# An Efficient E2E Verifiable E-voting System without Setup Assumptions

Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang

**Abstract**—End-to-end (E2E) verifiability has been widely identified as a critical property for the adoption of e-voting systems in real world election procedures. In this work, we present a new e-voting system that is E2E verifiable without any additional "setup" assumption or access to a random oracle. Previously known E2E verifiable e-voting systems required such additional assumptions (specifically, either the existence of a "randomness beacon" or were only shown secure in the random oracle model). Furthermore, our scheme only employs conventional cryptographic building blocks, such as the ElGamal encryption, and it does not require any cryptographic functionality at the client-side to cast a vote. The E2E verifiability property of our scheme is guaranteed information theoretically given the existence of a consistent bulletin board, and the voter privacy is achieved under the well-known DDH assumption.

**Index Terms**—End-to-end verifiability, internet voting, e-voting, election integrity, security modeling

## 1 Introduction

In an end-to-end (E2E) verifiable election system, voters have the ability to verify that their vote was properly cast, recorded and tallied into the election result. Intuitively, the security property that an E2E verifiable election intends to capture is the ability of the voters to detect a malicious election authority that tries to misrepresent the election outcome.

E2E verifiability mandates that the voter can obtain a *receipt* at the end of the ballot casting procedure that can allow her to verify that her vote was (i) cast as intended, (ii) recorded as cast, and (iii) tallied as recorded. Furthermore, any external third party should be able to verify that the election procedure was executed properly. In fact, it is imperative that the receipts in an E2E system are *delegatable* i.e., the voter may outsource the task of verifiability to any interested third party, for instance an international organization that aggregatively performs verification and is trusted by some voters to do so. This requirement, as well as the fact that it should be infeasible for the voter to use her receipt as a proof of the way she voted (this is necessary to deter vote-selling/buying), make the design of E2E verifiable systems a challenging problem.

All known e-voting systems that offer E2E verifiability are able to support it only under some "setup" assumption, such as (i) the existence of a trusted party that provides a stream of unbiased and unpredictable random coins (a randomness beacon) or a trustily generated common reference string (CRS), or (ii) in the random oracle (RO) model. Notably, E2E verifiability can be argued, yet never formally proven, for Helios [2] in the RO model while for Remotegrity/Scantegrity II [3], [4] under the presence of a randomness beacon. More general approaches for defining auditable multiparty computation have recently been proposed [5] and also rely on a setup assumption such as a CRS.

A limitation of using setup assumptions for establishing E2E verifiability in e-voting is the fact that a leap of faith will be required in order to accept the setup assumption and thus the election result. This can be an unfortunate state of affairs: since the election authority (EA) cannot unequivocally convince the voters that the election is correct, then the election outcome can be always subject to dispute.

**Our contributions.** Motivated by the above, we design a new e-voting system that we can prove E2E verifiable *information theoretically without any setup assumption* except the existence of a bulletin board (BB) which provides a global consistent view of the election. The requirement for BB consistency can be seen as a tight condition since without it, it is easy to see that E2E verifiability of the election cannot be achieved: by controlling the BB, an adversarial EA can distribute voters to their own separate "islands" where within each one the voters will have their own verifiable view of an election result that can be - in reality - completely skewed. The latter is in agreement with the compromise for tolerating the possibility of *independent execution attacks* in networks without any authentication mechanism discussed by Barak et al. [6].

Our result is further strengthened by the fact that we make the absolute minimal assumptions on the computation capabilities of the voters: no cryptographic operations at the voter side during ballot-casting. (Note the auditing stage after the election would require the capability of

- *Aggelos Kiayias and Thomas Zacharias are with the Department of Informatics, The University of Edinburgh, UK. E-mail: Aggelos.Kiayias@ed.ac.uk and tzachari@inf.ed.ac.uk.*
- *Bingsheng Zhang is the corresponding author, and is with School of Computing and Communications, Lancaster University, UK. E-mail: b.zhang2@lancaster.ac.uk*

*A preliminary version [1] of this paper was presented at the 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2015). This research was partly supported by ERC project #259152 (CODAMODA), Horizon 2020 project #653497 (PANORAMIX), and project FINER, Greek Secretariat of Research and Technology, funded under action ARISTEIA 1.*

cryptographic operations but they are optional and can be performed at any time, in the post-election stage).

Our construction cherry picks ideas drawn from previous works, specifically, code-voting and double ballots from [7], and secret-sharing homomorphisms from [8], but also introduces a number of novel elements that enable us to prove E2E verifiability. In order to achieve verifiability, our system collects coins from the voters to form the challenge (specifically, a single random coin per voter). Given that a certain proportion of voters are honest and properly follow the protocol, the sequence of voter contributed randomness has sufficient entropy to be used as the challenge of zero-knowledge protocols. applying these techniques, we prove information theoretically the E2E verifiability for our scheme. For voter privacy, we utilize complexity leveraging to show that in the underlying commitment scheme is hiding then our system offers voter privacy.

In summary, our e-voting system is the first construction achieving E2E verifiability and voter privacy without any setup assumption assuming a consistent BB, which can be seen as a minimal security assumption. Furthermore, we propose a new formal cryptographic security framework. Compared with [1], we simplify the E2E verifiability definition. Under the new security framework, we prove E2E verifiability information theoretically, while we assume the hardness of a well-studied cryptographic problem (Decisional Diffie-Hellman (DDH) assumption) for voter privacy. Finally, our system does not require that the voters perform cryptographic operations at their engagement in the voting procedure.

**Summary of existing techniques for verifiability.** To motivate further our approach it is worth-while to emphasize in which way previous works fail to attain E2E verifiability without any setup assumption. Helios, culminates a long line of previous schemes that employ homomorphic type of voting [9], [10] and utilizes the Benaloh challenge [11] as the fundamental mechanism to attain verifiability. Helios by design requires the voter to utilize a voter supporting device to prepare a ciphertext and after an indeterminate number of trials, the voter will cast the produced ciphertext. Such ciphertexts are to be homomorphically tallied and thus they should be accompanied by a proof of proper computation. While such proofs can only be proved interactively (which is insufficient in our setting since a corrupt EA together with a corrupt voter may cook up a malformed proof that is indistinguishable from a proper one). Instead, Helios adopts a RO-based proof, thus imposing the RO restriction on the adversary during an E2E verifiability attack. On the other hand, in the case of Remotegrity/Scantegrity the required randomness needs to be obtained from a randomness beacon in order to prove the result correct. It is easy to verify that the system is insecure in terms of E2E verifiability in case the randomness beacon is biased. As before, the only parties active are the EA and the voters who cannot implement the randomness beacon required in the construction.

