Assurable, Transparent, and Mutual Restraining E-voting Involving Multiple Conflicting Parties

Xukai Zou*, Huian Li*, Yan Sui*, Wei Peng* and Feng Li†
*Department of Computer and Information Science
Indiana University-Purdue University Indianapolis, Indiana 46202, USA
Email: {xkzou, huili, ysui, pengw}@cs.iupui.edu
†Department of Computer and Information Technology
Indiana University-Purdue University Indianapolis, Indiana 46202, USA
Email: fengli@iupui.edu

***Abstract*—E-voting techniques and systems have not been widely accepted and deployed by society due to various concerns and problems. One particular issue associated with many existing e-voting techniques is the lack of transparency, leading to the failure to deliver voter assurance. In this work, we propose an assurable, transparent, and mutual restraining e-voting protocol that exploits the existing two-party political dynamics in the US. The proposed e-voting protocol consists of three original technical contributions – universal verifiable voting vector, forward and backward mutual lock voting, and in-process check and enforcement – that, in combination, resolves the apparent conflicts in voting such as anonymity vs. accountability and privacy vs. verifiability. Especially, the trust is split equally among tallying authorities who have conflicting interests and will technically restrain each other. The voting and tallying processes are transparent to voters and any third party, which allow any voter to verify that his vote is indeed counted and also allow any third party to audit the tally.**

***Index Terms*—E-voting, Verifiability, Anonymity, Privacy, Assurance, Transparency, Mutual restraining voting**

I. Introduction

Voting is the pillar of modern democracies, and a voting system is the central piece for people to execute their rights. Traditional voting, however, suffers from both low efficiency and unintentional errors. The event surrounding the 2000 US presidential election demonstrated the shortcomings of punch-cards and other antiquated voting systems. This drives the government to deploy more advanced voting systems.

Electronic-voting (e-voting) has been an active research topic with many advantages over traditional voting, but poses its own unique challenges. For example, if a discrepancy is found in tally, votes need to be recounted and the source of the discrepancy needs to be identified. The recounting and investigation should nevertheless preserve votes' anonymity and voters' privacy. Other voting requirements, such as verifiability and receipt-freeness, make the problem even more challenging due to their inherently contradicting nature [16], [26].

Several e-voting solutions [39], [28], [37], [53] have been proposed. Some suggest keeping non-electronic parallels of electronic votes, or saving copies of votes in portable storage devices. These solutions either fail to identify sources of discrepancy or are susceptible to vote selling/coercion. Most solutions are based on cryptographic techniques, such as secret sharing, mix-net, and blind signature. These solutions are often *opaque*: Besides casting their votes, voters do not directly participate in collecting and tallying votes. This raises concerns over the trustworthiness and transparency of the voting process. In addition, these solutions sometimes entrust the fairness of the voting process to the impartiality of authorities. Voting under multiple conflicts-of-interest parties is not addressed by these solutions.

In this work, we propose a robust, assurable, transparent, and mutual restraining e-voting protocol that exploits conflicts of interest in multiple tallying authorities, such as the two-party political system in the US. The new protocol consists of a few novel techniques – universal verifiable voting vector, forward and backward mutual lock voting, and proven in-process check and enforcement – that, in combination, resolves the apparent conflicts such as anonymity vs. accountability and privacy vs. verifiability. These new techniques function as below:

- The tallied voting vector allows both individual and universal verification without jeopardizing voters' privacy and anonymity. A voter is assured that his vote is counted, thus achieving voter assurance.
- Mutual lock voting ensures that a voter can cast one and only one vote, thus preventing multiple and fake voting.
- The in-process check and enforcement mechanism verifies any voter's vote in an anonymous manner, identifies any invalid vote caused by misconduct, and ensures that voters follow the voting protocol without deviation. The trust is split equally among tallying authorities who have conflicting interests and will check and restrain each other.
- Both the vote-casting process using simplified (N,N) secret sharing, and the vote-tallying process based on incremental aggregation (i.e., simple integer addition), are transparent (viewable) to voters. This, in turn, will further win trust and confidence of voters.

To the best of our knowledge, this is the first fully transparent e-voting protocol and is able to deliver voter assurance. The most groundbreaking feature of our e-voting protocol, different from all existing ones, is the separation and guarantee of two distinct voter assurances: 1) *vote-casting assurance* on *secret ballots* – any voter is assured that the vote-casting is indeed completed (i.e., the secret ballot is confirmatively cast and viewably aggregated), thanks to the openness of secret ballots and incremental aggregations, and 2) *vote-tallying assurance* – any voter is assured that their vote is viewably counted in the final tally, thanks to the seamless transition from secret ballots having no information to public votes having complete (but anonymous) information offered by the simplified (N,N) secret sharing scheme.

We understand that our protocol may not scale to very large number of voters. However, a typical voting system [57] has a hierarchical tree structure where vote totals from precincts (say, towns or counties) are sent to an upper level for consolidating. In such a voting structure, each precinct can apply our protocol on its own efficiently because of the relatively low population. Consequently, scalability is not a significant issue. However, we will address this in our future work, so that the protocol can apply to more general cases.

The rest of the paper is organized as follows. Related works are reviewed in Section II. Section III introduces models/assumptions and building blocks. The technical details of the protocol are presented in Section IV, followed by proof and analysis of the protocol in Section V. Simulation results are shown in Section VI. Section VII concludes our paper.

For convenience, notations used in the following sections are summarized in Table I for reference.

TABLE I: Notations

$V; V_1, \cdots, V_N$	Voters. $1, \cdots, N$ are voters' indices
N	Number of voters
M	Number of candidates
$C; C_1, C_2$	Collectors
$\mathbf{Z_A}, g$	Finite cyclic group, generator of the group
$\mathbf{v_i}$	V_i's voting vector
v_i; v_i'	V_i's forward and backward voting values
$\mathbf{V_A}$; $\mathbf{V_A'}$	Aggregated voting vector
$\mathbf{L_i}$	V_i's location vector
$\mathbf{L_A}$	Aggregated location vector
$\hat{L}_i$	V_i's Chosen location in one round
Ł	Length of location vector
L	Length of each voter's voting vector
L_i	Voter V_i's real location (which is private to V_i) or to say, voter V_i's row number
$L_B^i, L_{B+1}^i, \cdots, L_E^i$	V_i's voting positions/bits or to say, the columns in row L_i
L_c^i $(L_B^i \leq L_c^i \leq L_E^i)$	A voting position where V_i sets to 1 (cast vote) also referred to as a voting element or bit
$s_{ij}(s_{ij}')$	V_i's share of v_i (v_i')
$p_i; p_i'$	Sum of shares held by V_i, V_i's secret ballot
$P; P'$	Sum of p_i $(1 \leq i \leq N)$
$S_{i,C_j}; S_{i,C_j}'$	Sum of shares C_j (j = 1, 2) generates for V_i
$\tilde{S}_{i,C_j}; \tilde{S}_{i,C_j}'$	Sum of shares that C_j generates for other voters and needs to send to V_i for secret sharing
(N,N)-SS	(N,N) secret sharing
LAS	Location anonymity scheme
STPM	Secure two-party multiplication

II. Related Works

The voting technology has experienced tremendous progress over the years. Extensive research on voting, particularly e-voting recently, has been conducted and different voting techniques and systems have been proposed in [6], [18], [19], [52], [55], [38], [54], [23], [34], [25], [39], [28], [37], [53], [30], [5].

Most voting techniques are based on cryptography, such as mix-nets, blind signature, homomorphic encryption, and secret sharing. The first voting scheme was proposed by Chaum [10] utilizing anonymous channels (so-called mix-nets) in 1981. Since then, more schemes and practical systems based on mix-nets have been proposed [15], [35], [56], [12], [31].

