none of these e-voting schemes support assigning of weights to the votes. In normal elections, every voter has only one vote. However, in special scenarios such as shareholders voting, a voter may have multiple votes (or weights). Hence, the tallying should be computed based on the different weights assigned to voters.

III. Preliminaries

A. NIZK Proofs

An efficient non-interactive zero-knowledge proof [27] for the relation $R \in L$ consists of three PPT algorithms $\Gamma = (K,P,V)$ as described below:

- ☐ K generates the common reference string. It takes as input a λ at returns a common reference string, i.e. $\sigma \leftarrow K(1^\lambda)$.
- ☐ P is the prover algorithm. It takes as input the common reference string σ a statement $x \in L$ and a witness w such that $R(x,w) = True$, and returns a proof π. That is, $\pi \leftarrow P(\sigma,x,w)$.
- ☐ V is the verifier algorithm. It takes as input the common reference string σ, the statement x, the proof π, and returns $v \in \{0,1\}$. That is, $v \leftarrow V(\sigma,x,\pi)$.

Any non-interactive zero knowledge proof system (K,P,V) must satisfy the following properties.

- *Completeness:* The NIZK proof system (K,P,V) is complete if $Pr[\sigma \leftarrow K(1^\lambda); \pi \leftarrow P(\sigma,x,w), V(\sigma,x,\pi) = 1 \wedge R(x,w) = True] = 1$
- *Soundness:* The NIZK proof system (K,P,V) is sound if $Pr[\sigma \leftarrow K(1^\lambda); (x,\pi) \leftarrow \mathcal{D}(\sigma); V(\sigma,x,\pi) = 1 \wedge x \notin L] = 0$
- *Zero Knowledge:* The NIZK proof system is zero knowledge if there exists a simulator $S = (S_1,S_2)$ such that for all probabilistic polynomial time adversaries $\mathcal{D} = (\mathcal{B}_1,\mathcal{B}_2)$, $Adv_{\mathcal{B}}^{NIZK}(\lambda) = \left| Pr\left[Expt_{\mathcal{B}}^{REAL}(\lambda) = 1\right] - Pr\left[Exp_{\mathcal{B},S}^{SIM}(\lambda) = 1\right] \right| \leq negl(\lambda)$

B. Mathematical setup

Let G be a multiplicative group of order q. Also assume that g, and $\tilde{g}$ are two generators of G, such that the relationship between them is unknown to anyone. All operations are modular in G with reference to a prime modulus p. We make an assumption that the Decisional Diffie Hellman assumption holds true in G. We write $x = \log_g A$, if for any $A \in G$, $g^x = A$. Again, for any $A, B \in G$, $DH_g(A,B) = A^{\log_g B} = B^{\log_g A}$.

C. Decisional Diffie Hellman assumption

$Exp_{\mathcal{A}}^{DDH}(\lambda)$
$A \xleftarrow{\$} G$
$B \xleftarrow{\$} G$
$C_0 = DH_g(A,B)$
$C_1 = DH_g(A,B) * g$
$b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}(g,A,B,C_b)$
Return $b = b'$

Let, $Adv_{\mathcal{A}}^{DDH}(\lambda) = \left| Pr[Exp_{\mathcal{A}}^{DDH}(\lambda) = 1] - \frac{1}{2} \right|$. We say that the Decisional Diffie-Hellman assumption holds in the group G, if for any PPT adversary $\mathcal{A}$, $Adv_{\mathcal{A}}^{DDH}(\lambda) \leq negl(\lambda)$.

IV. The scheme

a) **Setup:** The setup function takes as input the security parameter λ, the number of candidates c, and the set of weights $\mathcal{W}$, and returns a group G of prime order p, two mutually unrelated generators g, and $\tilde{g}$ of G. The DDH assumption holds in G. It also returns the set $\mathcal{V}$ of valid votes. $\mathcal{V}$ is the set of $1 \times c$ vectors such than exactly one element of each vector is 1, and the rest are 0s. The Voting Server (VS) uses the setup function to generate these parameters. Apart from these, the VS also holds a $1 \times c$ vector $X = (x_1, x_2, \ldots, x_c)$ which is initialized to 0. The VS also holds the dynamic tally variable $T = (T_1, T_2, \ldots, T_c)$ which is also initialized to 0. The VS is connected to an online bulletin board that functions as a public authenticated channel: the data is readable to everyone but writable only to authorized entities. The VS uses this channel to post its datagrams on the bulletin board. Initially, the VS stores the group G, $g, \tilde{g}$, and p on the bulletin board.

b) **Voting Method:** In this voting scheme, the voter votes remotely through her computer or a mobile device as long as Internet access is available. The election authority issues login credentials to every voter at the time of registration. The voter V_i uses these credentials to authenticate herself to the VS. Once she is logged into the voting system, the system presents the names of all c candidates and allows her to choose one of them. Let us assume she chooses candidate j. Let, $(v_{i1}, v_{i2}, \ldots, v_{ic})$ be such that

$$v_{ik} = \begin{cases} 1 \text{ if } k = j \\ 0 \text{ otherwise} \end{cases}$$