In light of the above, our construction offers a new paradigm in e-voting design: the randomness for the verification of the election can be collected distributively from the entropy generated by the voters' interaction with the system. This entropy is *internal* with respect to the election environment, therefore the need for trusting an outer source of randomness or restricting the adversary in the RO sense is eliminated.

## 2 BUILDING BLOCKS

In this section, we describe the election notions and functionalities that our system realizes with the use of cryptographic tools. Similar building blocks can be found in other noticeable systems, e.g. [2], [12]. In our text, we use the term *negligible* to denote that a value gradually becomes very small with respect to (w.r.t.) the increase of system's parameters. Formally, negligible refers to a real function $f(n) : \mathbb{N} \mapsto \mathbb{R}$ over the integers that is asymptotically smaller than the inverse of any polynomial in $n$.

### 2.1 Implementing an electronic envelope

To enable the correct counting of the votes while preserving privacy, we deploy a *commitment scheme* to realize the concept of an "electronic envelope" that is (i) *binding*, in the sense that an adversary cannot open an envelope to a value different than the originally enclosed message and (ii) *hiding*, i.e. an adversary obtains no information about the enclosed message until the envelope is opened. A generic commitment scheme consists of the following phases and algorithms:

Key generation: is an algorithm that on input some security parameter $\lambda$, it outputs a commitment key $\mathsf{ck}$.

Commit phase: given $\mathsf{ck}$, one can enclose a message $\mathrm{M}$ in a digital envelope, by generating a commitment $\mathsf{Com}_{\mathsf{ck}}(\mathrm{M})$ for $\mathrm{M}$.

Opening phase: the commitment $\mathsf{Com}_{\mathsf{ck}}(\mathrm{M})$ can be disclosed given $\mathsf{ck}$ and opening data $(\mathrm{M}, r)$, so that one can verify the correct opening of $\mathsf{Com}_{\mathsf{ck}}(\mathrm{M})$.In our system we utilize lifted ElGamal encryption to instantiate a commitment scheme. Hence, the commitment scheme inherits the security properties of the ElGamal cryptosystem. Namely, the perfect correctness of ElGamal decryption implies the perfect binding property against adversaries of unlimited computational power. Moreover, the semantic security of ElGamal implies the hiding property against computationally bounded adversaries assuming the hardness of the Decisional Diffie-Hellman (DDH) problem DDH problem for the underlying encryption group.

### 2.2 Generating electronic envelopes for valid votes

In our system, the messages that need to be enclosed under the ElGamal commitment scheme are valid encodings of a candidate from a list $P_1, \ldots, P_m$. We encode candidate $P_j$ as the unit vector $e_j \in \mathbb{Z}_m$, where the $j$-th position is 1 and all the rest $m-1$ positions are 0. For example, in the case of three candidates $P_1, P_2, P_3$ the candidate encodings are $(1,0,0), (0,1,0), (0,0,1)$, respectively. A commitment to $e_j$ is vector of $m$ commitments in one-to-one correspondence with the components of $e_j$. In the rest of the paper, we will often refer to *vector commitments* to denote a component-wise vector of commitments to a vector of messages.

### 2.3 Facilitating secure vote counting

Besides binding and hiding, the lifted ElGamal commitment scheme is *additively homomorphic*. Namely, multiplying two commitments for messages $\mathrm{M}_1, \mathrm{M}_2$ under the same commitment key $\mathsf{ck}$ produces a commitment for $\mathrm{M}_1 + \mathrm{M}_2$ as shown below:

$$\boxed{\mathsf{Com}_{\mathsf{ck}}(\mathrm{M}_1) \cdot \mathsf{Com}_{\mathsf{ck}}(\mathrm{M}_2) = \mathsf{Com}_{\mathsf{ck}}(\mathrm{M}_1 + \mathrm{M}_2)\ .}$$

In addition if $(\mathrm{M}_1, r_1)$ and $(\mathrm{M}_2, r_2)$ are the opening data for $\mathsf{Com}_{\mathsf{ck}}(\mathrm{M}_1)$ and $\mathsf{Com}_{\mathsf{ck}}(\mathrm{M}_2)$ respectively, then the opening data for the homomorphically added commitment $\mathsf{Com}_{\mathsf{ck}}(\mathrm{M}_1 + \mathrm{M}_2)$ are $(\mathrm{M}_1 + \mathrm{M}_2, r_1 + r_2)$.

Homomorphic additivity naturally extends to multiple vector commitments. Hence, it allows for the tally of all the votes enclosed in the "electronic envelopes" as if these were added within a large envelope. The latter denotes the *homomorphic product* vector commitment consisting of $m$ ElGamal commitments to the components of the tally vector $T := (t_1, \dots, t_m)$, where $t_j$ represents the number of votes for the $j$-th candidate. In turn, the large envelope can be opened to $T$. The tally is performed without revealing the individual votes that summed to it.

### 2.4 Proving ballot validity

As part of the verification mechanism of our system, it is checked whether the envelopes (commitments of the votes) included in the tally are well-formed, i.e. they enclose the valid encoding of some candidate to a unit vector. This is done by utilizing a special case of interactive zero-knowledge proofs, called *$\Sigma$-protocols*. In a $\Sigma$-protocol, a *prover* $\mathsf{Prv}$ with some private input $w$ interacts with a *verifier* $\mathsf{Vrf}$ to convince it of the validity of some statement $x$. More formally, $\mathsf{Prv}$ and $\mathsf{Vrf}$ are modeled as *interactive Turing Machines*, where $\mathsf{Prv}$ is unbounded and $\mathsf{Vrf}$ runs in polynomial time. The statement validity is expressed as $x$ belonging in some $\mathbf{NP}$ language $\mathcal{L}$ and $\mathsf{Prv}$'s private input is typically a witness $w$ for $x$. The interaction is in three moves; in the first move, $\mathsf{Prv}$ sends to $\mathsf{Vrf}$ a *commitment* message $\Sigma^{(1)}$. In the second move $\mathsf{Vrf}$ provides $\mathsf{Prv}$ with a *challenge* $\rho$ and in the third move, $\mathsf{Prv}$ replies with a *response* $\Sigma^{(2)}$. Given its whole view of the protocol, $\mathsf{Vrf}$ must output an accept or reject message.