A blind signature allows an authority to sign an encrypted message without knowing the message's context [42], [41], [33], [11], [9]. However, it is difficult to defend against misbehavior by the authority. In addition, some participants (e.g., the authority) know the intermediate results before the counting stage. This violates fairness of the voting process. Ring signature is proposed to replace the single signing authority. The challenge of using the ring signature is in preventing voters from double voting. Chow et al. propose using a linkable ring signature, in which messages signed by the same member can be correlated, but not traced back to the member [17]. A scheme combining blind signature and mix-nets is proposed in [41].

E-voting schemes based on homomorphic encryption can trace back to the seminal works by Benaloh [3], [7] and later developments in efficiency [49], [20], and receipt-freeness [27], [36], [1], [45]. Rjaskova's scheme [27], [36], [1] achieves receipt-freeness by using deniable encryption, which allows a voter to produce a fake receipt to confuse the coercer.

Several e-voting schemes exploit homomorphism that is provided by secret sharing [27], [36], [45], [3], [7], [1]. Some schemes [20], [49] are based on Shamir's threshold secret sharing [51] and focus on providing universal verifiability, privacy, and robustness for e-voting. Iftene proposes a binary (yes/no) e-voting scheme based on the Chinese remainder theorem and oblivious transfer [29]: A voter sends shares of his vote to a set of subservers; each subserver sends aggregated shares to a central server for tallying. The scheme does not provide individual verifiability and accountability: A misbehaving voter can disrupt the entire process.

Experimental voting systems include Prêt à Voter [15], Scytl [50], ADDER [32], Helios [2], Punchscan/Scantegrity [24], [14], [13], ThreeBallot [43], [44], Bingo Voting [8], VoteBox [48], Prime III [21], SplitBallot [40], and STAR-Vote [5].

VoteBox [48] utilizes a distributed broadcast network and replicated log, providing robustness and auditability in case of failure, misconfiguration, or tampering. Prime III [21] is a multimodal voting system especially devoted to the disabled. SplitBallot [40] is a (physical) split ballot voting mechanism by splitting the trust between two conflict-of-interest parties/tallying authorities. Interestingly, ThreeBallot [44] is a multi-ballot protocol that provides some of the benefits of a cryptographic voting system but without using cryptography. Scytl [50] uses a verification module (a physical device) on top of DRE. The trust, previously on the DRE, was transferred to the verification module. This solution assumes that the verification module is trusted, which may result in a "single point of failure". STAR-Vote [5] is also a DRE-style system. Prêt à Voter [15] encodes a voter's vote using a randomized candidate list. Vote privacy is ensured through randomization.

Unfortunately, due to the strict and conflicting requirements in voting, as indicated in [25], there is no current scheme/system satisfying all voting properties. For example, Grewal et al. [26] acknowledge that voter-coercion is hard to address so they redefine its meaning/scope. Helios [2], using Benaloh vote-casting approach [4] and the Sako-Kilian mixnet [46], has been proven to suffer from clash attacks [34].

III. Assumptions and Building Blocks

Here we present security assumptions and building blocks of our protocol. Web based bulletin board is also briefly discussed.

A. Assumptions and Attack Models

Suppose there are N ($N > 3$) *voters*, V_i ($1 \le i \le N$), and two tallying parties, or *collectors*, C_1 and C_2[1]. C_1 and C_2 *have conflicting interests*: Neither will share information with the other. The assumption of the existence of multiple conflict-of-interest collectors was previously proposed by Moran and Naor [40], and applied to real world scenarios like the two-party political system in the US.

In our model, collectors mutually check and restrain each other, and thus, are assumed to follow the protocol. However, unlike many previous works, we do not assume they are fully trustworthy, but only that they will not collude with each other due to conflicts of interest. Our protocol ensures that neither of them can tally the ballots with the information they have before the final tallying. Some (but not all) voters could be malicious in our model: They can send inconsistent information to different parties or deliberately deviate from the protocol. We will show that the protocol can detect such misbehaviors and identify the perpetrators without compromising honest voters' privacy.

Although (N,N) secret sharing theoretically involves mutual interaction among all voters, only a secure unicast channel between a voter and each collector is needed. Such a secure channel can be easily provided by equipping each collector with a public key cryptosystem.

We consider two types of adversaries: passive and active adversaries. Passive adversaries honestly follow the protocol but try to infer more information, while active adversaries (i.e., misbehaving voters) aim to violate the protocol by multiple voting, disturbing others' voting, disturbing the tally and eventually bringing down the protocol. When we talk about disturbing others' voting, we assume that there is no reason/incentive for a voter to give up his own voting right and disturb other unknown voters. However, he may disturb others and cast his vote simultaneously; this potential threat will be analyzed.

[1]The protocol can be extended to more than 2 with no essential difficulties.

Based on the unconditional security of (N,N) secret sharing (see Theorem 1 in [58]), a voter cannot infer any bit of information about any other voter's vote from the shares and information given by other voters. Furthermore, even though k voters collude, as long as $k \le N-2$, they together cannot gain any bit of information about the vote of any of the other voters either. Due to the in-process check and enforcement during the vote casting, misbehavior of any voter can be pinpointed during the vote casting and the dishonest voter can be identified.

B. Technical Components (TPs)

TP1: Universal verifiable voting vector. For N voters and M candidates, a voting vector $\mathbf{v}_i$ for V_i is a binary vector of $L = N \times M$ bits. The vector can be visualized as a table with N rows and M columns. Each candidate corresponds to a column. Via a robust location anonymization scheme described in Section IV-B, each voter *secretly* picks a unique row. A voter V_i will put a 1 in the entry at the row and column corresponding to a candidate V_i votes for (let the position be L_c^i), and put 0 in all other entries. During tallying, all voting vectors will be aggregated. From the tallied voting vector (denoted as $\mathbf{V_A}$), the votes for candidates can be incrementally tallied. Any voter can check his vote and also visually verify that his vote is indeed counted in the final tally. Furthermore, anyone can verify the vote totals for each candidate.

TP2: Forward and backward mutual lock voting. From V_i's voting vector (with a single entry of 1 and the rest of 0), a forward value v_i (=$2^{L-L_c^i}$) and a backward value v_i' (=$2^{L_c^i-1}$) can be derived. Importantly, $v_i \times v_i' = 2^{L-1}$, regardless which candidate V_i votes for. During the vote-casting, V_i uses simplified (N,N)-SS to cast their vote using both v_i and v_i' respectively. v_i and v_i' jointly ensure the correctness of the vote-casting process, and enforce V_i to cast *one and only one* vote; any deviation, such as multiple voting, will be detected.

Notes: 1) There is no incentive for a voter to give up his own voting right and disrupt others. However, if a voter indeed puts the single 1 in another voter's location, the misbehaving voter's voting location in $\mathbf{V_A}$ and $\mathbf{V_A'}$ will be 0. If this happens, C_1 and C_2 can jointly find this location and then, along with the information collected during location anonymity, identify the perpetrator. 2) To prevent a collector from having all $N-1$ shares for a voter, each collector creates half of $N-1$ shares. 3) Interestingly, the new e-voting model deliberately distinguishes between a *private vote* and its *secret ballot*. Different from existing voting systems, a voter's vote is kept secret to himself. However, its corresponding ballot, even called *secret ballot*, is revealed to the public in the vote-casting.

TP3: In-process check and enforcement. During the voting processes, collectors will jointly perform two cryptographic checks on the voting values of each voter. The first check uses STPM to prevent a voter from wrongly generating his share in the vote-casting stage. The second check prevents a voter from publishing an incorrect *secret ballot* when collectors collect it from a voter. The secret ballot is the modular addition of a voter's own share and the share summations that the voter received from collectors.

C. Cryptographic Primitives

1) Simplified (N,N) Secret Sharing ((N,N)-SS): A secret s is partitioned into N shares s_i ($1 \leq i \leq N$) such that $s = \sum_{i=1}^{N} s_i$. For a group of N members, each receives one of the shares. All N members need to pool their shares together to recover s. This secret sharing scheme is additively homomorphic [58]: The sum of two shares s_i and s'_i (corresponding to secrets s and s' respectively) is a share of the sum of the two secrets s and s'.