When she chooses candidate j, the VS selects $r_{i1}, r_{i2}, \ldots, r_{ic} \in_R \mathbb{Z}_p$, and computes a ballot $B_i = (B_{i1}, B_{i2}, \ldots, B_{ic})$, where $B_{ik} = \langle b_{ik}, \tilde{b}_{ik} \rangle$, $b_{ik} = g^{r_{ik}} g^{w_i * v_{ik}}$, and $\tilde{b}_{ik} = \tilde{g}^{r_{ik}}$ for all $k \in [1,c]$. Here, $w_i \in \mathcal{W}$ is the publicly known weight associated with the voter V_i. The voting server returns B_i to the voter who can print it on paper or can save it some other way. The voting server also posts the same on the bulletin board. Then the voter is given two choices either to audit this ballot or to confirm it. If the voter chooses to audit her ballot, the voting server returns the the randomnesses $(r_{i1}, r_{i2}, \ldots, r_{ic})$ to the voter, and also posts it on the bulletin board. The voter can print them on paper or can save them locally. The voting server marks B_i and the randomnesses posted on the bulletin board as an audited ballot. The audited ballot is not taken into account while computing the tally. Every time a voter audits a ballot, the voting server needs to start afresh from the beginning, letting the voter make a fresh selection of candidate. A voter can audit her ballot as many times as she wishes to, however, in the end, the voter has to confirm her ballot in order for getting her vote counted in the final tally. When the voter chooses to confirm her ballot, the voting server computes non-interactive zero knowledge proof $\Pi_i = (\pi_{i1}, \pi_{i2} \ldots, \pi_{ic}, \pi_i)$ of well-formedness of the ballot B_i. The voting server appends Π_i on the bulletin board and marks both B_i, and Π_i as the confirmed ballot of voter V_i. For $k \in [1, c]$, π_{ik} proves that given w_i, b_{ik}, and $\tilde{b}_{ik}$, v_{ik} is either 0 or 1. π_i proves that $\sum_{k=1}^{c} v_{ik} = 1$. Thus, as a whole Π_i proves that v_{ik} is 1 for exactly one value of k in $[1, c]$. The VS updates X and T as follows: $x_k = x_k + r_{ik}$ and $T_k = T_k + w_i * v_{ik}$ for all $k \in [1, c]$. The construction of these NIZK proofs can be found in [16], [17], [24], and [34]. While computing the NIZK proofs, the identifier of each ballot should be included in the construction of the proof (i.e., the hash function for the Fiat-Shamir transformation) in order to prevent clash attacks. Once all the n voters have voted, the VS posts T and X on the bulletin board. The tally is correct if the following equations hold

$$\prod_{i=1}^{n} b_{ik} = g^{x_k} g^{T_k} : \forall k \in [1, c]$$

$$\prod_{i=1}^{n} \tilde{b}_{ik} = \tilde{g}^{x_k} : \forall k \in [1, c]$$

V. Analysis

A. Correctness

The following theorem proves that our shareholder voting scheme is correct.

Lemma 1: The shareholder e-voting system outputs the correct tally. That is in the scheme of section IV, $T_k = \sum_{i=1}^{n} w_i * v_{ik}, \forall k \in [1, c]$.

Proof 1: Let us for the sake of argument assume that the VS returns (X', T') instead of (X, T) after the voting ends. Let's assume $X' = (x'_1, x'_2, \ldots, x'_c)$, and $T = (T'_1, T'_2, \ldots, T'_c)$. Hence, these parameters should satisfy the verification equations. Thus, $\prod_{i=1}^{n} b_{ik} = g^{x'_k} g^{T'_k} : \forall k \in [1, c]$ and $\prod_{i=1}^{n} \tilde{b}_{ik} = \tilde{g}^{x'_k} : \forall k \in [1, c]$ Now, we know that $\prod_{i=1}^{n} b_{ik} = g^{x_k} g^{T_k} : \forall k \in [1, c]$, and $\prod_{i=1}^{n} \tilde{b}_{ik} = \tilde{g}^{x_k} : \forall k \in [1, c]$. Hence, $\tilde{g}^{x_k} = \tilde{g}^{x'_k} : \forall k \in [1, c]$, and $g^{x_k} g^{T_k} = g^{x'_k} g^{T'_k} : \forall k \in [1, c]$. This means that $x_k = x'_k$ and $T_k = T'_k : \forall k \in [1, c]$. Hence, the tally is correct.

B. Verifiability

In this section, we show that our SHE e-voting scheme is verifiable. For this purpose, we use the verifiability notion of Smyth et al. [36] and prove that our scheme satisfies the two requirements mentioned in their paper, namely 'individual verifiability' and 'universal verifiability'. In our e-voting scheme, the voter verifies her ballot through auditing as mentioned before. Auditing can ensure verifiability of the ballot only if the voting system satisfies the Individual Verifiability property mentioned in [36], that is if a ballot corresponds to a unique weighted vote.

1) Individual Verifiability: Let us consider the following experiment $Exp_{\mathcal{A}}^{IV}(\lambda)$.

```
$Exp_{\mathcal{A}}^{IV}(\lambda)$
$(G, g, \tilde{g}, V, n_{\max}, p) \leftarrow Setup(1^{\lambda}, c, \mathcal{W})$
$BB = (G, g, \tilde{g}, V, n_{\max}, p, c, \mathcal{W})$
$(v, v', w, w') \leftarrow \mathcal{A}^{BallotGen(,)}(BB)$
if $v = v'$
    return 0
$B \leftarrow BallotGen(v, w)$
$B' \leftarrow BallotGen(v', w')$
return $(B = B') \wedge (B \neq \perp) \wedge (B' \neq \perp)$
```

```
$BallotGen(v, w)$
if $v \notin V || w \notin \mathcal{W}$ return $\perp$
Parse $v$ as $(v_1, v_2, \ldots, v_c)$
$r_1, r_2, \ldots, r_c \in_R \mathbb{Z}_p$
$b_k = g^{r_k} g^{v_k}, \tilde{b}_k = \tilde{g}^{r_k}, \forall k \in [1, c]$
$B_k = \langle b_k, \tilde{b}_k \rangle$
$B = (B_1, B_2, \ldots, B_k)$
```

Let us define $Succ_{\mathcal{A}}^{IV}(\lambda) = Pr[Exp_{\mathcal{A}}^{IV}(\lambda) = 1]$. The scheme is individually verifiable if $Succ_{\mathcal{A}}^{IV}(\lambda) \leq negl(\lambda)$. The following lemma proves that our SHE scheme is individually verifiable.

Lemma 2: Our SHE scheme is individually verifiable.