A $\Sigma$-protocol satisfies the following three properties, here stated informally:

1) **Completeness:** $\mathsf{Vrf}$ always accepts the proof of $\mathsf{Prv}$ for every valid statement $x \in \mathcal{L}$.
2) **Soundness:** $\mathsf{Vrf}$ does not accept the proof of a potentially malicious prover $\mathsf{Prv}^*$ for some invalid statement $x \notin \mathcal{L}$, except from some negligible soundness error probability $\epsilon$.
3) **Honest verifier zero-knowledge (HVZK):** There exists an efficient simulator, s.t. for every $x \in \mathcal{L}$, can simulate a valid transcript $(\Sigma^{(1)}, \rho, \Sigma^{(2)})$ without accessing to the witness.

In our system, we adopt *disjunctive Chaum-Pedersen (CP) proofs* [13] which are $\Sigma$-protocols for proving that an ElGamal commitment commits to a unit vector. Specifically, for every vector commitment $C$, we show that it commits to a valid encoded vote as follows: for each of the $m$ components of $C$, we generate a CP proof that the component commits to a bit value. Then, applying the additive homomorphic property we generate a CP proof that the products of the components is a commitment of 1.

In the following proposition, we state the properties of a CP proof, which can be seen as an adaptation of [1, Theorem 1] to the CP proof setting.

**Proposition 1.** *A Chaum-Pedersen proof achieves completeness, soundness with probability error $2^{-\kappa}$ and HVZK, as long as the min-entropy of the challenge string provided by the verifier* $\mathsf{Vrf}$ *is at least $\kappa$ bits.*

## 3 Syntax and Threat Model

We build upon the syntax and threat model introduced in [1]. We use $\lambda$ as the security parameter. We denote by $\mathcal{V} = \{V_1, \dots, V_n\}$ and $\mathcal{P} = \{P_1, \dots, P_m\}$ the set of voters and candidates respectively, which sizes $n$ and $m$ are polynomial in $\lambda$.

We stress that for simplicity, we consider an abstract centralized election authority (EA) that administers the setup, vote casting and tally phases of the election. This is consistent with our definition of E2E verifiability in an all-malicious setting, where the EA is adversarially controlled.

Regarding vote privacy, we will require the EA to be honest, as it is fully aware of the encoding of the voters' candidate selections. In Sect. 6.1 we show how to distributing the EA to set of trustees.

### 3.1 Syntax

The entities involved in an e-voting system are as follows:

- The *election authority* (EA) that manages the entire election; in particular, EA prepares all the election information, distributes the voters' ballots, collects the votes and is responsible for computing the tally and announcing the election result.
- The *voters* $\mathcal{V} = \{V_1, \dots, V_n\}$, possibly equipped with *voting supporting devices (VSDs)* and *auditing supporting devices (ASDs)*.
- A publicly accessible and consistent *bulletin board* (BB) where the election result and all audit information is posted.

An e-voting comprises four phases: 1) election setup, 2) vote casting, 3) election tally, and 4) auditing. The interaction among e-voting entities at the four phases is illustrated in Figure 1.

### 3.2 Modeling E2E verifiability

The adversarial goal for an attacker against system's integrity is to cause deviation from the intended tally while election auditing remains successful. Given that the candidates are encoded as unit vectors and the tally is represented as a vector in $\mathbb{Z}^m$, we measure tally deviation via the $\mathrm{d}_1$ metric derived by the $\ell_1$-norm, $\|\cdot\|_1$ scaled to half, i.e.,

$$\begin{array}{llll} \mathrm{d}_1 : & \mathbb{Z}_+^m \times \mathbb{Z}_+^m & \longrightarrow \mathbb{R} \\ & (R, R') & \longmapsto \frac{1}{2} \cdot \sum_{i=1}^n |R_i - R'_i| \end{array}$$

where $R_i, R'_i$ is the $i$-th coordinate of $R, R'$ respectively. We consider an adversary that controls the entire election

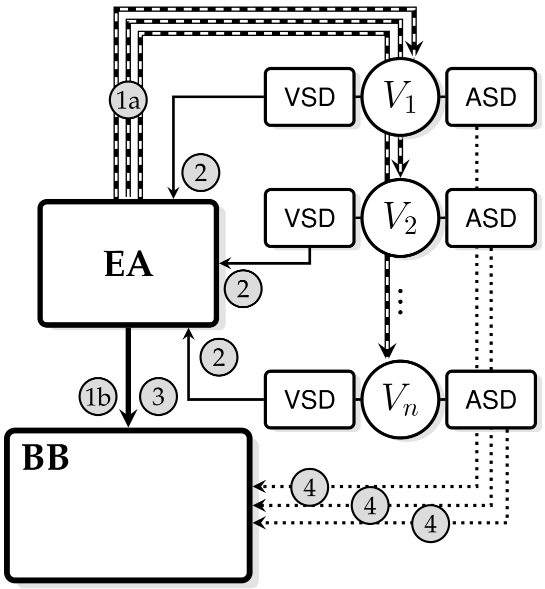

FIGURE 1: The interaction among the entities in an e-voting execution. The dotted lines denote read-only access to the BB. The dashed arrows denote untappable channels for voters' private inputs distribution. **Annotation:** (1a): distribution of voter's private inputs; (1b): pre-election BB data; (2): vote casting; (3): post-election BB data; (4): auditing.

by corrupting the EA and all the VSDs. In addition, it can control up to a number of voters and their devices. Our E2E verifiability threat model is presented in Fig. 2.