2) Secure Two-party Multiplication (STPM): STPM is proposed by Samet and Miri [47]. Initially, each of parties, M_i ($i = 1,2$), holds a *private* input x_i. At the end of the protocol, M_i will have a *private* output r_i, such that $x_1 \times x_2 = r_1 + r_2$. The protocol works as follows: 1) M_1 chooses a private key d and a public key e for an additively homomorphic public-key encryption scheme, with encryption and decryption functions being E and D, respectively. 2) M_1 sends $E(x_1,e)$ to M_2. 3) M_2 selects a random number r_2, computes $E(x_1,e)^{x_2}E(r_2,e)^{-1}$, and sends the result back to M_1. 4) M_1 decrypts the received value into $D(E(x_1,e)^{x_2}E(r_2,e)^{-1},d)$ and takes it as r_1.

D. A Web Based Bulletin Board

A web based bulletin board allows anyone to monitor the dynamic vote casting and tallying in real time. This makes the voting and tallying process totally visible (i.e., transparent) to all voters. The bulletin board will dynamically display 1) ongoing vote casting; 2) incremental aggregation of the secret ballots; and 3) incremental vote counting/tallying.

Note that all the incremental aggregations of secret ballots, except the final one, reveal no information of any individual vote or any candidate's counts. Only at the time when the final aggregation is completed are all individual votes suddenly visible in their entirety, but in an anonymous manner. It is this sudden transition that precludes any preannouncement of partial voting results. Moreover, this transition creates a seamless connection between vote-casting & ballot confirmation and vote-tallying & verification so that both voter privacy and voter assurance can be achieved simultaneously. This is a unique feature of the new e-voting system, comparing to all existing ones.

IV. Mutual Restraining Voting Protocol

We first elaborate on each of the stages along with the applications of the technical components (TPs), and then give the design of our location anonymity scheme (LAS).

A. Description of Voting Stages with applications of TPs

Stage 1: Initialization. The following computations are carried out on a cyclic group $\mathbf{Z_A}$, on which the Discrete Logarithmic Problem (DLP) is intractable. $\mathbf{A} = max\{\mathbf{A}_1, \mathbf{A}_2\}$, in which $\mathbf{A}_1$ is a prime larger than 2^{1024} and $\mathbf{A}_2$ is a prime larger than $2^{2L} - 2^{L+1} + 1$. Let the number of voters be N and the number of candidates be M. $V_1, \cdots, V_N$ are voters.

Each voter generates his voting vector by the following two steps (TP1): 1) Voter V_i, by collaboratively executing a LAS (Section IV-B) with other voters, obtains a unique and secret location L_i. 2) V_i arranges a voting vector $\mathbf{v}_i$ of the length $L = N \times M$ bits into N rows (corresponding to N voters) and M columns (corresponding to M candidates); V_i fills a 1 in his row (i.e., the L_ith row) and the column for the candidate he votes, and 0 in all other entries. This arrangement can support voting scenarios including "yes-no" voting for one candidate and 1-out-of-M voting for M candidates with abstaining or without.

Stage 2: Vote casting. From the voting vector $\mathbf{v}_i$ (with a singleton 1 and all other entries 0), V_i derives two decimal numbers v_i and $v_i^{'}$ from $\mathbf{v}_i$: 1) v_i is the decimal number corresponding to the binary string represented by $\mathbf{v}_i$; 2) $v_i^{'}$ is the decimal number corresponding to $\mathbf{v}_i$ *in reverse*.

In other words, if V_i sets the L_c^ith bit of $\mathbf{v}_i$ to 1, we have $v_i = 2^{L-L_c^i}$ and $v_i^{'} = 2^{L_c^i - 1}$, thus $v_i \times v_i^{'} = 2^{L-1}$: v_i and $v_i^{'}$ are said to be *mutually restrained*. This technique will be used to develop an effective enforcement mechanism that enforces the single-voting rule with privacy guarantee: The vote given by a voter will not be disclosed as long as the voter casts one and only one vote.

Next, V_i indirectly shares v_i and $v_i^{'}$ with other voters using (N,N)-SS. Note that the sharing processes of v_i and $v_i^{'}$ are the same but independent from each other. Our initial design is to let each voter create N shares and then distribute $N-1$ shares to the rest voters. However, this requires synchronous operations among all voters. An alternate design allows asynchronous operations, but still retains all properties of (N,N)-SS.

In this asynchronous vote-casting, voters do not need to interact with each other. Instead, two collectors generate respective shares for all voters. They work with every voter with steps as below:

- Two collectors generate $N-1$ shares for each voter in advance, half per voter per collector.
- Whenever a voter V_i logs into the system to cast vote, two collectors will each send their half shares (in fact, the sum of these shares) to this voter, i.e., $C_j (j = 1,2)$ sends S_{i,C_j} to V_i. V_i computes his own share $s_{ii} = v_i - S_{i,C_1} - S_{i,C_2}$. Similarly, V_i gets s_{ii}'. V_i then sends his commitments (i.e., $g^{s_{ii}}$, $g^{s_{ii}^{'}}$, $g^{s_{ii}s_{ii}^{'}}$) to two collectors. Notes: under the assumption that two collectors have conflicting interests, neither of them should be able to derive V_i's vote from his commitment.
- Two collectors verify V_i's vote using **Sub-protocol 1** (described later) and if passed, send the shares of the other $N-1$ voters (one for each voter) to V_i. Specially, C_j sends $\tilde{S}_{i,C_j}$ (i.e., the sum of the shares C_j generates for the $N/2$ voters) to V_i. V_i then publishes the secret ballot $p_i = s_{ii} + \tilde{S}_{i,C_1} + \tilde{S}_{i,C_2}$. Two collectors can verify V_i's

published value using **Sub-protocol 2** (described later). The process of $p_i^{'}$ is the same.

As proven by Theorem 1 in [58], it is clear that neither of two collectors can obtain any voter's own share or vote, unless two collectors collude and exchange the shares they have generated.

Stage 3: Collection/Tally. Collectors (and voters if they want) collect secret ballots p_i $(1 \le i \le N)$ from all voters and obtain $P = \sum_{i=1}^{N} p_i$. P is decoded into a tallied binary voting vector $\mathbf{V_A}$ of length L. The same is done for p'_i $(1 \le i \le N)$ to obtain P', and consequently $\mathbf{V'_A}$. If voters have followed the protocol, these two vectors will be reverse to each other by their initialization in Stage 1.

It might be possible that some voters do not cast their votes, purposely or not, which can prevent $\mathbf{V_A}$ or $\mathbf{V'_A}$ from being computed. Two possible solutions are: 1) after LAS, let all voters cast a default vote (e.g., abstain), and 2) subtract S_{i,C_1} and S_{i,C_2} of a voter who didn't cast from the aggregation of the secret ballots p_i, and $S_{i,C_1}^{'}$ and $S_{i,C_2}^{'}$ from p'_i.

Stage 4: Verification. Anyone can verify whether $\mathbf{V_A}$ is a reverse of $\mathbf{V'_A}$ and whether each voter has cast one and only one vote. V_i can verify the entry L_c^i (corresponding to the candidate that V_i votes for) has been correctly set to 1 and the entries for other candidates are 0. Furthermore, the tallied votes for all candidates can be computed and verified via $\mathbf{V_A}$ and $\mathbf{V'_A}$. In summary, both individual and universal verification are naturally supported by this protocol.

Stage 5: In-process check and enforcement. A voter may misbehave in different ways. Examples include: 1) multiple voting; 2) disturbing others' voting; and 3) disturbing the total tally. All examples of misbehavior are equivalent to an offender inserting multiple 1s in the voting vector. The following two sub-protocols, which are collectively known as *in-process check and enforcement* (TP3), ensure that each voter will put a single 1 in his voting vector, i.e., vote once and only once.

