USENIX Journal of Election Technology and Systems (JETS)
Volume 2, Number 3 • July 2014
www.usenix.org/jets/issues/0203

Every Vote Counts: Ensuring Integrity in Large-Scale Electronic Voting

Feng Hao, Newcastle University, UK
Matthew N. Kreeger, Thales E-Security, UK
Brian Randell, Newcastle University, UK
Dylan Clarke, Newcastle University, UK
Siamak F. Shahandashti, Newcastle University, UK
Peter Hyun-Jeen Lee, Newcastle University, UK

This paper presents a new End-to-End (E2E) verifiable e-voting protocol for large-scale elections, called Direct Recording Electronic with Integrity (DRE-i). In contrast to all other E2E verifiable voting schemes, ours does not involve any Tallying Authorities (TAs). The design of DRE-i is based on the hypothesis that existing E2E voting protocols' universal dependence on TAs is a key obstacle to their practical deployment. In DRE-i, the need for TAs is removed by applying novel encryption techniques such that after the election multiplying the ciphertexts together will cancel out random factors and permit anyone to verify the tally. We describe how to apply the DRE-i protocol to enforce the tallying integrity of a DRE-based election held at a set of supervised polling stations. Each DRE machine directly records votes just as the existing practice in the real-world DRE deployment. But unlike the ordinary DRE machines, in DRE-i the machine must publish additional audit data to allow public verification of the tally. If the machine attempts to cheat by altering either votes or audit data, then the public verification of the tallying integrity will fail. To improve system reliability, we further present a fail-safe mechanism to allow graceful recovery from the effect of missing or corrupted ballots in a publicly verifiable and privacy-preserving manner. Finally, we compare DRE-i with previous related voting schemes and show several improvements in security, efficiency and usability. This highlights the promising potential of a new category of voting systems that are E2E verifiable and TA-free. We call this new category "self-enforcing electronic voting".

1. INTRODUCTION

Background. An electronic voting (e-voting) system is a voting system in which the election data is recorded, stored and processed primarily as digital information [VoteHere 2002]. Depending on the implementation, e-voting can be either local or remote. Local e-voting occurs at a supervised polling station, normally using a touch-screen machine to record votes directly. Such a machine is often called a Direct Recording Electronic (or DRE) machine [Kohno et al. 2004]. In contrast, remote e-voting can be conducted at any location, usually through a web browser [Adida 2008; Adida et al. 2009].

E-voting has already been widely deployed across the world. As shown in USA Today [Wolf 2008], the use of DRE expanded rapidly in the United States following the 2000 national election: from 12% of the votes cast in that election to 29% in 2004, and to 38% in 2006. India moved to full DRE voting in their 2004 national election, and Brazil started its first fully DRE-based election in 2002 [Blanc 2007]. In 2007, Estonia became the first country to allow Internet voting for national elections [Krimmer et al. 2007]. Many other countries have been actively pursuing the implementation of e-voting [Alvarez et al. 2011; Pieters 2011].

Controversy. However, e-voting has become controversial. In 2004, Kohno *et al.* critically analysed a type of e-voting machine that had been widely used in the US, and discovered serious software vulnerabilities and bugs [Kohno et al. 2004]. The alarming level of security flaws was especially worrying because the U.S. government had earlier certified the machine to be "trustworthy". In response to these and other similar findings [Sherman et al. 2006; Jefferson et al. 2004] regarding other manufacturers' machines, many people have demanded that e-voting be abandoned completely. Several U.S. states consequently abandoned the use of e-voting machines in 2008, causing a rapid decline of DRE usage from 38% in 2006 to 32% in 2008 [Wolf 2008]. Similar problems have also been reported in other countries, e.g., Germany, Netherlands and Ireland have all sus-

This work is supported by the European Research Council (ERC) Starting Grant (No. 106591) on "Self-enforcing electronic voting: trustworthy elections in the presence of corrupt authorities".

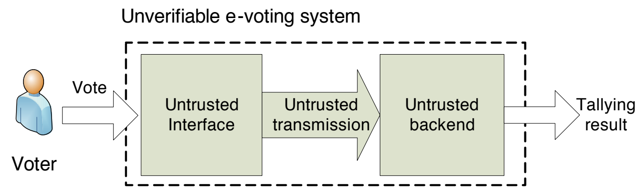

Fig. 1. An unverifiable (black-box) e-voting system

pended e-voting in 2008-2009 [Alvarez et al. 2011; Pieters 2011]. In 2010, researchers also started to seriously question the integrity of e-voting machines used for elections in India [Wolchok et al. 2010].

A fundamental problem with many deployed e-voting systems (including the discarded/suspended ones) is that they are *unverifiable* [Pieters 2011]. Essentially each system works like a black-box (Figure 1). After voting, the voter has no means of telling whether her vote was correctly recorded. At the end of the election, the system announces the tallied result for each candidate, but any independent verification of this result is impossible.

Typically, a black-box e-voting system comprises three components: a voting interface, a transmission mechanism and a tallying back-end (see Figure 1). The voting interface may be a touch screen in a local DRE-based election, or it may be a web browser in a remote Internet-based election. In either case, a compromised voting interface (touch-screen DRE or a web browser) may surreptitiously change the voter's choice; the transmission of electronic votes (either off-line or on-line) may be intercepted and the votes modified; and the back-end may maliciously change the tally to support some particular candidate regardless of the actual vote count. In summary, there are many opportunities for an attacker to tamper with the electronic data without the public being aware of the change.

This can be contrasted with elections that involve votes being recorded on a visible physical medium such as a printed paper ballot form or a punched card. The processing of such votes can be easily and effectively monitored (e.g., by multiple independent poll-watchers, both professionals and amateurs). And these votes can be retained in case of a challenge, and if necessary be recounted. However, similar direct physical monitoring is not possible in electronic voting.

Government certification of an e-voting system's hardware and software was perceived by many countries as the solution to the problem of achieving trustworthy e-voting [Alvarez et al. 2011; Pieters 2011], but has proved inadequate for several reasons. First of all, it requires people to trust the probity and competence of the certification authority. Second, it does not solve the fundamental problem, because a certified black-box is still a black-box (i.e., its operation is unverifiable). Third, researchers have repeatedly demonstrated that attackers can successfully compromise "certified" e-voting systems, altering election results without their activities being detected [Sherman et al. 2006; Kohno et al. 2004]. All these greatly reduce public confidence in such government certification.

Therefore, for e-voting to succeed in the future, it is important that the voting system be verifiable [Adida 2008; Adida et al. 2009]. However, it is worth noting that the idea of verifiable e-voting is not new; it has existed for over twenty years [Benaloh 1987]. Although progress has been made in trialling verifiable e-voting protocols in practice [Chaum et al. 2008a; Adida et al. 2009], so far the impact on real-world national elections has been limited. In practice, many countries around the world are still using unverifiable (black-box) e-voting systems.