Proof 2: If our SHE scheme is not individually verifiable then $Succ_{\mathcal{A}}^{IV}(\lambda) > negl(\lambda)$. As such, there exists v, v', w, w' with non-negligible probability, such that $w * v \neq w' * v'$, and $B = B'$, where $B = (B_1, B_2, \ldots, B_c)$, and $B' = (B'_1, B'_2, \ldots, B'_c)$. Here, $B_k = \langle b_k, \tilde{b}_k \rangle$ and $B'_k = \langle b'_k, \tilde{b}'_k \rangle$ for all $k \in [1, c]$. Hence, $b_k = b'_k$, and $\tilde{b}_k = \tilde{b}'_k$ for all $k \in [1, c]$. Let us assume $v = (v_1, v_2, \ldots, v_c)$, and $v' = (v'_1, v'_2, \ldots, v'_c)$. Hence, $b_k = DH_{\tilde{g}}(g, \tilde{b}_k) * g^{wv_k} = DH_{\tilde{g}}(g, \tilde{b}'_k) * g^{wv_k}$ for all $k \in [1, c]$. But, $b_k = b'_k = DH_{\tilde{g}}(g, \tilde{b}'_k) * g^{w'v'_k}$ for all $k \in [1, c]$. Hence, $g^{wv_k} = g^{w'v'_k}$ for all $k \in [1, c]$. Thus, $wv_k = w'v'_k$ for all $k \in [1, c]$. This is a contradiction since, according to our assumption, $wv \neq w'v'$. Hence, the lemma is correct.

2) Universal verifiability: Let us consider another experiment $Exp_{\mathcal{A}}^{UV}(\lambda)$. Here, $VFY\_NIZK(B, \Pi)$ returns 1 if the NIZK proof Π proves the well-formedness of the ballot B, and returns 0 otherwise.

```
$Exp_{\mathcal{A}}^{UV}(\lambda)$
$(G, g, \tilde{g}, V, n_{\max}, p) \leftarrow Setup(1^{\lambda}, c, \mathcal{W})$
$BB = (G, g, \tilde{g}, V, n_{\max}, p, c, \mathcal{W})$
$(tally', X) \leftarrow \mathcal{A}^{BallotGen(,)}(BB)$
$tally \leftarrow Tally(BB)$
return $(tally \neq tally') \wedge Verify(BB, X, tally')$
```

```
$Tally(BB, X)$
extract all ballots $(B_i, \Pi_i) : i \in [1, n]$ in $BB$
Parse $B_i$ as $(B_{i1}, B_{i2}, \ldots, B_{ic})$
Parse $B_{ij}$ as $\langle b_{ij}, \tilde{b}_{ij} \rangle$ for $j \in [1, c]$
if $VFY\_NIZK(B_i, \Pi_i)$ is 0 for any $i \in [1, n]$
return $\perp$
return $(t_1, t_2, \ldots, t_n)$ such that
$t_j = \log_g \left( \frac{\prod_{i=1}^n b_{ij}}{DH_{\tilde{g}}(g, \prod_{i=1}^n \tilde{b}_{ij})} \right) : j \in [1, c]$
```

```
$Verify(BB, X, tally')$
extract all ballots $(B_i, \Pi_i) : i \in [1, n]$ in $BB$
Parse $B_i$ as $(B_{i1}, B_{i2}, \ldots, B_{ic})$
Parse $X$ as $(x_1, x_2, \ldots, x_c)$
Parse $B_{ij}$ as $\langle b_{ij}, \tilde{b}_{ij} \rangle$ for $j \in [1, c]$
if $VFY\_NIZK(B_i, \pi_i)$ is 0 for any $i \in [1, n]$
return 0
Parse $tally'$ as $(t_1, t_2, \ldots, t_n)$
if $\exists j \in [1, c]$, such that
$\prod_{i=1}^n b_{ij} \neq g^{t_j + x_j}$ or $\prod_{i=1}^n \tilde{b}_{ij} \neq \tilde{g}^{x_j}$, then
return 0
return 1
```

```
$BallotGen(v, w)$
if $v \notin V || w \notin \mathcal{W}$ return $\perp$
Parse $v$ as $(v_1, v_2, \ldots, v_c)$
$r_1, r_2, \ldots, r_c \in_R \mathbb{Z}_p$
$b_k = g^{r_k} g^{v_k}, \tilde{b}_k = \tilde{g}^{r_k}, \forall k \in [1, c]$
$B_k = \langle b_k, \tilde{b}_k \rangle$
$B = (B_1, B_2, \ldots, B_k)$
$R = (r_1, r_2, \ldots, r_c)$
return $\langle B, R \rangle$
```

```
$Exp_{\mathcal{A}}^{POB}(\lambda)$
$(G, g, \tilde{g}, V, n_{\max}, p) \leftarrow Setup(1^\lambda, c, \mathcal{W})$
$Tally = Tally' = 0$
$X = (0, 0, \ldots, 0)$
$BB = (G, g, \tilde{g}, V, n_{\max}, p, c, \mathcal{W})$
$b \xleftarrow{\$} \{0, 1\}$
$st \leftarrow \mathcal{A}^{BallotGen(,,)}(BB)$
if $Tally \neq Tally'$ abort
$b \leftarrow \mathcal{A}(st, BB, X, T)$
return $b = b'$
```

```
$BallotGen(v_0, v_1, w)$
if $v_0 \notin V || v_1 \notin V || w \notin \mathcal{W}$ abort
$V_0 = V_0 \bigcup \{v_0\}$, $V_1 = V_1 \bigcup \{v_1\}$
Parse $v_b$ as $(v_{b1}, v_{b2}, \ldots, v_{bc})$
$r_1, r_2, \ldots, r_c \in_R \mathbb{Z}_p$
$b_k = g^{r_k} g^{v_{bk}}, \tilde{b}_k = \tilde{g}^{r_k}, \forall k \in [1, c]$
$B_k = \langle b_k, \tilde{b}_k \rangle$
$B = (B_1, B_2, \ldots, B_k)$
$R = (r_1, r_2, \ldots, r_c)$
$\Pi = NIZK(B, R)$
$X = X + R$
$BB = BB \cup \{B, \pi\}$
$Tally = Tally + w * v_b$
$Tally' = Tally' + w * v_{1-b}$
```

Fig. 1. The security experiment of vote privacy

Lemma 3: Our SHE e-voting scheme is universally verifiable.