Sub-protocol 1: 1) Recall that in Stage 2, $C_j (j = 1, 2)$ has S_{i,C_j} and S'_{i,C_j} for V_i.

2) Since V_i publishes $g^{s_{ii}}$ and $g^{s_{ii}^{'}}$, C_1 can compute and publish $(g^{s_{ii}})^{S_{i,C_1}^{'}}$ and $(g^{s_{ii}^{'}})^{S_{i,C_1}}$. In addition, C_1 publishes $g^{S_{i,C_1} S_{i,C_1}^{'}}$. Similarly, C_2 publishes $(g^{s_{ii}})^{S_{i,C_2}^{'}}$, $(g^{s_{ii}^{'}})^{S_{i,C_2}}$, and $g^{S_{i,C_2} S_{i,C_2}^{'}}$.

3) C_1 and C_2 cooperatively compute $g^{S_{i,C_1} S_{i,C_2}^{'}} \times g^{S_{i,C_1}^{'} S_{i,C_2}}$. A straightforward application of Diffie-Hellman key agreement [22] to obtain $g^{S_{i,C_1} S_{i,C_2}^{'}}$ and $g^{S_{i,C_1}^{'} S_{i,C_2}}$ will not work[2]. Therefore, STPM is used to compute $g^{S_{i,C_1} S_{i,C_2}^{'}} \times g^{S_{i,C_1}^{'} S_{i,C_2}}$ without disclosing $g^{S_{i,C_1}}, g^{S_{i,C_1}^{'}}, g^{S_{i,C_2}}$ and $g^{S_{i,C_2}^{'}}$ as follows:

- Executing STPM, C_1 and C_2 obtain r_1 and r'_2 respectively such that $r_1 + r'_2 = S_{i,C_1} S_{i,C_2}^{'}$;
- Executing STPM, C_1 and C_2 obtain r'_1 and r_2 respectively such that $r'_1 + r_2 = S_{i,C_1}^{'} S_{i,C_2}$;
- C_1 computes $g^{r_1 + r'_1}$, C_2 computes $g^{r_2 + r'_2}$ and then they exchange the results;
- Both collectors obtain $g^{r_1 + r'_2 + r'_1 + r_2} = g^{S_{i,C_1} S_{i,C_2}^{'}} \times g^{S_{i,C_1}^{'} S_{i,C_2}}$.

4) Each collector uses V_i's commitment and the above computation results to obtain $g^{s_{ii} s_{ii}^{'}} \times (g^{s_{ii}})^{S_{i,C_1}^{'}} \times (g^{s_{ii}^{'}})^{S_{i,C_1}} \times g^{S_{i,C_1} S_{i,C_1}^{'}} \times (g^{s_{ii}})^{S_{i,C_2}^{'}} \times (g^{s_{ii}^{'}})^{S_{i,C_2}} \times g^{S_{i,C_2} S_{i,C_2}^{'}} \times g^{S_{i,C_1} S_{i,C_2}^{'}} \times g^{S_{i,C_1}^{'} S_{i,C_2}}$. The collectors can verify that the product equals $g^{2^{L-1}}$. If not, V_i must have shared v_i or $v_i^{'}$ incorrectly.

Sub-protocol 2: While Sub-protocol 1 ensures that V_i generates s_{ii} properly, Sub-protocol 2 enforces that V_i will faithfully publish the secret ballots, p_i and $p_i^{'}$.

1) Recall that $C_j (j = 1, 2)$ has $\tilde{S}_{i,C_j}$ and $\tilde{S}'_{i,C_j}$ for V_i, so C_j publishes $g^{\tilde{S}_{i,C_j}}$ and $g^{\tilde{S}'_{i,C_j}}$.

2) From the published p_i and $p_i^{'}$, the collectors compute g^{p_i} and $g^{p_i^{'}}$. Since $g^{s_{ii}}$ and $g^{s_{ii}^{'}}$ are published and verified in Sub-protocol 1, collectors will verify that $g^{s_{ii}} g^{\tilde{S}_{i,C_1}} g^{\tilde{S}_{i,C_2}} = g^{p_i}$ and $g^{s_{ii}^{'}} g^{\tilde{S}_{i,C_1}^{'}} g^{\tilde{S}_{i,C_2}^{'}} = g^{p_i^{'}}$. If either of these fails, V_i must have published the wrong secret ballots p_i and/or $p_i^{'}$.

B. Design of a Robust and Efficient LAS

Inspired by the work in [58], we propose a new location anonymity scheme (LAS) that is robust and efficient. Our new scheme solves the following problem with the previous schemes: If a member misbehaves in next rounds by selecting multiple locations or a location that is already occupied by another member, the location selection in [58] may never finish. Our new LAS is based on the mutual lock voting mechanism and works as follows:

1) Each voter V_i initializes a location vector $\mathbf{L}_i$ (of length $Ł$) with 0s. V_i randomly selects a location $\hat{L}_i$ $(1 \le \hat{L}_i \le Ł)$ and sets the $\hat{L}_i$th element/bit of $\mathbf{L}_i$ to 1.
2) From $\mathbf{L}_i$, V_i obtains two values l_i and $l_i^{'}$ by: 1) encoding $\mathbf{L}_i$ into a decimal number l_i[3]; and 2) reversing $\mathbf{L}_i$ to be $\mathbf{L}_i^{'}$ and encoding it into a decimal number $l_i^{'}$. For example, if $\mathbf{L}_i = [000010]$, we obtain $l_i = 10$ and $l_i^{'} = 10000$. Evidently, $l_i \times l_i^{'} = 10^{Ł-1}$.
3) V_i shares l_i and $l_i^{'}$ using (N,N)-SS as in Stage 2. All voters can obtain the aggregated location vector $\mathbf{L}_A$ and $\mathbf{L}_A^{'}$. If V_i has followed the protocol, $\mathbf{L}_A$ and $\mathbf{L}_A^{'}$ are the reverse of the other.

[2]If C_1 exchanges his $g^{S_{i,C_1}}$ and $g^{S_{i,C_1}^{'}}$ with C_2's $g^{S_{i,C_2}}$ and $g^{S_{i,C_2}^{'}}$, since $g^{s_{ii}}$ and $g^{s_{ii}^{'}}$ are published by V_i, C_1 and C_2 each can obtain $g^{s_{ii} + S_{i,C_1} + S_{i,C_2}}$ and $g^{s_{ii}^{'} + S_{i,C_1}^{'} + S_{i,C_2}^{'}}$ which correspond to g^{v_i} and $g^{v_i^{'}}$. Because there are only L possibilities of each voter's vote, C_1 and C_2 each can simply try L values to find the vote, v_i and $v_i^{'}$. This violates both vote anonymity (vote is known) and voter privacy (location is known).

[3]A decimal encoding, instead of a binary one, is used to encode $\mathbf{L}_i$. The motivation is illustrated below. Assume that the binary encoding is adopted. Let the location vectors of voters V_i, V_j and V_k be $\mathbf{L}_i = 000010$, $\mathbf{L}_j = 000010$, and $\mathbf{L}_k = 000100$, respectively. Therefore, $\mathbf{L}_A = 001000$: Voters cannot tell if they have obtained unique locations. This will not be the case if $\mathbf{L}_i$ uses a larger base; however, encoding $\mathbf{L}_i$ in a larger base consumes more resources. Decimal is a trade-off we adopted to strike a balance between fault tolerance and performance. The probability of having more than 10 voters collide at the same location is considerably lower than that of 2.