E2E verifiability. To explain the limitations of existing verifiable e-voting technology, we first need to clarify what is meant by being "verifiable". In general verifiability has two levels of meaning: individual and universal [Chaum et al. 2008a]. At the individual level, all voters should be able to verify that their votes have been correctly recorded and have been correctly included into the tally. At the universal level, anyone in the world should be able to verify the integrity of the tallying result, based on publicly available audit data. E-voting systems that satisfy the verifiability at both levels are generally termed as being End-to-End (E2E) verifiable [Adida 2008]. We refer the reader to papers by Küsters *et al.* [Küsters and Vogt 2010] and Popoveniuc *et al.* [Popoveniuc et al. 2010] for more formal definitions of E2E verifiability.

Some researchers suggested adding a Voter Verifiable Paper Audit Trail (VVPAT) to a DRE machine. Most notably, the method proposed by Mercuri [Mercuri 2001] works as follows: when the voter makes a selection on the touch-screen, the machine prints the selected choice on a paper receipt in plain text. The voter can visually inspect the receipt under a layer of glass before it is automatically transferred to a secure location. The voter is not allowed to take the receipt home as that would reveal (to a coercer) how she had voted. Overall, this method improves the individual verifiability by allowing voters to check if their votes have been recorded correctly. Also, it provides a physical paper trail that permits a manual recount in case of a dispute. However, the VVPAT method provides no means for voters to check whether the recorded votes will be securely transported to the tallying unit and whether the votes will be tallied correctly. Therefore, a DRE system based on VVPAT alone is not E2E verifiable.

Thus, the dual, and potentially conflicting, challenges faced by the designers of any voting system are to ensure the system is publicly verifiable and meanwhile to preserve the voter's privacy. To satisfy the E2E verifiability, it is necessary to provide the voter a receipt, which can be checked against a public bulletin board [Chaum et al. 2008a]. In order to prevent coercion and vote selling, the receipt must not reveal any information about how the voter had voted. On the other hand, if the receipt does not show how the voter had voted, how can she be sure it is a correct record of her vote? These requirements may seem clearly contradictory, but past research has shown that they can be met by combining various techniques, e.g., cryptography and voter-initiated auditing [Adida et al. 2009; Benaloh 2007].

To date, many E2E verifiable voting protocols have been proposed. Well-known examples include: Adder [Kiayias et al. 2006], Civitas [Clarkson et al. 2008], Helios [Adida 2008; Adida et al. 2009], Scantegrity [Chaum et al. 2008b], Scantegrity II [Chaum et al. 2008a], Prêt à Voter [Ryan et al. 2009], MarkPledge [Adida and Neff 2006] and Chaum's visual cryptographic scheme [Chaum 2004]. All these protocols rely on there being multiple independent Tallying Authorities (TAs) to perform and control the tallying process in a publicly verifiable manner. Hence, we choose to categorise them as "TA-based E2E verifiable e-voting".

Protocols in this category generally work as follows (Figure 2): the voter, using a voting interface, casts a vote and obtains a receipt. The receipt is encrypted under a set of tallying authorities' public keys (or one joint public key). At the end of the election, the system publishes all the receipts on a public bulletin board (e.g., a mirrored public web site), so that voters can check if their votes have been recorded. However, individual voters are unable to decrypt their receipts to confirm their votes have been correctly recorded. Instead they are provided with some other (indirect) way of gaining confidence that this is the case (through voter-initiated auditing, as we will explain in Section 2).

Since all the data on the bulletin board is encrypted, the tallying authorities are needed to perform the decryption and tallying process. This process can be done in a publicly verifiable manner, so that the TAs do not need to be trusted for the integrity of the tallying result. However, they need to be trusted to some extent for the secrecy of individual votes. The common mitigating measure is to put the TAs under a k/n threshold control, where n is the total number of TAs and k is the threshold. Only if more than a threshold k number of TAs are corrupted will they be able to decrypt each individual vote. Furthermore, it is normally assumed that the TAs are selected from different parties with conflicting interests, hence they supposedly lack the incentive to collude. (Nonetheless, it is important to ensure the TAs use *independent* software, because "if all trustees (TAs) use tallying software from a single source, then this software might collude without the trustees' knowledge." [Karlof et al. 2005])

Implementing E2E verifiability. Although many TA-based E2E verifiable voting protocols have been proposed, only a few have actually been implemented in practice. The Helios voting system is notable for being the first web-based implementation of an E2E verifiable voting system. Initially, Helios (v1.0) used mix-net based tallying [Adida 2008], and later it (v2.0) was changed to using

USENIX Journal of Election Technology and Systems (JETS)
Volume 2, Number 3 • July 2014
www.usenix.org/jets/issues/0203

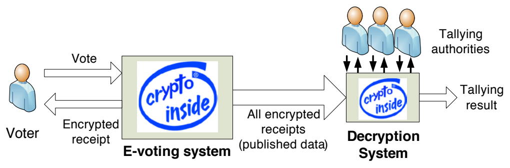

Fig. 2. TA-based e-voting with E2E verifiability

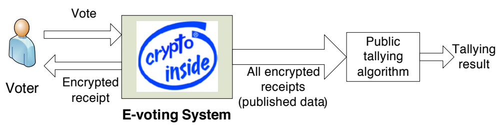

Fig. 3. Self-enforcing e-voting with E2E verifiability

homomorphic tallying [Adida et al. 2009]. In 2009, a customized variant of Helios 2.0 was adopted by the Université catholique de Louvain (UCL) in a campus election to elect the university president.

As highlighted in the Helios paper [Adida et al. 2009], the practical implementation of tallying authorities has proved to be "a particularly difficult issue". To ensure the fairness in the representations, the tallying authorities were chosen from various groups (students, administrative staff and so on) with different backgrounds (not just computer science). However, it turned out that the chosen authorities did not have the required technical expertise to perform complex cryptographic operations. Hence, a group of "external experts" (whose identities are not mentioned in the Helios paper [Adida et al. 2009]) were invited to first perform the key generation on behalf of the tallying authorities. The whole procedure included purchasing brand new laptops, removing the hard disk drives, disabling wireless network cards, booting up the machines using standard linux live-CDs and loading the key generation code (written in Python) through the USB sticks. Subsequently, the tallying authorities' private keys were generated and stored on the USB sticks, which were then distributed to the authorities. In the mean time, all of the generated private keys were centrally backed up by one trusted third party (a notary public). After the election, "a similar procedure was followed when those keys were used for decryption" [Adida et al. 2009]. Clearly, the tallying authorities' further dependence on "external experts" and a single trusted third party for backup has significantly complicated the trust relationships in the election management.