Proof 3: Let us first assume that our scheme is not universally verifiable. Then the adversary can generate a bulletin board BB a vector X, and a tally $tally'$, such that $Verify(BB, X, tally')$ is true but $tally'$ is not the correct tally. Let us assume that the set of ballots are (B_i, Π_i), where $B_i = (B_{i1}, B_{i2}, \ldots, B_{ic})$, and $B_{ij} = \langle b_{ij}, \tilde{b}_{ij} \rangle$. If all the NIZK proofs Π_i verify then the correct tally will be $(T_1, T_2, \ldots, T_c)$, where $\prod_{i=1}^n b_{ij} = g^{T_j} * DH_{\tilde{g}}(g, \prod_{i=1}^n \tilde{b}_{ij})$ for all $j \in [1, c]$. Let us assume that $tally' = (t_1, t_2, \ldots, t_c)$, and $X = (x_1, x_2, \ldots, x_c)$. Then if the verification holds then the following equations must hold.

$$\prod_{i=1}^{n} b_{ij} = g^{t_j} g^{x_j}, \forall j \in [1, c]$$

$$\prod_{i=1}^{n} \tilde{b}_{ij} = \tilde{g}^{x_j}, \forall j \in [1, c]$$

This means that $\prod_{i=1}^n b_{ij} = g^{t_j} * DH_{\tilde{g}}(g, \prod_{i=1}^n \tilde{b}_{ij}), \forall j \in [1, c]$. Hence, $T_j = t_j$ for all $j \in [1, c]$. Hence, our scheme is universally verifiable.

C. Security Property

a) **The Formal Model:** Here, we show that our shareholder e-voting system protects the privacy of the voters. In a voting protocol, the adversary should only learn the public tally and whatever she can infer from the tally. Any probabilistic polynomial time adversary who has colluded with an arbitrary number of voters cannot learn any extra information which she cannot compute from the tally itself. We define vote privacy with the help of the privacy model proposed by Benaloh et al. in [9]. However, since our e-voting scheme does not require any tallying authority, we have modified the definition of Benaloh et al. to suite our needs. For this purpose, we introduce the following security experiment $Exp_{\mathcal{A}}^{POB}(\lambda)$ as described in Figure 1. In this experiment, the challenger first uses the setup function to generate the public parameters of the protocol. The challenger initializes two $1 \times c$ vectors $Tally$ and $Tally'$ to 0. The public parameters are posted on the bulletin board. The adversary is given access to the bulletin board. The adversary can use the $BallotGen()$ oracle to caste votes. The $BallotGen$ oracle has 3 inputs namely v_0, v_1 and w. They come from the adversary. v_0 and v_1 are votes and w is the value of weight corresponding to them. The challenger chooses a random bit $b \in \{0, 1\}$. Depending on the value of the challenge bit b, the oracle chooses either v_0 or v_1 and generates a ballot. The oracle $BallotGen$ posts on the bulletin board, the ballot and the NIZK proofs of well-formedness of the ballot. Then the oracle updates the two variables $Tally$ and $Tally'$ as follows: $Tally = Tally + v_0$ and $Tally' = Tally' + v_1$. Once, the adversary has cast enough ballots, the challenger checks whether $Tally = Tally'$ or not. If they are different, the challenger aborts and the adversary loses. If they are same, the adversary is given access to the two variables X and T. Now, the adversary returns a bit. If this bit is same as b, the adversary wins the game and she loses otherwise. In the description of the oracle $BallotGen(\cdot)$ in Figure 1, the function $NIZK(B, R)$ generates the NIZK proof Π of well-formedness of the ballot B, given the randomness R.

We define the advantage of the adversary $\mathcal{A}$ against the security experiment $Exp_{\mathcal{A}}^{POB}(\lambda)$ as follows

$$Adv_{\mathcal{A}}^{POB}(\lambda) = \left| Pr[Exp_{\mathcal{A}}^{POB}(\lambda) = 1] - \frac{1}{2} \right|$$

Definition 1: Let us consider the following security experiment $Exp_{\mathcal{D}}^{DDH}(\lambda)$.

```
$Exp_{\mathcal{D}}^{DDH}(\lambda)$
$g \xleftarrow{\$} G$
$A \xleftarrow{\$} G,\ B \xleftarrow{\$} G$
$C_0 = DH_g(A, B)$
$C_1 \xleftarrow{\$} G$
$b \xleftarrow{\$} \{0, 1\}$
$b' = \mathcal{D}(g, A, B, C_b)$
return $(b = b')$
```

We denote the advantage of any adversary $\mathcal{D}$ against the security experiment as $Adv_{\mathcal{D}}^{DDH}(\lambda)$. We define it as follows

$$Adv_{\mathcal{D}}^{DDH}(\lambda) = \left| Pr[Exp_{\mathcal{D}}^{DDH}(\lambda) = 1] - \frac{1}{2} \right|$$

We say that the Decisional Diffie-Hellman assumption holds in a group G, if for all PPT adversary $\mathcal{D}$, $Adv_{\mathcal{A}}^{DDH}(\lambda) \leq negl(\lambda)$.

Definition 2:

Let us consider the following security experiment $Exp_{\mathcal{F}}^{DDH2}(\lambda)$. The advantage of an adversary $\mathcal{F}$ against the security experiment $Exp_{\mathcal{F}}^{DDH2}(\lambda)$ is denoted as $Adv_{\mathcal{F}}^{DDH2}(\lambda)$. $Adv_{\mathcal{F}}^{DDH2}(\lambda) = \left| Pr[Exp_{\mathcal{F}}^{DDH2}(\lambda) = 1] - \frac{1}{2} \right|$

```
$Exp_{\mathcal{F}}^{DDH2}(\lambda)$
$g \xleftarrow{\$} G$
$\tilde{g} \xleftarrow{\$} G$
$x \xleftarrow{\$} \mathbb{Z}_p$
$R \xleftarrow{\$} G$
$C_0 = (g^x, \tilde{g}^x)$
$C_1 = (R, \tilde{g}^x)$
$b \xleftarrow{\$} \{0, 1\}$
$b' \leftarrow \mathcal{F}(g, \tilde{g}, C_b)$
return $b = b'$
```

```
$Exp_{\mathcal{H}}^{DDH4}(\lambda)$
$g \xleftarrow{\$} G$
$\tilde{g} \xleftarrow{\$} G$
$x \xleftarrow{\$} \mathbb{Z}_p$
$R \xleftarrow{\$} G$
$C_0 = (g^x * g, \tilde{g}^x)$
$C_1 = (R, \tilde{g}^x)$
$b \xleftarrow{\$} \{0, 1\}$
$b' \leftarrow \mathcal{G}(g, \tilde{g}, C_b)$
return $b = b'$
```

```
$Exp_{\mathcal{G}}^{DDH5}(\lambda)$
$g \xleftarrow{\$} G$
$\tilde{g} \xleftarrow{\$} G$
$x \xleftarrow{\$} \mathbb{Z}_p$
$R \xleftarrow{\$} G$
$C_0 = (g^x, \tilde{g}^x)$
$C_1 = (g^x * g, \tilde{g}^x)$
$b \xleftarrow{\$} \{0, 1\}$
$b' \leftarrow \mathcal{G}(g, \tilde{g}, C_b)$
return $b = b'$
```

Lemma 4: $Adv_{\mathcal{F}}^{DDH2}(\lambda) \leq Adv_{\mathcal{D}}^{DDH}(\lambda)$.

Proof 4: We show that if there exists an adversary $\mathcal{F}$ against the security experiment $Exp_{\mathcal{F}}^{DDH2}(\lambda)$, then it could be used in the construction of another adversary $\mathcal{D}$ against the security experiment $Exp_{\mathcal{D}}^{DDH}(\lambda)$. $\mathcal{D}$ works as follows: it receives as inputs: g, A, B and a challenge C_b. $\mathcal{D}$ assigns $\tilde{g} = A$, $C_b' = (B, C_b)$. Note that if $C_b = DH_g(A, B)$, then $C_b' = (g^{\log_g B}, \tilde{g}^{\log_g B})$. Else, $C_b' = (R, \tilde{g}^y)$, where $R = B \in_R G$, and $y = \log_A C_b \in_R \mathbb{Z}_p$. If $\mathcal{F}$ can distinguish between these two cases, $\mathcal{D}$ will be able to find b. Hence, the lemma holds.

Definition 3: Let us consider another security experiment $Exp_{\mathcal{H}}^{DDH4}(\lambda)$. The advantage of an adversary $\mathcal{G}$ against this security experiment is defined as

$$Adv_{\mathcal{H}}^{DDH4}(\lambda) = \left| Pr[Exp_{\mathcal{H}}^{DDH4}(\lambda) = 1] - \frac{1}{2} \right|$$

Lemma 5: $Adv_{\mathcal{H}}^{DDH4}(\lambda) \leq Adv_{\mathcal{D}}^{DDH}(\lambda)$.

Proof 5: We show that if there exists an adversary $\mathcal{H}$ against the security experiment $Exp_{\mathcal{H}}^{DDH2}(\lambda)$, then it could be used in the construction of another adversary $\mathcal{D}$ against the security experiment $Exp_{\mathcal{D}}^{DDH}(\lambda)$. $\mathcal{D}$ works as follows: it receives as inputs: g, A, B and a challenge C_b. $\mathcal{D}$ assigns $\tilde{g} = A$, $C_b' = (B * g, C_b)$. Note that if $C_b = DH_g(A, B)$, then $C_b' = (g^{\log_g B} * g, \tilde{g}^{\log_g B})$. Else, $C_b' = (R, \tilde{g}^y)$, where $R = B * g \in_R G$, and $y = \log_A C_b \in_R \mathbb{Z}_p$. If $\mathcal{H}$ can distinguish between these two cases, $\mathcal{D}$ will be able to find b. Hence, the lemma holds.

From Lemma 4 and 5, we can state the following corollary.

Corollary 1: $Adv_{\mathcal{G}}^{DDH5}(\lambda) \leq Adv_{\mathcal{D}}^{DDH}(\lambda)$.

Let us define yet another security experiment.

Definition 4: Let us consider the following security experiment $Exp_{\mathcal{E}}^{DDH1}(\lambda)$.

```
$Exp_{\mathcal{E}}^{DDH1}(\lambda)$
$g \xleftarrow{\$} G$
$\tilde{g} \xleftarrow{\$} G$
$BB = BB \bigcup \{(g, \tilde{g})\}$
$b \xleftarrow{\$} \{0, 1\}$
$b' \leftarrow \mathcal{E}^{\mathcal{O}(,)}(BB)$
return $b = b'$
```

```
$\mathcal{O}(w_0, w_1)$
$w = w_b$
$x \xleftarrow{\$} \mathbb{Z}_p$
$B = (g^x g^w, \tilde{g}^x)$
$BB = BB \bigcup \{B, w_0, w_1\}$
```

The advantage of $\mathcal{E}$ against the security experiment $Exp_{\mathcal{E}}^{DDH1}(\lambda)$ is given by

$$Adv_{\mathcal{E}}^{DDH1}(\lambda) = \left| Pr[Exp_{\mathcal{E}}^{DDH1}(\lambda) = 1] - \frac{1}{2} \right|$$

Lemma 6: $Adv_{\mathcal{E}}^{DDH1}(\lambda) \leq Adv_{\mathcal{G}}^{DDH5}(\lambda)$.