4) V_i checks if the $\hat{L}_i$th element/bit of $\mathbf{L}_A$ is 1. If so, V_i has successfully selected a location without colliding with others. V_i also checks if everyone has picked a location without colliding with others by examining whether $max(\mathbf{L}_A) = 1$. If there is at least one collision, steps 1 through 3 will restart. In a new round, voters who have successfully picked a location without collision in the previous round select the same location, while others randomly select from locations not been chosen.
5) The in-process check and enforcement mechanism (Stage 5) is concurrently executed by collectors to enforce that a voter will select one and only one location in each round. Furthermore, the mechanism (proved in Section V-B) ensures that any attempt of inducing collision by deliberately selecting an occupied position will be detected and, hence, such misbehavior will be precluded.
6) Once all location collisions are resolved in a round, each voter removes non-occupied locations ahead of his own and obtains his real location $L_i = \sum_{j=1}^{\hat{L}_i} (\mathbf{L}_A)_j$. After the adjustment, the occupied locations become contiguous. The length of the adjusted $\mathbf{L}_i$, denoted as $\tilde{L}$, equals to the number of voters, N.

We will complement the above discussion with analysis (Section V-B) and simulation results (Section VI).

Notes: 1) Location anonymity, a special component in our protocol, seems to be an additional effort for voters. However, it is beneficial since voters not only select their secret locations, but also learn/practice vote-casting ahead of the real election. The experiments show that 2 to 3 rounds are generally enough. 2) Location anonymity can be executed asynchronously. 3) A malicious participant deliberately inducing a collision by choosing an already occupied location will be identified.

Under the assumption that C_1 and C_2 have conflicting interests and thus will check each other but not collude, more deterministic and efficient LAS can be designed. One algorithm can be: two collectors perform double encryption (of 1 to N) and double shuffle before sending results to voters in a way such that neither can determine which voter gets which number, even though a collector may collude with some voter(s).

V. Property Proof and Analysis

A. Proof of Robustness of Voting

The protocol is robust in the sense that a misbehaving voter will be identified. A misbehaving voter V_i may:

- submit an invalid voting vector $\mathbf{v}_i$ ($\mathbf{v}_i^{'}$) with more than one (or no) 1s;
- generate wrong s_{ii} ($s_{ii}^{'}$), thus wrong commitment $g^{s_{ii}}$ ($g^{s_{ii}^{'}}$);
- publish an incorrect secret ballot p_i ($p_i^{'}$) such that $p_i \neq s_{ii} + \tilde{S}_{i,C_1} + \tilde{S}_{i,C_2}$ ($p_i^{'} \neq s_{ii}^{'} + \tilde{S}_{i,C_1}^{'} + \tilde{S}_{i,C_2}^{'}$).

First, we show that a voter submitting an invalid voting vector $\mathbf{v}_i$ ($\mathbf{v}_i^{'}$) with more than one 1s will be detected. Without loss of generality, we assume two positions, L_c^i and $L_c^{i'}$, are set to 1. (A voter can also misbehave by putting 1s at inappropriate positions, i.e., positions assigned to other voters; we will analyze this later.) Thus the voter V_i obtains v_i ($v_i^{'}$), such that

$$\begin{aligned} v_i &= 2^{(L-L_c^i)} + 2^{(L-L_c^{i'})}, v_i^{'} = 2^{(L_c^i-1)} + 2^{(L_c^{i'}-1)}, \\ v_i \times v_i^{'} &= 2^{L-1} + 2^{L-1} + 2^{L-L_c^i+L_c^{i'}-1} + 2^{L-L_c^{i'}+L_c^i-1}. \end{aligned}$$

All the computations are moduli operations. If we use $\mathbf{Z_A}$, which has at least $2^{2L} - 2^{L+1} + 1$ elements/bits, we have $v_i \times v_i^{'} \neq 2^{L-1}$ and $g^{v_i \times v_i^{'}} \neq g^{2^{L-1}}$. Assuming V_i generates an invalid voting vector without being detected, this will lead to the following contradiction by Sub-protocol 1:

$$\begin{aligned} g^{2^{L-1}} &= g^{s_{ii}s_{ii}^{'}} \times (g^{s_{ii}})^{S_{i,C_1}^{'}} \times (g^{s_{ii}^{'}})^{S_{i,C_1}} \times g^{S_{i,C_1}S_{i,C_1}^{'}} \times (g^{s_{ii}})^{S_{i,C_2}^{'}} \\ &\quad \times (g^{s_{ii}^{'}})^{S_{i,C_2}} \times g^{S_{i,C_2}S_{i,C_2}^{'}} \times g^{S_{i,C_1}S_{i,C_2}^{'}} \times g^{S_{i,C_1}^{'}S_{i,C_2}} \\ &= g^{(s_{ii}+S_{i,C_1}+S_{i,C_2})(s_{ii}^{'}+S_{i,C_1}^{'}+S_{i,C_2}^{'})} = g^{v_i v_i^{'}}. \end{aligned}$$

Similar proof applies to an invalid voting vector without 1s.

Next, we show that V_i cannot generate wrong s_{ii} or $s_{ji}^{'}$ such that $s_{ii} + S_{i,C_1} + S_{i,C_2} \neq v_i$ or $s_{ii}^{'} + S_{i,C_1}^{'} + S_{i,C_2}^{'} \neq v_i^{'}$. If Sub-protocol 1 fails to detect this discrepancy, there is: $g^{(s_{ii}+S_{i,C_1}+S_{i,C_2})(s_{ii}^{'}+S_{i,C_1}^{'}+S_{i,C_2}^{'})} = g^{2^{L-1}}$. Since the computation is on $\mathbf{Z_A}$, we have: $(s_{ii}+S_{i,C_1}+S_{i,C_2})(s_{ii}^{'}+S_{i,C_1}^{'}+S_{i,C_2}^{'}) = 2^{L-1}$. Given that:

$$\begin{gathered} s_{ii}+S_{i,C_1}+S_{i,C_2} \neq v_i, \;\; s_{ii}^{'}+S_{i,C_1}^{'}+S_{i,C_2}^{'} \neq v_i^{'}, \\ (s_{ii}+S_{i,C_1}+S_{i,C_2})(s_{ii}^{'}+S_{i,C_1}^{'}+S_{i,C_2}^{'}) = 2^{L-1}, \end{gathered}$$

there must exist one and only one position $L_c^{i'}$ which is set to 1 and $L_c^{i'} \neq L_c^i$. This indicates that V_i gives up his own voting positions, but votes at a position assigned to another voter. In this case, V_i's voting positions in $\mathbf{V_A}$ and $\mathbf{V_A^{'}}$ will be 0[4]. If this happens, C_1 and C_2 can collaboratively find V_i's row that has all 0s in the voting vector (arranged in a $N \times M$ array).

Third, we show that a voter cannot publish an incorrect p_i ($p_i^{'}$) to disturb the tally. Given that a misbehaving V_i publishes p_i ($p_i^{'}$) such that $s_{ii} + \tilde{S}_{i,C_1} + \tilde{S}_{i,C_2} \neq p_i$ ($s_{ii}^{'} + \tilde{S}_{i,C_1}^{'} + \tilde{S}_{i,C_2}^{'} \neq p_i^{'}$), we obtain $g^{s_{ii}+\tilde{S}_{i,C_1}+\tilde{S}_{i,C_2}} \neq g^{p_i}$ ($g^{s_{ii}^{'}+\tilde{S}_{i,C_1}^{'}+\tilde{S}_{i,C_2}^{'}} \neq g^{p_i^{'}}$) which will fail in Sub-protocol 2. Note that $g^{s_{ii}}$ and $g^{s_{ii}^{'}}$ have passed the verification of Sub-protocol 1, and $\tilde{S}_{i,C_1}$ and $\tilde{S}_{i,C_2}$ (also, $\tilde{S}_{i,C_1}^{'}$ and $\tilde{S}_{i,C_2}^{'}$) are computed by two collectors with conflicts of interest. Thus, there is no way for the voter to publish an incorrect p_i ($p_i^{'}$) without being detected.

B. Proof of Robustness of Location Anonymity

The analysis in Section V-A shows that no voter can choose more than one positions during the location anonymization process. However, this does not address the problem that a malicious participant deliberately induces collisions by choosing a location that is already occupied by another voter. We will demonstrate that our proposed LAS is robust against this.