Removing TAs. A few researchers have investigated how to remove tallying authorities in electronic voting. Kiayias and Yung first studied this in 2002 with a boardroom voting protocol [Kiayias and Yung 2002], followed by Groth in 2004 [Groth 2004] and Hao-Ryan-Zieliński in 2010 [Hao et al. 2010]. Among these boardroom voting protocols, the Hao-Ryan-Zieliński's solution [Hao et al. 2010] is so far the most efficient in every aspect: the number of rounds, the computation load and the message size. In general, a boardroom voting protocol works by requiring voters to cooperatively interact with all other voters in a network in a number of rounds. In the best case [Hao et al. 2010], only two rounds of interactions are needed. The tallying result is usually computed by voters through exhaustive search. Essentially, the voting is totally decentralized and run by the voters themselves. A decentralized boardroom voting protocol, such as Kiayias-Yung's [Kiayias and Yung 2002], Groth's [Groth 2004], or Hao-Ryan-Zieliński's [Hao et al. 2010], can provide the theoretically-best protection of ballot secrecy. In order to learn a voter's secret choice, the attacker must compromise all other voters to form a full collusion against the voter [Kiayias and Yung 2002; Groth 2004; Hao et al. 2010].

A boardroom voting protocol is considered different from an E2E verifiable voting protocol for a number of reasons. First of all, they differ on the scales. The former is usually designed for small-scale voting in a boardroom, while the latter is normally for large-scale country voting. Using exhaustive search to determine the tally may be straightforward in boardroom voting, but it may prove expensive if the election is a large-scale one (especially for multi-candidate elections). Second, the system infrastructures are different. The former is decentralized; voters use their own *trusted* computing hardware/software to interact with all other voters through a fully connected network. There is no voter-receipt (as there is no entity to issue receipts) and there is no central bulletin board to check receipts [Hao et al. 2010]. The latter is centralized; there is little interaction between voters. People vote through some common voting interface (e.g., touch-screen DRE). A voter normally gets a receipt, which can be compared against a central bulletin board. Third, the security requirements are completely different. For example, in a boardroom voting protocol [Kiayias and Yung 2002; Groth 2004; Hao et al. 2010], a voter can trivially prove to a coercer how she had voted by revealing the ephemeral secret generated during the protocol. Furthermore, any arbitrary voter can easily disrupt a multi-round voting procedure by simply dropping out half-way in the protocol. While coercion, vote selling and voter disruption might not be considered serious issues in a small boardroom, they are important considerations in the design of an E2E verifiable voting system.

The scope of this paper is to focus on E2E verifiable voting systems for large-scale elections. Existing boardroom voting protocols are clearly unsuitable for any country-scale elections. However, they are still relevant to our study as they demonstrate that it is possible to remove TAs albeit only in the setting of a small-scale election. To the best of our knowledge, no one has investigated the feasibility of removing tallying authorities for large-scale elections. Indeed, existing E2E verifiable e-voting protocols designed for large-scale elections universally require involving external tallying authorities in the tallying process [Kiayias et al. 2006; Clarkson et al. 2008; Adida 2008; Adida et al. 2009; Chaum et al. 2008b; Chaum et al. 2008a; Ryan et al. 2009; Adida and Neff 2006; Chaum 2004].

Contributions. We initiate a study on whether it is feasible to remove the dependence on external tallying authorities in an E2E verifiable voting system. Along this direction, we propose to replace the tallying authorities and the decryption system in Figure 2 by a public algorithm. We define the resultant system as a "self-enforcing e-voting" system (see Figure 3). Because the algorithm is public, the tallying process is fully verifiable without any TA involvement. The main contributions of this paper are summarized below:

— We present the first E2E verifiable voting protocol that is TA-free. Our protocol is called Direct Recording Electronic with Integrity (DRE-i). Its "self-enforcing" property is realized by integrating a cancellation formula [Hao and Zieliński 2006] into the homomorphic tallying process: the encryption of votes follows a well-defined structure such that after the election multiplying the ciphertexts together will cancel out random factors and permit anyone to verify the tally. A similar tallying method was used in a previous Hao-Ryan-Zieliński boardroom voting protocol [Hao et al. 2010], but ours does not require exhaustive search. Although the two protocols share the same mathematical formula for cancelling random factors, they are designed for completely different election scenarios and have different security requirements.
— We effectively combine the basic DRE-i with several additional engineering designs to make it an overall secure and practical system, suitable for a DRE-based election at polling stations. The first is to seamlessly integrate the voter's initiated auditing into the natural confirm/cancel voting experience on a touch-screen DRE. As a result, the system is user-friendly to a voter who does not understand cryptography at all. Furthermore, we provide a fail-safe mechanism to allow graceful recovery of partially corrupted audit data in a publicly verifiable and privacy-preserving way. Finally, we support a distributed computation of secret keys to distribute trust and improve system availability. (Advantages of our system over previous ones will be detailed in Section 4.)

2. A SELF-ENFORCING E-VOTING PROTOCOL

In this section, we describe a self-enforcing e-voting protocol called Direct Recording Electronic with Integrity (DRE-i). In particular, we show how to apply the DRE-i protocol to enforce the tallying integrity of DRE-based local voting at the polling station. (It is possible to implement DRE-i for remote voting [Hao et al. 2012; Hao et al. 2013], but to avoid confusion, we will focus on local voting in this paper.) For the simplicity of discussion, we will consider a single-candidate election first, and then extend it to multiple candidates.

2.1. User roles

In an E2E verifiable e-voting protocol, there are generally three user roles as defined below [Adida et al. 2009].

(1) **Ordinary voter**: Someone who directly participates in the voting.

(2) **Auditor**: Someone who audits the system by performing real-time checks on the system during the voting process.
(3) **Universal verifier**: Anyone in the world who has the technical expertise to verify the audit data published by the voting system.

2.2. Integrity requirements

We also adopt the commonly accepted integrity requirements for an E2E verifiable voting protocol [Adida et al. 2009; Chaum et al. 2008b; Chaum et al. 2008a].

(1) **Ballot format integrity**: Everyone, including third party observers, should be able to verify that every encrypted ballot has the correct format to represent exactly one vote.
(2) **Ballot casting integrity**: All voters should be able to convince themselves that their cast ballots are recorded to the correct candidates.
(3) **Ballot transmission integrity**: All voters should be able to verify that their recorded ballots have been correctly transmitted to the tallying process.
(4) **Ballot tallying integrity:** Everyone, include third party observers, should be able to verify that the tallying result is correctly obtained from the recorded ballots.

Obviously, the integrity requirements must be satisfied without compromising the voter's privacy. In particular, the receipt that permits a voter to verify the integrity of the voting system must not reveal how she had voted. We will explain in Section 3 that this holds true in DRE-i.