Proof 6: We show that if there exists an adversary $\mathcal{E}$ against the security experiment $Exp_{\mathcal{E}}^{DDH1}(\lambda)$, then it could be used to construct another adversary $\mathcal{G}$ against the security experiment $Exp_{\mathcal{G}}^{DDH5}(\lambda)$. $\mathcal{G}$ receives as inputs the following items: $g, \tilde{g}$, and the challenge $C_b \in_R \{C_0, C_1\}$, where $C_0 = (g^x, \tilde{g}^x)$, and $C_1 = (g^x * g, \tilde{g}^x)$. $\mathcal{G}$ initializes BB as $(g, \tilde{g})$. $\mathcal{G}$ answers all oracle queries of $\mathcal{E}$. When $\mathcal{G}$ makes a call $\mathcal{O}(w_1, w_2)$, for some $w_1, w_2 \in \mathbb{Z}_p$, then $\mathcal{G}$ selects random $y \in_R \mathbb{Z}_p$, and computes $D' = g^y D^{w_1 - w_0} * g^{w_0}$, and $E' = \tilde{g}^y * E^{w_1 - w_0}$. Here, $(D, E) = C_b$. $\mathcal{E}$ updates BB as $BB = BB \bigcup \{(D, E), w_0, w_1\}$. Note that if $b = 0$, then $D = g^{y + (w_1 - w_0) * x} g^{w_0}$, and if $b = 1$, then $D = g^{y + (w_1 - w_0) * x} g^{w_1}$. Here $E = \tilde{g}^{y + (w_1 - w_0) * x}$. Now, let us denote $y + (w_1 - w_0) * x$ by x'. Since y is uniformly random in $\mathbb{Z}_p$, so is x'. Hence, $(D, E) = (g^{x'} g^{w_b}, \tilde{g}^{x'})$. Now, $\mathcal{G}$ returns what $\mathcal{E}$ returns. It is easy to see that the result holds.

Lemma 7: $Adv_{\mathcal{A}}^{POB}(\lambda) \leq poly(\lambda) * \left(Adv_{\mathcal{D}}^{DDH}(\lambda) + Adv_{\mathcal{B}}^{NIZK}(\lambda)\right)$.

Proof 7: We show that if there exists an adversary $\mathcal{E}$ against the security experiment $Exp_{\mathcal{A}}^{POB}(\lambda)$, it could be used to construct an adversary $\mathcal{E}$ against the security experiment $Exp_{\mathcal{E}}^{DDH1}(\lambda)$. $\mathcal{E}$ works as follows: It creates a bulletin board BB' and stores g and $\tilde{g}$ on it. When $\mathcal{E}$ is invoked with the input BB, $\mathcal{E}$ uses the $Setup()$ function to generate the set of votes $\mathcal{V}$, the maximum number of vote $n_{\max}$. The adversary $\mathcal{E}$ initializes the tally variables $Tally$ and $Tally'$ to 0. $\mathcal{E}$ posts $(G, g, \tilde{g}, \mathcal{V}, p, c, \mathcal{W})$ on the bulletin board BB'. Note that $n_{\max} \in poly(\lambda)$. $\mathcal{E}$ selects random $n_{\max} \in poly(\lambda)$. $\mathcal{E}$ answers all queries to $BallotGen(\cdot)$ oracle made by $\mathcal{A}$. Let the i'th query to $BallotGen(\cdot)$ be of the form (v_{0i}, v_{1i}, w_i). $\mathcal{E}$ first checks whether or not $v_{ji} \in \mathcal{V} : j \in [0,1]$ and $w_i \in \mathcal{W}$. If any of the inputs is incorrect, then $\mathcal{E}$ aborts and return a random bit. Else, $\mathcal{E}$ parses v_{0i} and v_{1i} as follows: $v_{0i} = (v_{0i1}, v_{0i2}, \ldots, v_{0ic})$, and $v_{1i} = (v_{1i1}, v_{1i2}, \ldots, v_{1ic})$. $\mathcal{E}$ makes oracle query to $\mathcal{O}(w_i * v_{0i\alpha}, w_i * v_{1i\alpha})$ for all $\alpha \in [1, c]$. The oracle returns $\langle B, B' \rangle$. $\mathcal{E}$ assigns $b_{i\alpha} = B, \tilde{b}_{i\alpha} = B'$. $\mathcal{E}$ assigns $B_{ik} = \langle b_{ik}, \tilde{b}_{ik} \rangle$, and $B_i = (B_1, B_2, \ldots, B_c)$. $\mathcal{E}$ also computes simulated NIZK proofs of well-formedness of B_i. Let this proof be denoted by Π_i. $\mathcal{E}$ posts B_i, Π on the bulletin board BB'. If $\mathcal{A}$ makes n queries to $BallotGen(\cdot)$, then $\mathcal{E}$ makes nc queries. Since, $c = O(1)$, the total number of queries made by $\mathcal{E}$ will be in $poly(\lambda)$. If $\mathcal{A}$ stops before making $n_{\max}$ queries to $BallotGen(\cdot)$, then $\mathcal{E}$ returns a random bit. If $\mathcal{A}$ makes $n_{\max}$'th query to $BallotGen(\cdot)$, then $\mathcal{E}$ checks whether or not $\sum_{i=1}^{n_{\max}} v_{0i} = \sum_{i=1}^{n_{\max}} v_{1i}$. If they are unequal $\mathcal{E}$ aborts and returns a random bit. Else, it selects $R = (r_1, r_2, \ldots, r_c) \in_R \mathbb{Z}_p^c$, and computes $b_{n_{\max}j} = g^{r_j} g^{t'_j} / (\prod_{k=1}^{n_{\max}-1} b_{kj})$, and $\tilde{b}_{n_{\max}j} = \tilde{g}^{r_j} / (\prod_{k=1}^{n_{\max}-1} \tilde{b}_{kj})$ for all $j \in [1, c]$. Here, $\sum_{i=1}^{n_{\max}} v_{0i} = (t_1, t_2, \ldots, t_c)$. Now $\mathcal{A}$ returns a bit. $\mathcal{E}$ returns the same bit.