[4]Unless, of course, another voter puts a 1 in V_i's position. We can either trace this back to a voter that has its positions all 0s, or there is a loop in this misbehaving chain, which causes no harm to non-misbehaving voters.

Let the collision happen at $\hat{L}_i$, i.e., $\hat{L}_i$ is chosen by V_i in the previous round, and both V_i and V_j claim $\hat{L}_i$ in the current round. In this case, V_j is the voter who deliberately induces collision. To identify a voter who chooses $\hat{L}_i$ in a given round, C_1 and C_2 do the following collaboratively. For each voter, using the STPM, C_1 and C_2 compute $Q = g^{g^{S_{i,C_1}+S_{i,C_2}}}$ ($Q^{'} = g^{g^{S^{'}_{i,C_1}+S^{'}_{i,C_2}}}$) and check whether $g^{g^{10^{\tilde{\hat{L}}-\hat{L}_i}}}/g^{s_{ii}} = Q$ ($g^{g^{10^{\hat{L}_i-1}}}/g^{s^{'}_{ii}} = Q^{'}$). By doing this, the collectors identify the voter who selects $\hat{L}_i$ without divulging other voters' locations. Although the honest voter V_i who chooses $\hat{L}_i$ is exposed along with the malicious V_j, V_i can restore location anonymity by selecting another position in the next round and V_j should be punished.

C. Analysis of Main Properties

Correctness. If all voters are honest and correctly follow the protocol, $\mathbf{V_A}$ ($\mathbf{V^{'}_A}$) is the aggregation of all votes. Assume that N voters vote at positions $L^1_c, L^2_c, \cdots, L^N_c$ ($L^1_c \neq L^2_c \neq \cdots \neq L^N_c$) respectively. Each V_i computes v_i as $v_i = 2^{(L-L^i_c)}$ $(i = 1, \cdots, N)$. Due to the homomorphism of (N,N)-SS, we have:

$$\begin{aligned} P &= v_1 + v_2 + \cdots + v_N \\ &= 2^{(L-L^1_c)} + 2^{(L-L^2_c)} + \cdots + 2^{(L-L^N_c)}, \\ \mathbf{V_A} &= \mathbf{v}_1 + \mathbf{v}_2 + \cdots + \mathbf{v}_N. \end{aligned}$$

Since each voter votes at one of his own voting positions (i.e., $L^i_c \neq L^j_c$ where $i \neq j$), there is no carry in the additions. Thus, each V_i can check the L^i_cth bit of $\mathbf{V_A}$ to see if his vote has been correctly counted. Similar arguments apply to $\mathbf{V^{'}_A}$.

Anonymity. The protocol preserves anonymity if no more than $N-2$ voters collude. The claim follows the proof in [58]. Furthermore, the protocol splits trust, traditionally vested in a central authority, now between two non-colluding collectors with conflicts of interest.

Verifiability. Both individual and universal verifiability are achieved since anyone can verify if his vote and vote totals.

Eligibility. Voters have to be authenticated for their identities before obtaining voting positions. Traditional authentication mechanisms can be integrated into the protocol.

Prevention of multiple voting. The forward and backward mutual lock voting allows a voter to set one and only one of his voting positions to 1 (enforced by Sub-protocol 1).

Fairness. Fairness is ensured due to the following unique property of (N,N)-SS: no one can obtain any information before the final tally, and only when all N secret ballots are aggregated is the sum of all secret votes obtained in its entirety and in an anonymous manner. It is this sudden transition that precludes any preannouncement of partial voting results.

Transparency and voter assurance. The protocol is transparent in that voters participate in the whole voting process, rather than entrusting the process to a central authority like in many previous e-voting solutions.

D. Analysis of Protocol Performance and Complexity

In this section, we analyze the computational complexity of forward and backward mutual lock voting and in-process enforcement. Suppose that each message takes T bits. Since the protocol works on a cyclic group $\mathbf{Z_A}$ ($\mathbf{A} = max\{\mathbf{A}_1, \mathbf{A}_2\}$, in which $\mathbf{A}_1$ is a prime greater than 2^{1024} and $\mathbf{A}_2$ is a prime greater than $2^{2L} - 2^{L+1} + 1$), we see that $T = O(L)$.

The forward and backward mutual lock voting involves two independent sharing processes of v_i and $v^{'}_i$. A voter's communication cost includes publishing his commitments and secret ballot p_i, so the total is $O(T)$. His computational cost includes computing v_i, s_{ii}, p_i, and the commitments (e.g., $g^{s_{ii}}$), each costing $O(T)$, $O(T)$, $O(T)$ and $O(T^3)$ respectively. The same cost applies to the sharing of $v^{'}_i$. **Notes**: The commitments can be typically computed by a calculator efficiently, thus, the complexity of $O(T^3)$ will not become a performance issue.

The collector C_j's communication cost involves: 1) sending sums of shares to each voter; 2) publishing $g^{s_{ii}S_{i,C_j}}, g^{s_{ii}S^{'}_{i,C_j}}$, and $g^{S_{i,C_j}S^{'}_{i,C_j}}$ for each voter; 3) publishing $g^{S_{i,C_1}S^{'}_{i,C_2}} \times g^{S^{'}_{i,C_1}S_{i,C_2}}$ for each voter; and 4) publishing $g^{\tilde{S}_{i,C_j}}$ or $g^{\tilde{S}^{'}_{i,C_j}}$ for each voter. Assume that the STPM messages are encoded into $\tilde{T}$-bits when computing $g^{S_{i,C_1}S^{'}_{i,C_2}} \times g^{S^{'}_{i,C_1}S_{i,C_2}}$. The communication costs are $O(T)$, $O(T)$, $O(\tilde{T})$, and $O(T)$, respectively. For N voters, the total cost for each collector is $(O(T) + O(T) + O(\tilde{T}) + O(T))N$.

The computation by C_j includes: 1) generating half of $N-1$ shares for each voter, which costs $O(N^2T)$ totally; 2) deriving S_{i,C_j}, $S^{'}_{i,C_j}$, $\tilde{S}_{i,C_j}$, and $\tilde{S}^{'}_{i,C_j}$ during the vote-casting process which costs $O(N^2T)$; 3) summing up the p_i during voting collection/tally which costs $O(NT)$; 4) the computational costs of Sub-protocol 1 and Sub-protocol 2. In Sub-protocol 1, both collectors: 1) compute $g^{s_{ii}S^{'}_{i,C_j}}, g^{s^{'}_{ii}S_{i,C_j}}$ and $g^{S_{i,C_j}S^{'}_{i,C_j}}$ $(j = 1, 2)$; 2) compute $g^{S_{i,C_1}S^{'}_{i,C_2}} \times g^{S^{'}_{i,C_1}S_{i,C_2}}$, which involve STPM; and 3) multiply the commitments. Computing $g^{s_{ii}S^{'}_{i,C_j}}, g^{s^{'}_{ii}S_{i,C_j}}$, and $g^{S_{i,C_j}S^{'}_{i,C_j}}$ costs $O(T^3)$. Multiplying the commitments costs $O(T^2)$. Computing $g^{S_{i,C_1}S^{'}_{i,C_2}} \times g^{S^{'}_{i,C_1}S_{i,C_2}}$ consists of obtaining r_1 ($r^{'}_1$) and r_2 ($r^{'}_2$) with STPM, computing $g^{r_1+r^{'}_1}$ and $g^{r_2+r^{'}_2}$, and multiplying $g^{r_1+r^{'}_1}$ and $g^{r_2+r^{'}_2}$. Let the complexity for STPM be $O(TPMC)$. The cost of computing $g^{S_{i,C_1}S^{'}_{i,C_2}} \times g^{S^{'}_{i,C_1}S_{i,C_2}}$ is $O(TPMC) + O(T^3) + O(T^2)$. The total computational cost of Sub-protocol 1 is $O(T^3) + O(T^2) + O(TPMC)$ for each voter. In Sub-protocol 2, the collectors: 1) compute $\tilde{S}_{i,C_j}$ and $\tilde{S}^{'}_{i,C_j}$; 2) compute $g^{\tilde{S}_{i,C_j}}$ and $g^{\tilde{S}^{'}_{i,C_j}}$; 3) multiply $g^{s_{ii}}$, $g^{\tilde{S}_{i,C_1}}$ and $g^{\tilde{S}_{i,C_2}}$, and also $g^{s^{'}_{ii}}$, $g^{\tilde{S}^{'}_{i,C_1}}$ and $g^{\tilde{S}^{'}_{i,C_2}}$; and 4) compute g^{p_i} and $g^{p^{'}_i}$. These computations cost $O(NT)$, $O(T^3)$, $O(T^2)$ and $O(T^3)$, respectively. The total cost of Sub-protocol 2 is $O(NT) + O(T^3) + O(T^2) + O(T^3)$ for each voter.