2.3. Trust assumptions

There are many other requirements to make a secure e-voting system. Since the satisfaction of those requirements is generally assumed in the literature [Kiayias et al. 2006; Clarkson et al. 2008; Adida 2008; Adida et al. 2009; Chaum et al. 2008b; Chaum et al. 2008a; Adida and Neff 2006; Chaum 2004], we make the same assumptions, namely:

(1) **User enrolment**: Only eligible users can be enrolled in the voter registration.
(2) **User authentication**: Only authenticated voters are allowed to vote during the election.
(3) **One-man-one-vote**: Each authenticated voter is allowed to vote just once.
(4) **Voting privacy**: Voting happens in a private space that no one else can observe.
(5) **Anonymity**: The voting machine that is used does not know the real identity of the voter.
(6) **Public bulletin board**: There is a publicly readable, append-only bulletin board (e.g., a mirrored public website), on which the legitimate voting system can publish audit data for verification (the authenticity of data can be ensured by the use of digital signatures).

If voting takes place in a supervised environment (say a polling station), it is relatively easy to meet the above assumptions. For example, the polling station staff can authenticate voters based on their ID documents or even biometrics. After successful authentication, the voter is free to take a single random authentication token, say a smart card. The voter then enters a private booth and uses the token to authenticate herself to the machine and starts voting [Kohno et al. 2004]. To ensure one-person-one-vote, the polling station can publish a list of the names of the people who voted, so that anyone can verify that the number of voters matches the number of cast votes [Chaum et al. 2008a]. Observers at a polling station can also independently count how many people have actually voted.

2.4. Three Stages of Voting

The DRE-i protocol consists of three phases: setup, voting and tallying. The following sections explain each phase in detail.

2.4.1. Setup phase. We describe the protocol in a multiplicative cyclic group setting (i.e., DSA-like group), though the same protocol also works in an additive cyclic group (i.e., ECDSA-like group). Let p and q be two large primes, where $q \mid p-1$. $\mathbb{Z}_p^*$ is a multiplicative cyclic group and

Table I. Setup phase before election

Ballot No	**Random public key**	**Restructured public key**	**Cryptogram of no-vote**	**Cryptogram of yes-vote**
1	g^{x_1}	g^{y_1}	$g^{x_1 \cdot y_1}$, 1-of-2 ZKP	$g^{x_1 \cdot y_1} \cdot g$, 1-of-2 ZKP
2	g^{x_2}	g^{y_2}	$g^{x_2 \cdot y_2}$, 1-of-2 ZKP	$g^{x_2 \cdot y_2} \cdot g$, 1-of-2 ZKP
…	…	…	…	…
n	g^{x_n}	g^{y_n}	$g^{x_n \cdot y_n}$, 1-of-2 ZKP	$g^{x_n \cdot y_n} \cdot g$, 1-of-2 ZKP

Note: Data in the first three columns are published on a public bulletin board before the election. They serve as commitment so that the values cannot be later changed. Data in the last two columns are kept secret; they are either computed on-demand during voting or pre-computed before the election.

G_q its subgroup of prime order q. Let g be the generator of G_q (any non-identity element in G_q can serve as a generator). We assume the Decision Diffie-Hellman (DDH) problem [Stinson 2006] in G_q is intractable. The parameters (p,q,g) are publicly agreed before the election starts. Unless the contrary is stated explicitly, all the modular operations are performed with respect to the modulus p. Hence, we omit the explicit "$\bmod p$" for simplicity.

First of all, the DRE machine generates a private signing key, say using DSA or ECDSA [Stinson 2006], and publishes the public key on the bulletin board. A tamper-resistant module is used to securely manage the private signing key, in line with industry standard practice [Anderson 2008]. The private signing key is generated on-board in the secure memory of the module and never leaves the protected boundary of the module.

Subsequently, the DRE machine computes a table as shown in Table I. The table contains n rows with each row corresponding to a ballot, so there are n ballots in total. The number n is the product of the total number of the eligible voters and a safety factor (> 1). The safety factor, say 10, is defined so as to allow the generation of extra ballots for auditing purposes (as we will explain later).

Each row in Table I corresponds to a ballot with encrypted data (cryptograms) to represent candidate choices. In a single-candidate election, the choices are "Yes" and "No". All rows are constructed to satisfy four properties. First, given any cryptogram in any row, one can easily verify that it is an encryption of one of the two values: "Yes" or "No" (which translate to 1 and 0 in the implementation). Second, given only a single cryptogram from any selected row, one cannot tell whether it is "Yes" or "No". Third, given both cryptograms (unordered) from any selected row, anyone will be able to easily tell which is "Yes" and which is "No". Fourth, given a set of cryptograms, each of which was arbitrarily selected, one from each row, one can easily check how many "Yes" values in total are in the set. In the following, we will explain how these four properties are fulfilled and how they are useful in building a self-enforcing e-voting system.

The system fills the table as follows. For each of the n ballots, the system computes a random public key g^{x_i}, where $x_i \in_R [1, q-1]$. When this has been done for all the ballots, the system computes $g^{y_i} = \prod_{j<i} g^{x_j} / \prod_{j>i} g^{x_j}$ for every ballot. Here, we call the obtained g^{y_i} a restructured public key, because it is constructed by multiplying all the random public keys before i and dividing the result by all the public keys after i. Note that anyone is able to compute g^{y_i} based on the published g^{x_i} values.

The "Yes"/"No" value in each ballot is encoded in the form of $C_i = g^{x_i y_i} \cdot g^{v_i}$ where $v_i = 0$ for "No" and 1 for "Yes". The no-vote, $g^{x_i y_i}$, is indistinguishable from random based on the DDH assumption (detailed proofs can be found in Section 3). Clearly, the yes-vote, $g^{x_i y_i} \cdot g$, is indistinguishable from random too. However, if both no-vote and yes-vote are published, then it is trivial to distinguish which is "No" and which is "Yes" (because the latter is the former multiplied by g).

In addition, the system needs to compute a 1-out-of-2 Zero Knowledge Proof (ZKP) for each yes/no value. This is to ensure that the value of the vote is indeed in the correct form of $C_i = g^{x_i y_i} \cdot g^{v_i}$ where $v_i \in \{0,1\}$. In other words, the value v_i can only be one of: 0 and 1. We adopt the standard 1-out-of-n ZKP technique (also known as the CDS protocol) due to Cramer, Damgård and Schoenmakers [Cramer et al. 1994]. Although the original CDS protocol is designed for ElGamal encryption, it is directly applicable here if we regard g^{y_i} as a public key. (The only difference is