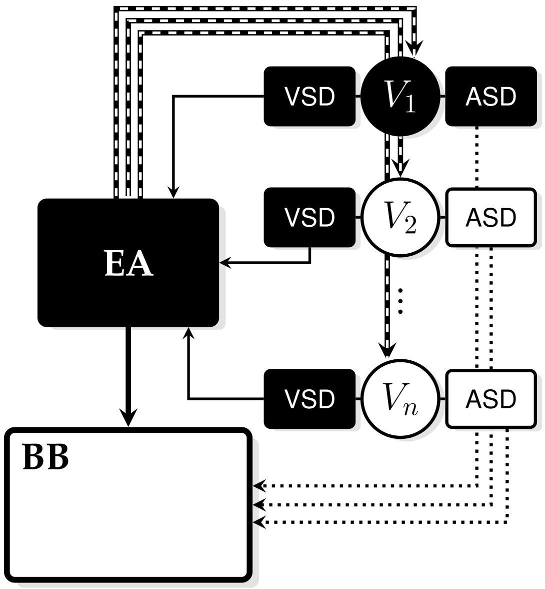

FIGURE 2: An adversary against E2E verifiability that controls voter $V_1$. The corrupted nodes are colored in black.

We provide the following definitions:

**Definition 1.** Let $R \in \mathbb{Z}^m$ be the announced result in an election run by an E2E verifiability adversary $\mathcal{A}$. Let $R_h \in \mathbb{Z}^m$ be the partial result derived by the intended votes of the honest voters, i.e. how the honest voters wanted their votes to be counted. We say that $\mathcal{A}$ *causes tally deviation* $\delta$, if for every possible partial adversarial result $R^* \in \mathbb{Z}^m$, it holds that

$$\mathrm{d}_1(R_h + R^*, R) \geq \delta\ .$$

Namely, it is impossible to explain the adversarial votes, so they produce a tally within a radius $\delta$ from $R$.

To provide intuition on Definition 1, consider an election with voters $V_1, V_2, V_3$ and candidates $P_3$, where voter $V_1$ and $V_2$ vote for $P_1$ and $V_3$ is corrupted. Hence, $R_h = (1,0,0) + (1,0,0) = (2,0,0)$ while the possible adversarial results are $(1,0,0), (0,1,0), (0,0,1)$ and the non-vote vector $(0,0,0)$. If $\mathcal{A}$ announces the result $(1,1,1)$, then we have the following four cases:

$$\begin{aligned}
\mathrm{d}_1\big((2,0,0) + (1,0,0), (1,1,1)\big) &= 2\\
\mathrm{d}_1\big((2,0,0) + (0,1,0), (1,1,1)\big) &= 1\\
\mathrm{d}_1\big((2,0,0) + (0,0,1), (1,1,1)\big) &= 1\\
\mathrm{d}_1\big((2,0,0) + (0,0,0), (1,1,1)\big) &= 3/2
\end{aligned}$$

Hence for every possible partial adversarial result $R^* \in \mathbb{Z}^m$, it holds that $\mathrm{d}_1(R_h + R^*, R) \geq 1$, which implies tally deviation 1 (switching one vote).

**Definition 2.** We say that an *e-voting system achieves E2E verifiability with error* $\epsilon$ for at least $\theta$ honest voters and tally deviation $\delta$, if the probability that an adversary $\mathcal{A}$ causes tally deviation $\delta$ while all $\theta$ honest voters verify successfully, is no more than $\epsilon$.

**Remark 1.** We simplify the original definition in [1] by not including an unbounded *vote extractor* algorithm $\mathcal{E}$ that explains the adversarial tally by brute-force decrypting the corrupted voters' ballots and deviation would be defined w.r.t. $\mathcal{E}$'s output. The reason is that the vote extractor approach and Definition 2 can be easily shown equivalent against unbounded adversaries, which is the case we are interested in the paper.

The proof is by contradiction; at a high level, if Definition 2 fails, then there is an adversary $\mathcal{A}$ that causes tally deviation $\delta$ by announcing the result $R$ according to Definition 1. Thus, for every adversarial partial tally, denoted by $R^{\mathcal{E}}$, that an extractor $\mathcal{E}$ might output, it holds that $\mathrm{d}_1(R_h + R^{\mathcal{E}}, R) \geq \delta$.

Conversely, consider the optimal extractor $\mathcal{E}^*$ that outputs an adversarial result $R^*$ that minimizes $\{R' \mid \mathrm{d}_1(R_h + R', R)\}$. Given that the E2E verifiability definition in [1] fails, there is an adversary $\mathcal{A}^*$ that wins $\mathcal{E}^*$, hence it holds that $\mathrm{d}_1(R_h + R^*, R) \geq \delta$. Consequently, the adversary $\mathcal{A}$ causes tally deviation $\delta$ and Definition 2 does not hold.[1]

## 3.3 Modeling voter privacy

As in [1], we model voter privacy by also capturing the property of coercion resistance against adversaries that passively monitor the voting procedure and request the voters' transcripts after vote casting, including their receipts. Such an adversary can corrupt all the voters' devices and schedule network traffic. It may also control up to a number of voters. However, the voters' private inputs are delivered

1. We acknowledge an anonymous reviewer of an IACR conference for helpful remarks regarding one direction of this equivalence statement.

by the EA via *untappable channels*. Our voter privacy threat model is presented in Fig. 3.

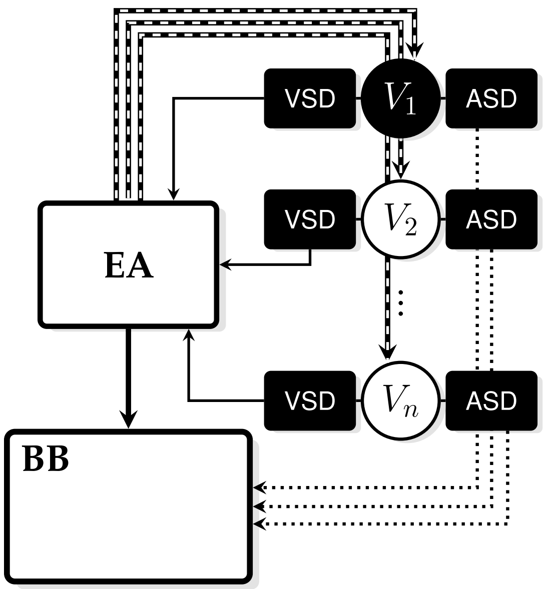

FIGURE 3: An adversary against voter privacy that controls voter $V_1$. The corrupted nodes are colored in black. The dashed arrows denote untappable channels for voters' private inputs distribution.