VI. Simulation Results

We implemented our protocol simulation in Java, and present results here. The experiments were carried out on a computer with a 1.87GHz CPU and 32G of memory. For each experiment, we took the average of 10 rounds of simulations.

1-out-of-2 voting process is simulated. Thus, the length of the voting vector is $L = 2N$ where N is the number of voters.

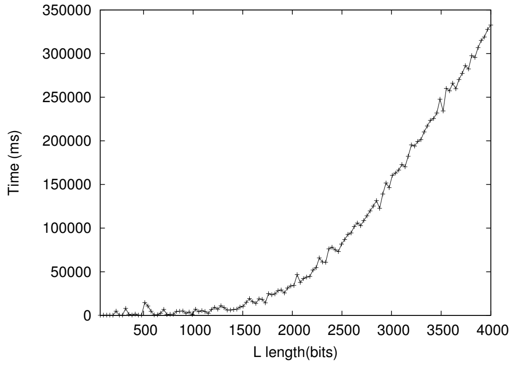

Fig. 1: Collectors run Sub-protocol 1 in TP3 against one voter

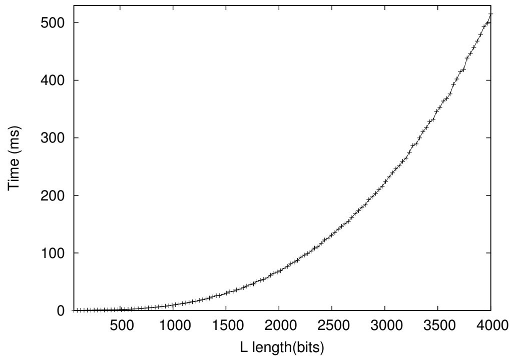

Fig. 2: Collectors run Sub-protocol 2 in TP3 against one voter

The computation time for a voter V_i is negligible since only two subtractions are needed for s_{ii} and two additions for p_i, and the commitments can be obtained by using a calculator sufficiently. Collectors however require heavy load of calculation, so our simulation focuses on collectors' operations.

Figure 1 and 2 show the computation time of in Sub-protocol 1 and Sub-protocol 2 respectively. Sub-protocol 1 was dominated by STPM, due to the computationally intensive Paillier Cryptosystem used in our implementation. However, this should not be an issue in real life since the collectors usually possess greater computing power.

Figure 3 shows the time for one collector to collect and tally votes. The execution time depends on the number of voters N and the length of $p_i = L$. As L increases, the voting collection/tally time increases by $NL = O(L^2)$.

The simulation results confirm the performance analysis in Section V-D. Most operations are quite efficient. For example, when $L = 4000$ (and $N = 2000$), collecting and tallying votes took only 0.005 seconds. For the in-process enforcement protocol however, it took the collectors 332 seconds to complete Sub-protocol 1 and 0.515 seconds to complete Sub-protocol 2. To amortize the relatively high cost, the collectors may randomly sample voters for misbehavior checking and only resort to full checking when a discrepancy in the tally is detected.

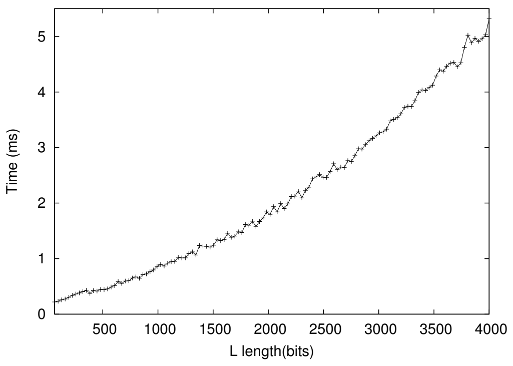

Fig. 3: One collector collects/tallies votes

VII. Conclusion

We proposed a robust, assurable, transparent, and mutual restraining e-voting protocol that is designed to exploit the conflicts of interest in multiple tallying authorities, such as in the two-party political system in the United States. Our protocol is built upon three novel technical contributions—verifiable voting vector, forward and backward mutual lock voting, and proven in-process enforcement. These three technical contributions, along with transparent vote-casting and tallying processes, incremental aggregation of secret ballots, and incremental vote tallying for candidates, deliver voter assurance. Each voter can be assured that his vote is counted both technologically and visibly. Through the analysis and simulation, we demonstrated the robustness, effectiveness, and feasibility of our protocol.

We plan to further improve our protocol in the future, particularly, in two aspects. One is the scalability, and the other is vote selling and voter coercion.

References

[1] A. P. Adewole, A. S. Sodiya, and O. A. Arowolo. A receipt-free multi-authority e-voting system. *Int. Jour. of Comp. App.*, 30(6):15–23, Sep. 2011.

[2] B. Adida. Helios: Web-based open-audit voting. In *USENIX Security Symposium*, volume 17, pages 335–348. USENIX Association, 2008.

[3] J. Benaloh. *Verifiable Secret Ballot Elections*. PhD thesis, Yale University, 1987.

[4] J. Benaloh. Simple verifiable elections. In *Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop*, pages 5–5. USENIX Association, 2006.

[5] J. Benaloh, M. Byrne, P. Kortum, N. McBurnett, O. Pereira, P. Stark, and D. Wallach. Star-vote: A secure, transparent, auditable, and reliable voting system. *arXiv preprint arXiv:1211.1904*, 2012.

[6] J. Benaloh and A. Tuinstra. Receipt-free secret-ballot elections. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 544–553, New York, 1994. ACM.

[7] J. Benaloh and M. Yung. Distributing the power of a government to enhance the privacy of voters. In *Proc. of the 5th annual ACM Symp. on Principles of distributed computing*, PODC '86, pages 52–62, 1986.

[8] J.M. Bohli, J. Müller-Quade, and S. Röhrich. Bingo voting: Secure and coercion-free voting using a trusted random number generator. *E-Voting and Identity*, pages 111–124, 2007.

[9] C. Boyd. A new multiple key cipher and an improved voting scheme. In *Proc. of the workshop on the theory and application of cryptographic techniques on Advances in cryptology*, EUROCRYPT '89, pages 617–625, 1990.

[10] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Comm. of the ACM*, 24(2):84–90, Feb. 1981.

[11] D. Chaum. Elections with unconditionally-secret ballots and disruption equivalent to breaking rsa. In *Lecture Notes in Comp. Science on Advances in Cryptology-EUROCRYPT'88*, pages 177–182, 1988.

[12] D. Chaum. Secret-ballot receipts: True voter-verifiable elections. *Security Privacy, IEEE*, 2(1):38 – 47, jan.-feb. 2004.

[13] D. Chaum, R. Carback, J. Clark, A. Essex, S. Popoveniuc, R.L. Rivest, P.Y.A. Ryan, E. Shen, and A.T. Sherman. Scantegrity ii: End-to-end verifiability for optical scan election systems using invisible ink confirmation codes. *EVT*, 8:1–13, 2008.

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

[15] D. Chaum, P. Ryan, and S. Schneider. A practical voter-verifiable election scheme. In *Proc. of the 10th European Conf. on Research in Comp. Security*, ESORICS'05, pages 118–139, Milan, Italy, 2005.