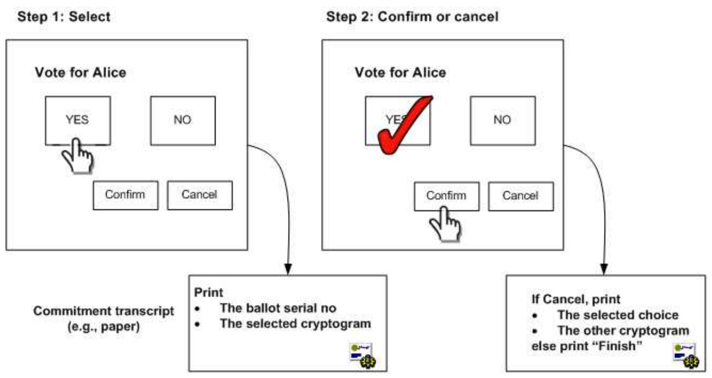

Fig. 4. A simple single-candidate voting interface. The receipt has two parts: the first includes the printout in Step 1 with a digital signature and the second includes the printout in Step 2 with a signature that covers the entire transcript.

that the public key in ElGamal encryption is statically fixed, while in our case, it is dynamically constructed from g^{x_i} values for each ballot.) Here, we use $n = 2$. The original three-move interactive CDS protocol can be made non-interactive by applying the standard Fiat-Shamir heuristics [Fiat and Shamir 1987]. The same CDS technique has been widely used in previous e-voting protocols to ensure the published ciphertext is well-formed.

As shown in Table I, the cryptogram of the no-vote contains $g^{x_i y_i}$ and a 1-out-of-2 ZKP; similarly, the cryptogram of the yes-vote comprises $g^{x_i y_i} \cdot g$ and a corresponding 1-out-of-2 ZKP.

Similar to the private signing key, all x_i secrets are generated on-board in the module and are stored within the module's secure memory. The corresponding public keys (g^{x_i}) are published on the bulletin board before the election; they serve as commitment so the values cannot be changed later. To ensure authenticity, all commitment data published on the bulletin board should be digitally signed. Let us assume $n = 10^5$ and a group setting of 2048-bit p and 256-bit q. The total size of x_i secrets is 3.2 MB. Hence, it is possible to store the x_i secrets entirely in the module's memory. (As an example, a high capacity smart card can have 16 MB non-volatile memory.)

In order to optimize the performance in voting, one may choose to pre-compute all the cryptograms (last two columns in Table I) before the election. In that case, the secrecy of pre-computed cryptograms needs to be protected at the same level as the x_i secrets. If the size of the cryptograms is more than what the module's memory can accommodate, one solution, as commonly adopted in industry, is to generate a master key on-board in the module and use the master key to encrypt blobs of data in an authentic manner, so that the encrypted blobs can be stored outside the module and be reloaded back to memory when needed [Anderson 2008]. This is a typical trade-off between memory and speed. Reloading the blob to memory will involve some decryption work, but since it is only a symmetric-key operation, it can be very fast.

2.4.2. Voting phase. As stated before, we assume the eligible voter has been properly authenticated. She first obtains a random authentication token, enters a private voting booth, uses the token to authenticate herself to the machine and starts voting. The voter is prompted to select a choice on a touch screen DRE (see Figure 4). To cast her ballot, the voter follows two basic steps below.

In step one, the voter selects a choice on the screen. Meanwhile, the machine prints the following data on the paper: the ballot serial number i, and the cryptogram of the selected choice. (The ballot serial number i may be incremental or randomly assigned; there is no significant difference from the protocol's perspective as long as the number is unique.) The printed data serve as a commitment, as it cannot be changed. The commitment transcript is digitally signed by the machine to prove its authenticity. As explained earlier, the machine's public key is publicly announced before the election, so the signature is universally verifiable.

Table II. Ballot tallying.

No i	**Random pub key** g^{x_i}	**Restructured pub key** g^{y_i}	**Published Votes** V_i	**ZKPs**
1	g^{x_1}	g^{y_1}	Valid: $g^{x_1 \cdot y_1}$	a 1-of-2 ZKP
2	g^{x_2}	g^{y_2}	Valid: $g^{x_2 \cdot y_2} \cdot g$	a 1-of-2 ZKP
3	g^{x_3}	g^{y_3}	Dummy: $g^{x_3 \cdot y_3}$, $g^{x_3 \cdot y_3} \cdot g$	two 1-of-2 ZKPs
…	…	…	…	…
n	g^{x_n}	g^{y_n}	Dummy: $g^{x_n \cdot y_n}$, $g^{x_n \cdot y_n} \cdot g$	two 1-of-2 ZKPs

Note: This entire table is published on the public bulletin board. A vote can be either valid or dummy. Ballot No. 1 shows an example of a valid “No” vote, and No. 2 shows an example of a valid “Yes” vote. Tallying involves multiplying all the V_i values (only including the “No” votes for the dummy case).

In step two, the voter has the option of either confirming or cancelling the previous selection. If she chooses to confirm, the system will print a “finish” message on the paper, and a valid encrypted vote has been cast. On the other hand, if she chooses to cancel, the DRE machine will reveal the selected choice in plain text ("Yes" or "No"), and also print the other cryptogram on the paper. In this case, a dummy vote has been cast. The touch screen will return to the previous step and provide another unused ballot. Voters are entitled to cast as many dummy votes as they wish[1], but are allowed to cast only a single valid vote.

The confirm/cancel option in step two serves to provide ballot casting assurance, namely: the voter needs to gain confidence that her actual vote has been recorded as she intended. For example, a corrupted machine might cheat by swapping the “No”/”Yes” cryptograms. The solution here is to have the machine initially commit to a value, and then give the voter an option to challenge the machine to reveal the commitment so that if the machine has cheated, it will be caught once the voter chooses to audit. Successful cheating on any large scale without being detected is extremely unlikely. Our auditing procedure is consistent, in spirit, with Benaloh’s idea of voter-initiated challenges [Benaloh 2007], but it has been more tightly integrated into the overall cryptographic system starting with the initial setup.

The commitment transcript, signed by the machine, for the entire voting session can be printed on a single piece of paper, which forms the voter’s receipt. The data on the receipt is also available on the public bulletin board. The voter is free to take home the receipt and compare it against the bulletin board, so gaining a degree of trust in the bulletin board’s contents. (This is just as in other verifiable e-voting protocols [Kiayias et al. 2006; Clarkson et al. 2008; Adida 2008; Adida et al. 2009; Chaum et al. 2008b; Chaum et al. 2008a; Adida and Neff 2006; Chaum 2004]). When all the voters have cast their votes, or the election time limit is up, the system will publish both the yes-vote and no-vote cryptograms for the remaining unused ballots and mark them as “dummy” votes.

2.4.3. Tallying phase. Tallying the ballots involves multiplying together all the published cryptograms V_i (for dummy votes, using only the *no*-vote; see Table II). Thus, we have:

$$\prod_i V_i = \prod_i g^{x_i y_i} g^{v_i} = \prod_i g^{v_i} = g^{\sum_i v_i}$$

The key to the tallying process is the fact that $\sum x_i y_i = 0$ (a cancellation formula first introduced in 2006 in the design of an anonymous veto protocol [Hao and Zieliński 2006]; we refer the reader to that paper for the proof). Thus, all random factors cancel each other out. Here, we combine this cancelation technique with the conventional homomorphic encryption to build a self-enforcing e-voting protocol. Compared with the existing mix-net or homomorphic aggregation based tallying methods, the new method has the distinctive feature of not requiring any secret keys (hence no TAs).