Recall that the adversary can also obtain the honest voter's transcript and receipt. In order to deceive the adversary on the way they voted, the honest voters must be able to lie about their interaction with the system by providing fake views that are indistinguishable from the actual ones.

**Definition 3.** We say that an adversary $\mathcal{A}$ *has negligible advantage against privacy* if for an election, it has only negligible additional probability to output the correct honest voter's votes compared with the trivial advantage.

For the better understanding of Definition 3, consider the previous example where three honest voters $V_1, V_2, V_3$ vote for $P_1, P_1, P_3$ and produce the tally $(2, 0, 1)$. The same tally would derive if $V_1, V_2, V_3$ had voted for $P_3, P_1, P_1$ or $P_1, P_3, P_1$, i.e. there are three ways in total. Then, $\mathcal{A}$ has negligible advantage against privacy if the probability of outputting the votes of $V_1, V_2, V_3$ is no more than $1/3 + \epsilon$, where the $1/3$ is the probability of random guessing the votes and $\epsilon$ is a negligible term.

**Definition 4.** We say that an *e-voting system achieves voter privacy* for at most $t$ corrupted voters, if any probabilistic polynomial time adversary $\mathcal{A}$ that controls up to $t$ voters has negligible advantage against privacy.

## 4 System Description

In this section, we will describe an e-voting system for 1-out-of-$m$ type of elections.

### 4.1 Data structure

Prior to the election, the voters shall already receive a voting card via a secure untappable channel. Fig. 4 illustrates an example of the basic structure of such a voting card. Each voting card has a serial number (SN) and consists of two functionally equivalent parts: Side-A and Side-B. Each part contains an unpredictable authentication code. Besides, it lists $m$ candidates together with their corresponding vote-code.

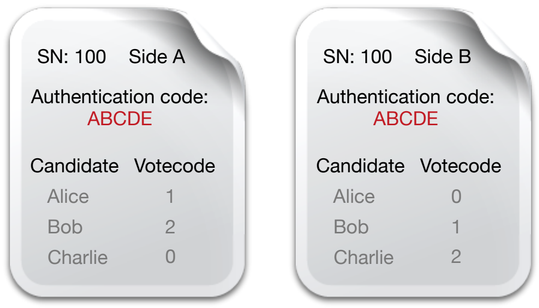

FIGURE 4: Voting card.

To enable verifiability, the EA needs to post necessary information regarding the election process on the BB. As depicted in Fig. 5, there are 7 columns and 2 rows for each ballot. Column I states a unique serial number, and column II shows Side-A and Side-B. Column III is used for vector commitments (cf. Section 2.2, above). Columns IV and V are used for the first move and the third move of the corresponding CP proofs for the validity of the vector commitments. Column VI is used to reveal some of the openings of the vector commitments in Column III. Finally, Column VII is used to mark the voters' submitted votecodes. Notice that Columns V, VI, VII are only partially filled, and this will be clarified in the due course.

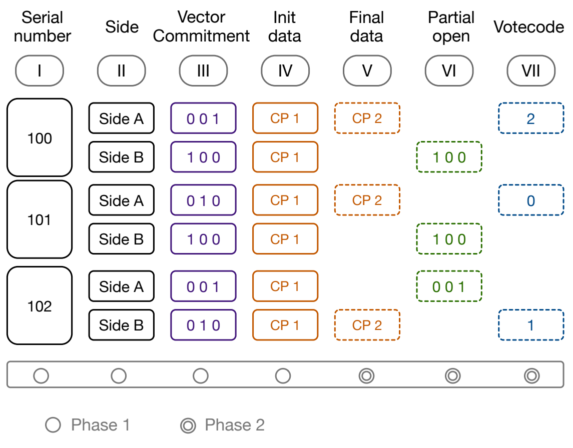

FIGURE 5: BB structure.

### 4.2 Election setup

The EA first generates a commitment key ck and posts ck along with the other public election information, e.g., election question, to the BB. For $\ell \in \{1, \ldots, n\}$, the EA then generates the meta-data for ballot $\mathcal{B}_\ell$ as follows. For $x \in \{\text{Side-A}, \text{Side-B}\}$, the EA picks a random number $r_{\ell,x} \leftarrow \mathbb{Z}_m$; the EA then commits an $m$-ary unit vector

$e_{r_{\ell,x}}$, where the $r_{\ell,x}$-th coordinate is 1 and the rest coordinates are 0. Denote this vector commitment as $C_{\ell,x}$. After that, the EA computes the first move of the CP proofs, denoted as $\Sigma^{(1)}_{\ell,x}$, showing that the vector commitment $C_{\ell,x}$ indeed contains a unit vector. (cf. Section 2.4, above.) For $j \in \{1, \ldots, m\}$, the EA sets the votecode of the $j$-th candidate as $O_{\ell,x,j} = j - r_{\ell,x} - 1 \pmod{m}$; namely, the votecode is the offset of the $j$-th candidate by $r_{\ell,x} - 1$. (Note that the votecodes in each side of a ballot is essentially a cyclic shift of the canonical ordering, i.e., $0, 1, \ldots, m$). Finally, for each ballot $\mathcal{B}_\ell$, the EA generates a random authentication code and produces the corresponding voting card $\mathcal{C}_\ell$ as in Fig. 4. Meanwhile, the EA posts information about Columns I, II, III, IV to the BB as depicted in Fig. 5 (phase 1), and it also keeps its state. After election setup, the voting cards are distributed to the voters via some secure untappable channels, e.g. postal service.

### 4.3 Vote casting

Having obtained the voting card, the voter $V_\ell$ can cast her vote once the election starts. To cast a vote, $V_\ell$ randomly selects a side to use, say Side-A. She looks up the votecode of her intended candidate in $A$ side of the voting card, say 2. She then inputs her authentication code to a VSD; after authentication, she utilizes the VSD to send the side and votecode, A-2 to the EA. At the end of vote casting, $V_\ell$ keeps the unused side, i.e., Side-B of her voting card and the submitted message, A-2 as the receipt, denoted as $\alpha_\ell$. (cf. Fig. 6, below.)