[16] B. Chevallier-Mames, P. A. Fouque, D. Pointcheval, J. Stern, and J. Traoré. Towards trustworthy elections. chapter on some incompatible properties of voting schemes, pages 191–199. 2010.

[17] S. Chow, J. Liu, and D. Wong. Robust receipt-free election system with ballot secrecy and verifiability. In *Proc. of Network and IT Conf., NDSS*, 2008.

[18] M. Clarkson, S. Chong, and A. Myers. Civitas: Toward a secure voting system. *Technical Report 2007-2081*, May 2008. Cornell University Computing and Info. Science.

[19] M. Clarkson, S. Chong, and A. Myers. Civitas: Toward a secure voting system. In *Proc. of the 2008 IEEE Symp. on Security and Privacy*, pages 354–368, 2008.

[20] R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *Proc. of the 16th annual Int. Conf. on Theory and App. of cryptographic techniques*, EUROCRYPT'97, pages 103–118, 1997.

[21] S. Dawkins, T. Sullivan, G. Rogers, E. Vincent Cross II, L. Hamilton, and J. E. Gilbert. Prime iii: an innovative electronic voting interface. In *IUI*, pages 485–486, 2009.

[22] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644 – 654, Nov 1976.

[23] J. Epstein. Electronic voting. *IEEE Computer*, (40):92–95, 2007.

[24] K. Fisher, R. Carback, and A.T. Sherman. Punchscan: Introduction and system definition of a high-integrity election system. In *Proceedings of Workshop on Trustworthy Elections*, 2006.

[25] L. Fouard, M. Duclos, and P. Lafourcade. Survey on electronic voting schemes. http://www-verimag.imag.fr/ duclos/paper/e-vote.pdf.

[26] G. S. Grewal, M. D. Ryan, S. Bursuc, and P. Y. Ryan. Caveat coercitor: coercion-evidence in electronic voting. In *IEEE Security and Privacy Symposium*, 2013.

[27] M. Hirt and K. Sako. Efficient receipt-free voting based on homomorphic encryption. In *Proc. of the 19th Int. Conf. on Theory and App. of cryptographic techniques*, pages 539–556, 2000.

[28] J. Howlader, V. Nair, S. Basu, and A. Mal. Uncoercibility in e-voting and e-auctioning mechanisms using deniable encryptiony. *International Journal of Network Security and Its Applications*, 3(2):97–109, 2011.

[29] S. Iftene. General secret sharing based on the chinese remainder theorem with applications in e-voting. *Electron. Notes Theor. Comp. Sci.*, 186:67–84, Jul. 2007.

[30] Douglas Jones and Barbara Simons. *Broken Ballots: Will Your Vote Count?* The University of Chicago Press, June 2012.

[31] A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant electronic elections. In *Proceedings of ACM workshop on Privacy in the electronic society*, WPES '05, pages 61–70, New York, NY, USA, 2005. ACM.

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

[33] K. Kim, J. Kim, B. Lee, and G. Ahn. Experimental design of worldwide internet voting system using pki. In *Proc. of SSGRR Int. Conf. on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet*, SSGRR2001, 2001.

[34] R. Kusters, T. Truderung, and A. Vogt. Clash attacks on the verifiability of e-voting systems. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 395 –409, may 2012.

[35] B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang, and S. Yoo. Providing receipt-freeness in mixnet-based voting protocols. In *Proc. of Info. Security and Cryptology*, volume 2971, pages 245–258, 2003.

[36] B. Lee and K. Kim. Receipt-free electronic voting through collaboration of voter and honest verifier. In *Proc. of JW-ISC2000*, pages 101–108, 2000.

[37] C. Lee, T. Chen, S. Lin, and M. Hwang. A new proxy electronic voting scheme based on proxy signatures. In James J. (Jong Hyuk) Park, Victor C.M. Leung, Cho-Li Wang, and Taeshik Shon, editors, *Future Information Technology, Application, and Service*, volume 164 of *Lecture Notes in Electrical Engineering*, pages 3–12. Springer Netherlands, 2012.

[38] C. Li and M. Hwang. Security enhancement of Chang-Lee anonymous e-voting scheme. *Int. Jour. of Smart Home*, 6(2):45–52, 2012.

[39] C. Li, M. Hwang, and Y. Lai. A verifiable electronic voting scheme over the internet. In *Information Technology: New Generations, 2009. ITNG '09. Sixth International Conference on*, pages 449 –454, april 2009.

[40] T. Moran and M. Naor. Split-ballot voting: Everlasting privacy with distributed trust. *ACM Trans. Inf. Syst. Security*, 13(2):16:1–16:43, 2010.

[41] M. Ohkubo, F. Miura, M. Abe, A. Fujioka, and T. Okamoto. An improvement on a practical secret voting scheme. In *Proc. of the Second Int. Workshop on Info. Security*, ISW '99, pages 225–234, London, UK, 1999.

[42] M. Radwin. An untraceable, universally verifiable voting scheme. *Seminar in Cryptology*, 1995.

[43] R.L. Rivest. The threeballot voting system, 2006. *Avaialable at: http://theory. lcs. mit. edu/rivest/Rivest-TheThreeBallotVotingSystem. pdf*, 2006.

[44] R.L. Rivest and W.D. Smith. Three voting protocols: Threeballot, vav, and twin. In *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, volume 16. USENIX Association, 2007.

[45] Z. Rjaskova. *Electronic Voting Schemes*. PhD thesis, Comenius University, 2002.

[46] K. Sako and J. Kilian. Receipt-free mix-type voting scheme. In *Advances in Cryptology-UROCRYPT'95*, pages 393–403. Springer, 1995.

[47] S. Samet and A. Miri. Privacy preserving ID3 using Gini index over horizontally partitioned data. In *Proc. of the 2008 IEEE/ACS*, AICCSA '08, pages 645–651, Washington, DC, USA, 2008.

[48] D. Sandler, K. Derr, and D.S. Wallach. Votebox: a tamper-evident, verifiable electronic voting system. In *USENIX Security Symposium*, volume 4, pages 349–364, 2008.

[49] B. Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic. In *Proc. of the 19th Annual Int. Cryptology Conf. on Advances in Cryptology*, CRYPTO '99, pages 148–164, London, UK, 1999.

[50] SCYTL. Pnyx.vm: Auditability and voter-verifiability for electronic voting terminals. In *White Paper*, 2004.

[51] A. Shamir. How to share a secret. *Comm. of the ACM*, 22(11):612–613, November 1979.

[52] F. Shirazi, S. Neumann, I. Ciolacu, and M. Volkamer. Robust electronic voting: Introducing robustness in civitas. In *2011 Int. Workshop on REVOTE*, pages 47 –55, Aug. 2011.

[53] O. Spycher, R. Koenig, R. Haenni, and M. Schlpfer. A new approach towards coercion-resistant remote e-voting in linear time. In George Danezis, editor, *Financial Cryptography and Data Security*, volume 7035 of *Lecture Notes in Computer Science*, pages 182–189. Springer Berlin / Heidelberg, 2012.

[54] CACM Staff. Seven principles for secure e-voting. *Commun. ACM*, 52(2):8–9, February 2009.

[55] M. Volkamer and R. Grimm. Determine the resilience of evaluated internet voting systems. In *First Int. Workshop on REVOTE*, pages 47 –54, Aug. 2009.

[56] S. Weber. A coercion-resistant cryptographic voting protocol - evaluation and prototype implementation. *Master's thesis, Darmstadt University of Technology*, 2006.

[57] Wikipedia. Vote counting system. 2012. [Online; accessed 21-Jun.-2012].

[58] X. Zhao, L. Li, G. Xue, and G. Silva. Efficient anonymous message submission. In *IEEE Int. Conf. on Comp. Comm., INFOCOM'12*, volume 2012, pages 2228–2236, May 2012.