The term $\sum_i v_i$ is the total number of the “yes” votes. Note that we do not need to compute the exponent of $g^{\sum_i v_i}$ (although this is doable by exhaustive search). Because the DRE system records the ballots directly, it announces the count of “yes” votes, β, right after the election, as is current

[1]In practice, a reasonable upper limit would be enforced.

practice in DRE-based elections. Anyone can verify whether g^{β} and $g^{\sum_i v_i}$ are equal. This takes only one exponentiation. Also, anyone can count the number of dummy votes from the bulletin board, which we denote as λ. Thus, the tally of "no" votes is $\alpha = n - \beta - \lambda$.

There are several ways to extend a single-candidate election to multiple candidates. One straightforward method is to have a Yes/No selection for each of the candidates [Hao et al. 2010]. Another method involves defining more efficient encoding values for candidates [Cramer et al. 1996]. These are standard techniques to extend a single-candidate election to a multiple-candidate election, while the underlying voting protocol remains unchanged.

3. SYSTEM ANALYSIS

In this Section, we analyze the DRE-i protocol with regard to security, efficiency, usability and dependability.

3.1. Security analysis

First of all, we show the encryption of the "No" vote is semantically secure: in other words, the value $g^{x_i y_i}$ for the ith ballot is indistinguishable from random. As explained earlier, the system selects random values $x_i \in_R [1, q-1]$ for $i = 1, \ldots, n$. The value y_i is defined from: $g^{y_i} = \prod_{j<i} g^{x_j} / \prod_{j>i} g^{x_j}$, hence $y_i = \sum_{j<i} x_j - \sum_{j>i} x_j$. Given that x_i is random, $y_i \neq 0$ holds with an overwhelming probability (i.e., $1 - 1/q$). Furthermore, y_i is random over $[1, q-1]$ and it is independent of x_i, the value $g^{x_i y_i}$ will be uniformly distributed over non-identity elements in G [Stinson 2006]. Therefore, the term $g^{x_i y_i}$ is indistinguishable from random based on the DDH assumption as long as the x_i values are kept secret. All the $g^{x_i y_i}$ values ($i \in [1, n]$) are related by the constraint that $\prod_i g^{x_i y_i} = 1$. In the following, we will prove that such a structural relationship does not reveal any information other than the tally.

ASSUMPTION 1 (DDH VARIANT). *For a generator g and $a, b \in_R [1, q-1]$, given a tuple (g, g^a, g^b, C) in which C is either g^{ab} or g^{ab+1}, it is hard to decide whether $C = g^{ab}$ or $C = g^{ab+1}$.*

LEMMA 3.1. *Assumption 1 is implied by the DDH assumption (i.e., the problem is at least as hard as the DDH problem).*

PROOF. Consider the following tuples:

$$(g, g^a, g^b, g^{ab}), \quad (g, g^a, g^b, R), \quad (g, g^a, g^b, R'g), \quad \text{and} \quad (g, g^a, g^b, g^{ab}g),$$

for random a, b, R, and R'. DDH guarantees that the first and second tuples are indistinguishable. The second and third tuples have the exact same distribution and hence are indistinguishable. DDH also guarantees that the third and fourth tuples are indistinguishable. Hence, the first and fourth tuples, i.e. (g, g^a, g^b, g^{ab}) and (g, g^a, g^b, g^{ab+1}) are indistinguishable. $\square$

Definition 3.2 (*Bare Bulletin Board*). A *bare* bulletin board is a bulletin board without the ZKPs and digital signatures.

In the following analysis, we will first consider a bare bulletin board for simplicity, assuming the underlying ZKPs and digital signature schemes are secure primitives. The ZKPs serve to prove that the ciphertexts published on the bulletin board are well-formed, and they do not reveal any information about the plaintext votes. The digital signatures serve to prove that all data published on the bulletin board are authentic; they are not related to the secrecy of votes.

LEMMA 3.3. *Consider two DRE-i elections in which all the votes are exactly the same except for two votes v_i and v_j which are swapped between the two elections. Under Assumption 1, the bare bulletin boards of these two elections are indistinguishable to an adversary that has the capability to determine an arbitrary number of votes other than v_i and v_j.*

PROOF. If the adversary is one of the voters, he is able to define his own vote. To make it general, we assume a more powerful adversary who can define an arbitrary number of votes except two: v_i

Table III. The simulated bare bulletin boards in the proof of Lemma 3.3.

k	$\mathbf{g^{x_k}}$	$\mathbf{g^{y_k}}$	$\mathbf{g^{x_k y_k} \cdot g^{v_k}}$
1	g^{x_1}	$1/\prod_{k>1} g^{x_k}$	$g^{x_1 y_1} \cdot g^{v_1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	g^a	$\prod_{k<i} g^{x_k} / \prod_{k>i} g^{x_k}$	$(g^a)^{\sigma_i} \cdot g/C$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
j	g^b	$\prod_{k<j} g^{x_k} / \prod_{k>j} g^{x_k}$	$(g^b)^{\sigma_j} \cdot C$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	g^{x_n}	$\prod_{k<n} g^{x_k}$	$g^{x_n y_n} \cdot g^{v_n}$

$\Leftrightarrow$

k	$\mathbf{g^{x_k}}$	$\mathbf{g^{y_k}}$	$\mathbf{g^{x_k y_k} \cdot g^{v_k}}$
1	g^{x_1}	$1/\prod_{k>1} g^{x_k}$	$g^{x_1 y_1} \cdot g^{v_1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	g^a	$\prod_{k<i} g^{x_k} / \prod_{k>i} g^{x_k}$	$(g^a)^{\sigma_i} \cdot g/C$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
j	g^b	$\prod_{k<j} g^{x_k} / \prod_{k>j} g^{x_k}$	$(g^b)^{\sigma_j} \cdot C$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	g^{x_n}	$\prod_{k<n} g^{x_k}$	$g^{x_n y_n} \cdot g^{v_n}$

Note: The two tables are identical except that $C = g^{ab}$ in one table and $C = g^{ab+1}$ in the other. They are indistinguishable as long as the two C values are indistinguishable.

and v_j. Let us assume w.l.o.g. that $i < j$. If $v_i = v_j$, the lemma holds trivially. In the following we give a proof for $v_i \neq v_j$.