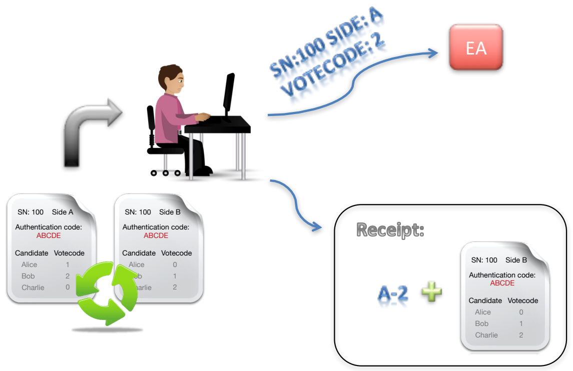

FIGURE 6: Vote Casting.

### 4.4 Election tally

After the election ends, the EA marks Column VII on the BB according to the received votes. For $\ell \in \{1, \ldots, n\}$, define the voter's coin $b_\ell = 0$ if Side-A was used and $b_\ell = 1$ otherwise. Consider all the voters' coins $B := (b_1, \ldots, b_n)$ as the $\Sigma$ protocol challenge of the CP proofs. The EA completes the third moves of the CP proofs, denoted as $\Sigma^{(2)}_\ell$, corresponding to the marked vector commitments for the side of the voting card that was used. For the unused side of the voting card, the EA opens all the corresponding vector commitments, denoted as $D_\ell$. The EA posts $\Sigma^{(2)}_\ell$ and $D_\ell$ to the BB.

For each cast ballot $\mathcal{B}_\ell$, the EA performs cyclic right shift on the corresponding vector commitment according to the submitted votecode. Note that the resulting vector commitment, denoted as $C'_\ell$, should only contain 1 at the $j_\ell$-th coordinate, where the $j_\ell$-th candidate is the intended choice of voter $V_\ell$. Let $\mathcal{V}$ be the set of all the voters who cast a vote. For all $V_\ell \in \mathcal{V}$, the EA entry-wise multiplies $C'_\ell$, obtaining a vector commitment of the final tally, denoted as $E$. (Cf. Section 2.3, above.) The EA then posts the decommitment of $E$ to the BB, opening $E$ to the election tally $T := (t_1, \ldots, t_m)$, where $t_j$ represents the number of votes for the $j$-th candidate.

### 4.5 Auditing

The voter $V_\ell$ can use her ASD to audit the election w.r.t. the BB information and her receipt $\alpha_\ell$. More specifically, she checks

- The annotated Column VII content is consistent with the side and votecode she submitted.
- The opened vector commitment in Column VI is consistent with the unused side of her voting card. More precisely, let $u$ be the opening of the vector commitment. If we apply cyclic right shift to $u$ according to the printed votecode (cf. Fig. 4, above) for the $j$-th candidate, then the resulting opening is the unit vector $e_j$.

Moreover, $V_\ell$ or any third-party auditor can use an ASD to check the following on the BB:

- All the CP proofs and commitment openings are valid.
- The announced tally is consistent with the decommitment of $E$.

## 5 SECURITY

### 5.1 E2E verifiability

In this section, we demonstrate the E2E verifiability that our system achieves under the security model in Section 3.2. We stress that our E2E verifiability theorem holds information theoretically in the standard model.

Before stating the theorem, we list the plausible attacks against the verifiability of our system, describing their effectiveness and detection probability at a high level. For simplicity, we exclude all the trivial attacks that the adversary may follow, i.e. the ones that will be detected with certainty (e.g. malformed or unreadable election transcript). Therefore, the meaningful types of attack that an adversary may launch against our system are the following:

- **Invalid encoding attack**: the adversary creates an option-encoding commitment to some invalid value, i.e. a non-unit vector that does not encode a legitimate candidate selection (e.g., multiple votes for some specific candidate). This attack can be prevented by the soundness of the CP protocol, except from the negligible soundness error. The proof verification is done via a trusted ASD.
- **Voting card attack**: the information in an honest voter's voting card side part is inconsistent with the

respective audit data committed in the BB, e.g., the votecode and candidate correspondence is altered. The tally deviation achieved from this type of attack is at most 1, by switching the voter's vote. The probability of detection is $1/2$, as the voter chooses to audit using the inconsistent voting card side with probability $1/2$.

- **Clash attack** [14]: the adversary instructs $y$ honest voters to point at the same BB location, by providing them with voting cards indexed under the same serial number. Thus it creates $y-1$ empty BB location that it can associate them with injected votes its choice. During auditing, all $y$ voters verify the correct counting of their votes by auditing the same information on the BB and hence, miss the injected votes that produce the tally deviation. The deviation achieved by this type of attack is $y-1$, whereas the probability of detection is $1-2^{-(y-1)}$. This is because the attack succeeds only when all $y$ voters choose the same voting card side to vote with, which happens with $2/2^y = 2^{-(y-1)}$ probability.

**Remark 2.** It can be easily shown that the above list exhausts all possible attack strategies against our system in our threat model; in the case where all ballot information is tabulated using all valid commitments in the BB without being deleted or replaced, the adversary can only perform a combination of voting card attacks and clash attacks on the honest votes. If no such combination occurs, then all honestly cast votes are in correct (yet unknown) one-to-one correspondence with the BB audit data, hence by the perfect binding property of the ElGamal commitment scheme, the opening of the homomorphic tally matches the intended result.

**Theorem 1.** *The e-voting system described in Section 4 achieves achieves E2E verifiability for at least $\theta$ honest voters and tally deviation $\delta$, with error $2^{-\theta} + 2^{-\delta}$.*

*Proof:* Let $\mathcal{A}$ be an adversary against the E2E verifiability of our system that allows at least $\theta$ voters to be honest and audit while it causes tally deviation at least $\delta$. Consider the following two cases:

**Case 1.** Assume that $\mathcal{A}$ performs at least one invalid encoding attack. Each of the honest voters contributes 1 bit of entropy by her coin flipping, thus the generated challenge for the CP proofs will have at least $\theta$ bits of min-entropy. By Proposition 1, the soundness error of each CP proof is $2^{-\theta}$, hence the probability that $\mathcal{A}$ succeeds is bounded by $2^{-\theta}$.