Now, we calculate the success probability of $\mathcal{E}$. $Pr[Exp_{\mathcal{E}}^{DDH1}(\lambda) = 1] = Pr[\mathcal{E}$ Aborts $] * Pr[Exp_{\mathcal{E}}^{DDH1}(\lambda) = 1 \big| \mathcal{E}$ Aborts $] + Pr[\mathcal{E}$ Does not Abort $] * Pr[Exp_{\mathcal{E}}^{DDH1}(\lambda) = 1 \big| \mathcal{E}$ Does not Abort $]$. If $\mathcal{E}$ aborts, then $\mathcal{E}$ returns a random bit. Hence, $Pr[Exp_{\mathcal{E}}^{DDH1}(\lambda) = 1 \big| \mathcal{E}$ Aborts $] = \frac{1}{2}$. $\mathcal{E}$ aborts when the total number of queries made by $\mathcal{A}$ to $BallotGen(\cdot)$ does not equal $n_{\max}$. Since, the total number of queries made by $\mathcal{A}$ is $poly(\lambda)$, $n_{\max} \in_R poly(\lambda)$, $Pr[\mathcal{E}$ Does not Abort $] = \frac{1}{poly(\lambda)}$. Thus, $Pr[Exp_{\mathcal{E}}^{DDH1}(\lambda) = 1] = \left(1 - \frac{1}{poly(\lambda)}\right) * \frac{1}{2} + \frac{1}{poly(\lambda)} * Pr[Exp_{\mathcal{A}}^{POB}(\lambda) = 1]$ We know that $Adv_{\mathcal{A}}^{POB}(\lambda) = Pr[Exp_{\mathcal{A}}^{POB}(\lambda) = 1] - \frac{1}{2} + Adv_{\mathcal{B}}^{NIZK}(\lambda)$. Hence, $Pr[Exp_{\mathcal{E}}^{DDH1}(\lambda) = 1] - \frac{1}{2} \geq \frac{Adv_{\mathcal{A}}^{POB}(\lambda) - Adv_{\mathcal{B}}^{NIZK}(\lambda)}{poly(\lambda)}$. Thus,

$$Adv_{\mathcal{A}}^{POB}(\lambda) \leq poly(\lambda) * \left(Adv_{\mathcal{E}}^{DDH1}(\lambda) + Adv_{\mathcal{B}}^{NIZK}(\lambda)\right)$$

Using Lemma 1 and 6, we can write,

$$Adv_{\mathcal{A}}^{POB}(\lambda) \leq poly(\lambda) * \left(Adv_{\mathcal{D}}^{DDH}(\lambda) + Adv_{\mathcal{B}}^{NIZK}(\lambda)\right)$$

Thus, our shareholder voting system is secure under Benaloh's model if the Decisional Diffie-Hellman assumption holds in G, and if the NIZK proof systems employed in the scheme are indeed zero knowledge.

VI. Efficiency

In this section, we discuss the efficiency of our scheme. Since exponentiation is the costliest operation in our scheme, we measure the efficiency in terms of the number of exponentiations performed by the VS and the verifier. In SHE, the VS needs to perform $2c$ exponentiations in order to generate the ballot for any user. If on average a voter audits β ballots before confirming her ballot ($\beta + 1$ ballots totally), the VS will need to perform $2c(\beta + 1)$ exponentiations. Each confirmed ballot corresponds to a NIZK proof. The VS needs to perform $6c + 2$ exponentiations for generating the NIZK proof Π_i per voter. Again, each ballot is of size $2c$. The audited ballots contain additional randomness vectors, each of size c, making the total size of audited ballots equal to $3c$. The NIZK proof associated with a confirmed ballot is of size $8c + 4$. Thus the total communication cost per voter is $c(3\beta + 8) + 4$.

Let us now analyze the verification cost. Each of the audited ballots can be verified by performing c exponentiations. The NIZK proof Π_i that correspond to each confirmed ballot, can be verified by performing $8c + 4$ exponentiations. Thus, on an average the verifier needs to perform $c(8 + \beta) + 4$ exponentiations. Table I shows a breakdown of the computation and communication overhead on the VS per voter in our SHE e-voting scheme.

	Computation Cost			Communication Cost			
Type	Ballot	NIZKP	Total	Ballot	Secret Key	NIZKP	Total
Audited	$2c$	–	$2c$	$2c$	c	–	$3c$
Confirmed	$2c$	$6c + 2$	$8c + 2$	$2c$	-	$8c + 4$	$10c + 4$

TABLE I
THE COMPUTATION AND COMMUNICATION COST FOR THE PROPOSED SHE E-VOTING SCHEME.

VII. Conclusion

In this paper, we proposed SHE, an e-voting scheme that allows shareholders of a company to vote remotely (or onsite) on certain corporate matters. Our SHE scheme is end-to-end verifiable according to the notion provided in the literature. In addition, our scheme provides strong guarantees of the voter privacy. When the voting server is completely compromised, what the attacker can learn is limited to the partial tally at the time of the compromise, which is minimum information leakage. We have also shown that our scheme incurs reasonable computational and communication load on the election management system which makes it suitable for real-world deployment.

Acknowledgement

This work is supported by the Royal Society grant, ICA/R1/180226.

References

[1] Demos-2: Scalable e2e verifiable elections without random oracles. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 352–363. ACM, 10 2015.

[2] Ben Adida. Helios: Web-based open-audit voting. In *Proceedings of the 17th Conference on Security Symposium*, SS'08, pages 335–348, Berkeley, CA, USA, 2008. USENIX Association.

[3] Ben Adida, Olivier De Marneffe, Olivier Pereira, Jean-Jacques Quisquater, et al. Electing a university president using open-audit voting: Analysis of real-world use of helios. *EVT/WOTE*, 9(10), 2009.

[4] Ben Adida and Ronald L. Rivest. Scratch & vote: Self-contained paper-based cryptographic voting. In *Proceedings of the 5th ACM Workshop on Privacy in Electronic Society*, WPES '06, pages 29–40, New York, NY, USA, 2006. ACM.

[5] Syed Taha Ali and Judy Murray. An overview of end-to-end verifiable voting systems. *Real-world electronic voting: Design, analysis and deployment*, pages 171–218, 2016.

[6] Carsten Baum, Ivan Damgård, and Claudio Orlandi. Publicly auditable secure multi-party computation. In Michel Abdalla and Roberto De Prisco, editors, *Security and Cryptography for Networks*, pages 175–196, Cham, 2014. Springer International Publishing.