Let us assume there is an adversary $\mathcal{A}$ that first chooses an arbitrary number of the votes other than v_i and v_j, and eventually distinguishes the two elections. Given a tuple (g, g^a, g^b, C), where $a, b \in_R [1, q-1]$ and C equals either g^{ab} or g^{ab+1}, we now construct an algorithm $\mathcal{S}$ that uses $\mathcal{A}$ to break Assumption 1. The algorithm $\mathcal{S}$ sets up the bulletin board with the generator g as below. Let $I = \{1, \ldots, n\} \setminus \{i, j\}$.

First, $\mathcal{S}$ chooses $n-2$ random values x_k for all $k \in I$. $\mathcal{S}$ sets $g^{x_i} \leftarrow g^a$, $g^{x_j} \leftarrow g^b$, and calculates g^{x_k} for all $k \in I$. Note that we implicitly have $x_i = a$ and $x_j = b$. Let $s_1 = \sum_{k<i} x_k$, $s_2 = \sum_{i<k<j} x_k$, and $s_3 = \sum_{k>j} x_k$. $\mathcal{S}$ also calculates s_1, s_2, and s_3 and then computes $\sigma_i = s_1 - s_2 - s_3$ and $\sigma_j = s_1 + s_2 - s_3$. Figure 5 illustrates the relations between x_k values and a, b, s_1, s_2, and s_3.

$$\boxed{\underbrace{x_1, x_2, \ldots, x_{i-1}}_{\sum\limits_{k<i} x_i = s_1},\ \underset{\substack{\downarrow \\ a}}{x_i}\ ,\underbrace{x_{i+1}, \ldots, x_{j-1}}_{\sum\limits_{i<k<j} x_i = s_2},\ \underset{\substack{\downarrow \\ b}}{x_j}\ ,\underbrace{x_{j+1}, \ldots, x_{n-1}, x_n}_{\sum\limits_{k>j} x_i = s_3}}$$

Fig. 5. x_i values used in the simulation

Now given all g^{x_k}, all g^{y_k} can be computed accordingly. Note that we implicitly have:

$$y_i = \sum_{k<i} x_k - \sum_{k>i} x_k = s_1 - (s_2 + b + s_3) = \sigma_i - b$$

$$y_j = \sum_{k<j} x_k - \sum_{k>j} x_k = (s_1 + a + s_2) - s_3 = \sigma_j + a$$

$\mathcal{A}$ chooses a set of votes $\{v_k\}_{k \in I_{\mathcal{A}}}$ for the set of indexes $I_{\mathcal{A}} \subseteq I$. Let us consider some arbitrary set of votes $\{v_k\}_{k \in I \setminus I_{\mathcal{A}}}$. $\mathcal{S}$ can calculate $g^{x_k y_k}$ for all $k \in I$, since it knows x_k and g^{y_k}. Hence, it can calculate $g^{x_k y_k} g^{v_k}$ for all $k \in I$. For $k = i, j$, $\mathcal{S}$ sets

$$g^{x_i y_i} g^{v_i} \leftarrow (g^a)^{\sigma_i} \cdot g/C \quad \text{and} \quad g^{x_j y_j} g^{v_j} \leftarrow \left(g^b\right)^{\sigma_j} \cdot C\,.$$

Now the calculation of the entire bare bulletin board is complete. Table III shows the simulated bare bulletin board.

In the case that $C = g^{ab}$, we have:

$$g^{x_i y_i} g^{v_i} \leftarrow (g^a)^{\sigma_i} \cdot g/C = (g^a)^{\sigma_i} \cdot g/g^{ab} = g^{a(\sigma_i - b)} g = g^{x_i y_i} g \quad \text{and}$$

$$g^{x_j y_j} g^{v_j} \leftarrow \left(g^b\right)^{\sigma_j} \cdot C = \left(g^b\right)^{\sigma_j} \cdot g^{ab} = g^{b(\sigma_j + a)} = g^{x_j y_j} \ ,$$

which means that in our bare bulletin board $v_i = 1$ and $v_j = 0$.

In the case that $C = g^{ab+1}$, we have:

$$g^{x_i y_i} g^{v_i} \leftarrow (g^a)^{\sigma_i} \cdot g/C = (g^a)^{\sigma_i} \cdot g/g^{ab+1} = g^{a(\sigma_i - b)} = g^{x_i y_i} \quad \text{and}$$

$$g^{x_j y_j} g^{v_j} \leftarrow \left(g^b\right)^{\sigma_j} \cdot C = \left(g^b\right)^{\sigma_j} \cdot g^{ab+1} = g^{b(\sigma_j + a)} g = g^{x_j y_j} g \ ,$$

which means that in our bare bulletin board $v_i = 0$ and $v_j = 1$.

$\mathcal{S}$ then gives $\mathcal{A}$ the constructed bare bulletin board as input. If $\mathcal{A}$ is able to distinguish which of the above two cases the given bare bulletin board corresponds to, $\mathcal{S}$ will be able to successfully distinguish the two cases for C and hence break Assumption 1. □

THEOREM 3.4 (MAIN THEOREM). *We term the votes that are determined by the adversary "the adversarial votes" and the rest "the non-adversarial votes". Under the DDH assumption and that the ZKP primitive used in the protocol is secure, the DRE-i bulletin board does not reveal anything about the secrecy of the votes other than the tally of non-adversarial votes to an adversary that is able to determine an arbitrary number of votes.*

PROOF. We first restrict our attention to the bare bulletin board and consider the additional ZKP and digital signatures later. To prove that the bare bulletin board does not reveal anything other than the tally of non-adversarial votes, we prove that given only a tally of non-adversarial votes t_{H} and a set of adversarial votes $\{v_k\}_{k \in I_{\mathcal{A}}}$, a bare bulletin board can be simulated which is indistinguishable from any other bare bulletin board with the same non-adversarial vote tally and given adversarial votes. We do this in two steps: first, we show how to simulate a random bare bulletin board with the same t_{H} and given adversarial votes; and second, we show that such a random bare bulletin board is indeed indistinguishable from any other bare bulletin board with the same non-adversarial vote tally and given adversarial votes.

Step 1. Given the adversarial votes $\{v_k\}_{k \in I_{\mathcal{A}}}$, we randomly choose the rest of the votes $\{v_k\}_{k \notin I_{\mathcal{A}}}$ such that their tally is t_{H}. Choosing random values for x_k for all k, we can simulate a bare bulletin board with $\{v_k\}_{k \in I_{\mathcal{A}}}$ as the adversarial votes and $\{v_k\}_{k \notin I_{\mathcal{A}}}$ as the non-adversarial votes.