**Case 2.** If $\mathcal{A}$ performs no invalid encoding attack. Then, all cast votes are legitimate, therefore the adversarial ones will produce an acceptable partial result $R^* \in \mathbb{Z}^m$ included in the election result $R$ that $\mathcal{A}$ announces. By Definition 1, we have that

$$\mathrm{d}_1(R_h + R^*, R) \geq \delta \ ,$$

where and $R_h$ is partial result derived by the vote intention of the honest voters. Therefore, we have that

$$\mathrm{d}_1(R_h, R - R^*) \geq \delta \ ,$$

where now $R - R^*$ is the partial result produced by homomorphically tallying all the honest ballots that $\mathcal{A}$ (acting as a malicious EA) marked for counting. The latter suggests that at least $\delta$ honest votes where altered via combinations of voting card attacks and clash attacks. In the attacks description, we showed that when performing voting card attacks or clash attacks, the increase of tally deviation by 1 implies a drop of success probability for by $1/2$. Therefore, by combinations of these two attacks, $\mathcal{A}$ has no more than $2^{-\delta}$ probability to achieve tally deviation at least $\delta$.

By taking the disjunction of the two cases, $\mathcal{A}$'s strategy has no more than $2^{-\theta} + 2^{-\delta}$ success probability. ■

## 5.2 Voter privacy

In the following theorem, we show how the voting card format, generation and distribution, as well as the security of the applied cryptographic tools serve to protect the voter privacy of our system.

**Theorem 2.** *Assume there exists a constant $c$, $0 < c < 1$ such that for any adversary $\mathcal{A}$ running in $2^{\lambda^c}$-time cannot break the hiding property of the underlying ElGamal commitment scheme. The e-voting system described in Section 4 achieves privacy for at most $t$ corrupted voters for any $t = \lambda^{c'}$ for any constant $0 < c' < c$.*

*Proof:* Recall that the voting cards are delivered to the voters via some secure untappable channels. Therefore, when the EA is honest, the adversary does not know the link between votecodes and candidates. Hence, the adversary gains no information about honest voters' choices by eavesdropping their cast votecodes.

In addition, the voters are able to lie about their views, consistently with the passive coercion resistance property. Specifically, since the voting cards are not authenticated, they merely display a correspondence between the candidates and integers from 0 to $m-1$ that can be generated and printed by any party. Thus, the voting cards can be faked and provide no proof of how the voter has voted.

As a result, the system's components that the voter privacy adversary is left to attack, is its cryptographic payload, i.e. the ElGamal commitments and the CP proofs. The CP proofs, used for ballot validity, reveal no information about the intended vote associated with the ballot, due to the HVZK property. Besides, the ElGamal commitments, playing the role of electronic envelopes, do not leak information about the enclosed candidate, due to the hiding property. It is easy to show that any adversary $\mathcal{A}$ can break the the hiding assumption of the ElGamal commitment with less than $2^{\lambda^c}$ time if she can break privacy for $t = 2^{\lambda^{c'}}$. ■

## 6 Our system in the real-world

The proposed e-voting system has been fully implemented and applied in the following use cases.

1) In a pilot experiment during the European Elections in May 2014 (747 participants), testing the usability and people's trust in the system.
2) In a pilot experiment during the Greek National Elections in January 2015 (400 participants), studying the effect of E2E verifiability support on people's trust in the system.

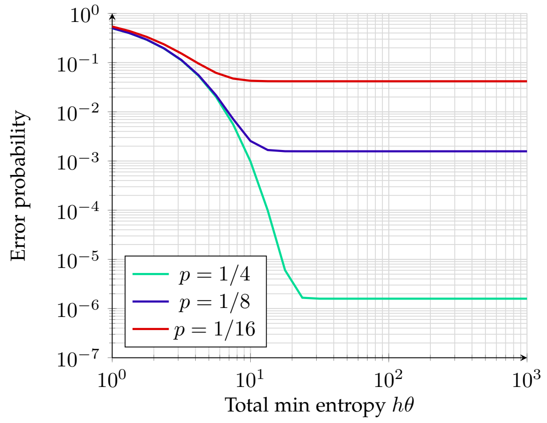

FIGURE 7: E2E verifiability error bound w.r.t. total min entropy $h\theta$, for tally deviation $\delta = 100$ and auditing participation rate $p = 1/4, 1/8, 1/16$.

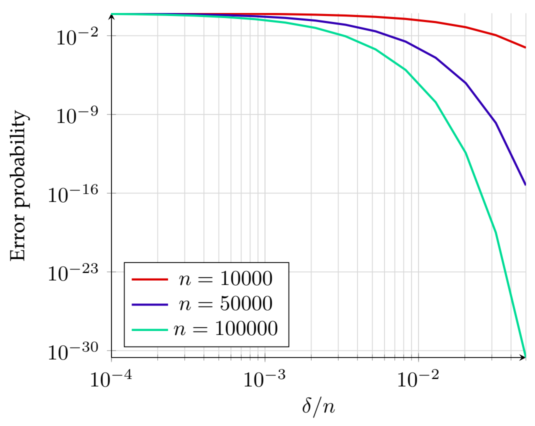

FIGURE 8: E2E verifiability error bound w.r.t. normalized tally deviation $\delta/n$ for at least $n/2$ honest voters with parameters $h = 0.234$, $p = 2.8\%$, and $n = 10000, 50000, 100000$ voters.

3) As the fundamental component of the platform that will support the electronic democratic procedures of the General Confederation of Workers of Greece (GSEE), one of the largest work unions in Greece.

In the following subsections, we provide an overview of the implementation design and comment on the importance of the human factor in the system's security along with real-world evaluation from the two pilot experiments.

## 6.1 Distributing the EA

In our security model, we consider the EA as a single entity that is malicious in the verifiability game and honest in the privacy game. In practice one may want to distribute the EA to a number of "sub-authorities" that collectively implement the EA functionality to improve the resiliency of the privacy property. Our notion of voter privacy can be easily extended to allow corrupted sub-authorities in the same way that the adversary controls corrupted voters. In the real-world implementation of our system, we distribute the EA to a setup authority, a vote collector and a set of trustees $T_1, \ldots, T_k$. During the election setup phase, the setup authority prepares all the ballots and voting cards. After that, the setup authority distribute its distributes its private state as $k$ shares to $T_1, \ldots, T_k$ via a *verifiable secret sharing scheme*. This allows the trustees to jointly open the tally and complete the necessary CP proofs at the end of the election. When setup is finished, the working tape of the setup authority is destroyed and the election run is privacy-preserving, as long as at least one trustee is not corrupted.