[7] Susan Bell, Josh Benaloh, Michael D. Byrne, Dana Debeauvoir, Bryce Eakin, Philip Kortum, Neal McBurnett, Olivier Pereira, Philip B. Stark, Dan S. Wallach, Gail Fisher, Julian Montoya, Michelle Parker, and Michael Winn. Star-vote: A secure, transparent, auditable, and reliable voting system. In *2013 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 13)*, Washington, D.C., August 2013. USENIX Association.

[8] Josh Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing*, STOC '94, pages 544–553, New York, NY, USA, 1994. ACM.

[9] Josh C Benaloh and Moti Yung. Distributing the power of a government to enhance the privacy of voters. In *Proceedings of the Fifth Annual ACM Symposium on Principles of Distributed Computing*, PODC '86, pages 52–62, New York, NY, USA, 1986. ACM.

[10] Josh Daniel Cohen Benaloh. *Verifiable Secret-ballot Elections*. PhD thesis, New Haven, CT, USA, 1987. AAI8809191.

[11] D. Bernhard, V. Cortier, D. Galindo, O. Pereira, and B. Warinschi. Sok: A comprehensive analysis of game-based ballot privacy definitions. In *2015 IEEE Symposium on Security and Privacy*, pages 499–516, May 2015.

[12] David Bernhard, Véronique Cortier, Olivier Pereira, Ben Smyth, and Bogdan Warinschi. Adapting helios for provable ballot privacy. In Vijay Atluri and Claudia Diaz, editors, *Computer Security – ESORICS 2011*, pages 335–354, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[13] David Bernhard, Véronique Cortier, Olivier Pereira, and Bogdan Warinschi. Measuring vote privacy, revisited. pages 941–952, 10 2012.

[14] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, pages 626–643, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[15] David Bernhard, Olivier Pereira, and Bogdan Warinschi. On necessary and sufficient conditions for private ballot submission. *IACR Cryptology ePrint Archive*, 2012:236, 2012.

[16] Stefan A Brands. *Rethinking public key infrastructures and digital certificates: building in privacy*. Mit Press, 2000.

[17] Jan Camenisch and Markus Stadler. Proof systems for general statements about discrete logarithms. Technical report, ETH Zurich, Technical Report No. 260, 1997.

[18] Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. Verifiable elections that scale for free. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *Public-Key Cryptography – PKC 2013*, pages 479–496, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[19] D. Chaum, A. Essex, R. Carback, J. Clark, S. Popoveniuc, A. Sherman, and P. Vora. Scantegrity: End-to-end voter-verifiable optical- scan voting. *IEEE Security & Privacy*, 6(3):40–46, May 2008.

[20] David Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE security & privacy*, 2(1):38–47, 2004.

[21] David Chaum, Richard Carback, Jeremy Clark, Aleksander Essex, Stefan Popoveniuc, Ronald L. Rivest, Peter Y. A. Ryan, Emily Shen, and Alan T. Sherman. Scantegrity ii: End-to-end verifiability for optical scan election systems using invisible ink confirmation codes. In *Proceedings of the Conference on Electronic Voting Technology*, EVT'08, pages 14:1–14:13, Berkeley, CA, USA, 2008. USENIX Association.

[22] Benoît Chevallier-Mames, Pierre-Alain Fouque, David Pointcheval, Julien Stern, and Jacques Traoré. *On Some Incompatible Properties of Voting Schemes*, pages 191–199. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[23] Véronique Cortier, David Galindo, Stéphane Glondu, and Malika Izabachène. Election verifiability for helios under weaker trust assumptions. In *19th European Symposium on Research in Computer Security - Volume 8713*, ESORICS 2014, pages 327–344, New York, NY, USA, 2014. Springer-Verlag New York, Inc.

[24] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Annual International Cryptology Conference*, pages 174–187. Springer, 1994.

[25] Édouard Cuvelier, Olivier Pereira, and Thomas Peters. Election verifiability or ballot privacy: Do we need to choose? In Jason Crampton, Sushil Jajodia, and Keith Mayes, editors, *Computer Security – ESORICS 2013*, pages 481–498, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[26] Kevin Fisher, Richard Carback, and Alan T. Sherman. Punchscan: Introduction and system definition of a high-integrity election system. In *Workshop on Trustworthy Election. 2006*, 2006.

[27] Jens Groth. Simulation-sound nizk proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology – ASIACRYPT 2006*, pages 444–459, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[28] Feng Hao, Matthew N. Kreeger, Brian Randell, Dylan Clarke, Siamak F. Shahandashti, and Peter Hyun-Jeen Lee. Every vote counts: Ensuring integrity in large-scale electronic voting. In *2014 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 14)*, San Diego, CA, August 2014. USENIX Association.

[29] A. Kiayias, M. Korman, and D. Walluck. An internet voting system supporting user privacy. In *2006 22nd Annual Computer Security Applications Conference (ACSAC'06)*, pages 165–174, Dec 2006.

[30] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. End-to-end verifiable elections in the standard model. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, pages 468–498, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[31] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Accountability: Definition and relationship to verifiability. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 526–535, New York, NY, USA, 2010. ACM.

[32] C. Andrew Neff. Practical high certainty intent verification for encrypted votes, 2004.

[33] P. Y. A. Ryan, D. Bismark, J. Heather, S. Schneider, and Z. Xia. Prêt á voter voter: a voter-verifiable voting system. *IEEE Transactions on Information Forensics and Security*, 4(4):662–673, Dec 2009.

[34] Siamak F Shahandashti and Feng Hao. Dre-ip: a verifiable e-voting scheme without tallying authorities. In *European Symposium on Research in Computer Security*, pages 223–240. Springer, 2016.

[35] Ben Smyth and David Bernhard. Ballot secrecy and ballot independence coincide. In Jason Crampton, Sushil Jajodia, and Keith Mayes, editors, *Computer Security – ESORICS 2013*, pages 463–480, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[36] Ben Smyth, Steven Frink, and Michael R. Clarkson. Computational election verifiability: Definitions and an analysis of helios and jcj. *IACR Cryptology ePrint Archive*, 2015:233, 2015.

[37] Alan Szepieniec and Bart Preneel. New techniques for electronic voting. *USENIX Journal of Election Technology and Systems (JETS)*, 2015.