Thanks to this design, our systems achieves (a) E2E verifiability in an all-malicious setting and (b) voter privacy as long as at least one trustee remains honest.

## 6.2 Humans as a source of randomness

For simplicity in our security analysis, we assumed that all honest voters that engage in an election run with our system will (i) flip a fair coin and (ii) always verify the election. In this idealized setting, at least $\theta$ honest voters generate at least $\theta$ bits of entropy that implies the $2^{-\theta} + 2^{-\delta}$ error bound in E2E verifibiality proven in Theorem 1. Nonetheless, in a real-world execution one can not expect such an idealized human behavior. To approximate an actual election setting, we have to made the following relaxations:

1) The voters flip biased coins that have min entropy at least $h \leq 1$.
2) The voters will perform audit with some probability at least $p \leq 1$.

By the properties of binomial distribution and without proceeding to technical details, we can show that the E2E verifiability error is now

$$2^{-h\theta} + \left(1 - \frac{p}{2}\right)^{\delta} \qquad (1)$$

for any $p \leq 1/2$ (which is what we could expect in the real-world). Clearly the error in Eq. (1) is upper bounded by $2^{-\theta} + 2^{-\delta}$, for the special ideal case where $h = p = 1$.

In Fig. 7, we provide the error bound w.r.t. total min entropy $h\theta$ for tally deviation $\delta = 100$ and various values of $p$. Note that if we normalize tally deviation per the number of voters, say $n = 10000, 100000, 1000000$, the deviation is $\delta/n = 1\%, 0.1\%, 0.01\%$ respectively. Hence, the impact of an attack with a certain success probability diminishes rapidly as we move towards large (national) scale elections.

From the data collected by our pilot experiments we deduced that the voters were choosing to use side A to vote with probability $\approx 85\%$. If, for simplicity, we treat the voters' behavior uniformly (this can be argued assuming the adversary does not hold any history of every individual's behavior), then the latter implies min entropy per voter of $h = -\log(\max\{0.85, 0.15\}) \approx 0.234$. In addition, the ratio of participants that audited the bulletin board was $p = 2.8\%$. In Fig. 8, we plot the probability error for the specific values $h, p$ w.r.t. normalized tally deviation $\delta/n$ and various electorate sizes, given that the majority of voters is honest (otherwise, clearly, the adversary can completely manipulate the result). We observe that even in this pessimistic scenario, the probability that the election integrity is violated beyond $\delta/n = 1\%$ is below $< 10^{-6}$ for $n = 100000$ (note that the same probability error holds for $\delta = 0.1\%$ and $n = 1000000$).

## References

[1] A. Kiayias, T. Zacharias, and B. Zhang, "End-to-end verifiable elections in the standard model," in *EUROCRYPT 2015 Proceedings, Part II*, ser. LNCS, vol. 9057. Springer, 2015, pp. 468–498.

[2] B. Adida, "Helios: Web-based open-audit voting," in *USENIX 2008 Proceedings*. USENIX Association, 2008, pp. 335–348.

[3] F. Zagórski, R. Carback, D. Chaum, J. Clark, A. Essex, and P. L. Vora, "Remotegrity: Design and use of an end-to-end verifiable remote voting system," in *ACNS 2013 Proceedings*, ser. LNCS, vol. 7954. Springer, 2013, pp. 441–457.

[4] D. Chaum, R. Carback, J. Clark, A. Essex, S. Popoveniuc, R. L. Rivest, P. Y. A. Ryan, E. Shen, A. T. Sherman, and P. L. Vora, "Scantegrity II: end-to-end verifiability by voters of optical scan elections through confirmation codes," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 4, pp. 611–627, 2009.

[5] C. Baum, I. Damgård, and C. Orlandi, "Publicly auditable secure multi-party computation," in *SCN 2014 Proceedings*, ser. LNCS, vol. 8642. Springer, 2014, pp. 175–196.

[6] B. Barak, R. Canetti, Y. Lindell, R. Pass, and T. Rabin, "Secure computation without authentication," in *CRYPTO 2005, Proceedings*, ser. LNCS, vol. 3621. Springer, 2005, pp. 361–377.

[7] D. Chaum, "Surevote: Technical overview," in *Proceedings of the Workshop on Trustworthy Elections*, ser. WOTE, Aug. 2001.

[8] J. C. Benaloh, "Secret sharing homomorphisms: Keeping shares of a secret sharing," in *CRYPTO 1986 Proceedings*, ser. LNCS, vol. 263. Springer, 1987, pp. 251–260.

[9] J. D. Cohen and M. J. Fischer, "A robust and verifiable cryptographically secure election scheme (extended abstract)," in *FOCS 1985 Proceedings*. IEEE Computer Society, 1985, pp. 372–382.

[10] R. Cramer, R. Gennaro, and B. Schoenmakers, "A secure and optimally efficient multi-authority election scheme," *ETT*, vol. 8, no. 5, pp. 481–490, 1997.

[11] J. Benaloh, "Simple verifiable elections," ser. USENIX EVT 2006 Proceedings. USENIX Association, 2006, pp. 5–5.

[12] D. Chaum, A. Essex, R. Carback, J. Clark, S. Popoveniuc, A. T. Sherman, and P. L. Vora, "Scantegrity: End-to-end voter-verifiable optical-scan voting," *IEEE Security & Privacy Proceedings*, vol. 6, no. 3, pp. 40–46, 2008.

[13] D. Chaum and T. P. Pedersen, "Wallet databases with observers," in *CRYPTO 1992 Proceedings*, ser. LNCS, vol. 740. Springer, 1993, pp. 89–105.

[14] R. Küsters, T. Truderung, and A. Vogt, "Clash attacks on the verifiability of e-voting systems," in *Security & Privacy Proceedings*. IEEE Computer Society, 2012, pp. 395–409.