Step 2. Consider any two possible bare bulletin boards BB and BB' with the same non-adversarial vote tally and given adversarial votes as above. First note that BB and BB' have the same adversarial votes, they have the same adversarial vote tally $t_{\mathcal{A}}$, and since they have the same non-adversarial vote tally t_{H} as well, they have the same total tally $t = t_{\mathrm{H}} + t_{\mathcal{A}}$. We know that any two bulletin boards with the same total tally (and hence BB and BB') differ on an *even* number of votes. Let this vote difference between BB and BB' be $2d$. This means that with d swaps, one can get from one bare bulletin board to the other. Lemma 3.3 guarantees that in all these d steps, under Assumption 1, the two bare bulletin boards involved are indistinguishable to an adversary choosing $\{v_k\}_{k \in I_{\mathcal{A}}}$. Note that the adversarial votes remain fixed between the swaps. Furthermore, Assumption 1 is implied by DDH according to Lemma 3.1. Hence, a standard hybrid argument implies that the original bare bulletin boards (BB and BB') are indistinguishable under the DDH assumption.

A secure 1-of-2 ZKP [Cramer et al. 1994], by definition, does not reveal any information more than the one-bit truth of the statement: whether the ciphertext is a correct encryption of one of the two values. Hence, it does not reveal the secrecy of the encrypted value. The digital signatures serve to prove that all data published on the bulletin board are authentic; they are not related to the secrecy of votes. Hence, we conclude that the theorem holds for the full bulletin board. □

The Theorem 3.4 guarantees the highest possible privacy level for DRE-i. To see this, note that the election tally is public in any election, and hence an adversary controlling a number of adversarial votes inevitably finds out the non-adversarial vote tally. The above theorem ensures that this inevitable knowledge is the only knowledge the adversary gains and in this sense proves the highest privacy level for DRE-i.

A corollary of the above theorem can be stated as below for a passive adversary that does not determine any votes, but only observes the bulletin board.

COROLLARY 3.5 (PRIVACY AGAINST PASSIVE ADVERSARIES). *Under the DDH assumption and that the ZKP primitive used in the protocol is secure, the DRE-i bulletin board does not reveal anything about the secrecy of the votes other than the tally of the votes to a passive adversary.*

Although we have proven that encrypted votes in DRE-i are protected at the highest possible level, it is important to note that breaking encryption is *not* the only way to compromise ballot secrecy. There are other potentially more effective attacks, and security is determined by the weakest link in the chain. For example, an untrustworthy voting interface is one weak link in the chain; a corrupted interface can trivially disclose the voter's secret choice [Karlof et al. 2005; Estehghari and Desmedt 2010]. The setup phase is another potentially weak link. Existing E2E verifiable voting Protocols [Kiayias et al. 2006; Clarkson et al. 2008; Adida 2008; Adida et al. 2009; Chaum et al. 2008b; Chaum et al. 2008a; Adida and Neff 2006; Chaum 2004] generally require a secure setup phase, in which TAs securely generate and distribute key shares. If the setup phase is compromised by attackers, then the *secrecy* of the vote will be breached. These issues also apply to DRE-i. On the other hand, in DRE-i even if the setup phase is completely corrupted, the tallying *integrity* will remain unaffected. This property is claimed for existing E2E verifiable protocols [Chaum et al. 2008a; Adida et al. 2009]. We now explain how this also holds for DRE-i.

We will show DRE-i satisfies all the four integrity requirements as defined in Section 2.2, even if the setup phase is compromised. The use of the CDS technique (i.e., the 1-out-of-n Zero Knowledge Proof) ensures the correct format of the ballot [Cramer et al. 1994], and fulfills the first requirement. The second requirement is satisfied by the voter-initiated auditing (i.e., voter challenge), which is adopted in most verifiable e-voting protocols. The third requirement, that on transmission integrity, is satisfied by the voter being able to check the receipt against the public bulletin board. The fourth requirement, that on tallying integrity, is fulfilled by using homomorphic aggregation combined with the random-factor cancelation, so that anyone is able to verify the tally based on the audit data published on the public bulletin board without relying on any TA. In summary, if an insider attacker attempts to compromise the integrity of the election at any stage, this will most likely be caught by the public because the protocol is E2E verifiable [Adida et al. 2009; Chaum et al. 2008a].

Finally, it is important to ensure that a receipt does not reveal the voter's choice to a coercer. This is a property formally defined as "receipt-freeness" [Delaune et al. 2006]. Previous E2E verifiable voting protocols [Kiayias et al. 2006; Adida 2008; Adida et al. 2009; Chaum et al. 2008b; Chaum et al. 2008a; Adida and Neff 2006; Chaum 2004] generally satisfy this requirement. We explain our protocol conforms to it too. As explained earlier, if the voter chooses to confirm her vote, the receipt does not leak any information about the choice made. If, on the other hand, the voter opts to cancel her vote, the receipt will reveal the selected choice, but the vote will be declared to be a dummy. A dummy vote is of course useless to a would-be coercer.

3.2. Performance evaluation

In DRE-i, we pre-compute all random factors used for encryption before the election with the commitment published on the bulletin board. This pre-computation strategy, combined with the cancellation technique, is one key to realizing the "self-enforcing" property of the voting system. The same strategy also permits pre-computation of all cryptograms, hence optimizing the performance during voting.

We evaluate the system performance by starting from ballot generation. As shown in Table I, we need to compute g^{y_i} for each ballot. At first glance, this is very expensive, taking approximately n multiplications to compute g^{y_1} (recall that n is the total number of ballots, which may be hundreds of thousands). However, note that $g^{y_2} = g^{y_1} \cdot g^{x_2} \cdot g^{x_1}$. More generally, $g^{y_i} = g^{y_{i-1}} \cdot g^{x_i} \cdot g^{x_{i-1}}$ for $i > 1$. Thus, computing g^{y_i}, for $i = 2, 3, \ldots, n$, incurs negligible cost.

For each ballot i, exponentiation is the predominant cost factor. It takes one exponentiation to compute g^{x_i}, one to compute $g^{x_i y_i}$ and four[2] to compute the 1-out-2 ZKP [Cramer et al. 1996] for each no/yes vote, totalling ten exponentiations.

In the ballot casting stage, the computational cost incurred by the DRE machine is small. If we opt for the option of pre-computing all cryptograms before the election, the delay imposed of voting would be almost negligible, since the machine merely needs to print out the pre-computed cryptogram according to the voter's choice and sign it with the digital signature key. Obviously, pre-computing the cryptograms would mean we need to do more preparation work for an election, but that seems a worthwhile trade-off.

The data published on the bulletin board is universally verifiable. Anyone is able to check that the published random public keys g^{x_i} lie within the prime-order group, and that the values of g^{y_i} are correctly computed. To verify the ZKP for the published vote V_i, it is necessary to first validate the order of V_i. This requires an exponentiation (for both the valid and dummy cases); it takes a further four exponentiations to verify the 1-out-of-2 ZKP [Cramer et al. 1994; Cramer et al. 1996]. In total, it takes roughly 5 exponentiations to verify a ZKP. In principle, it suffices for at least one person to verify all the ZKPs in a batch (those who lost the election would be motivated to verify the tally).

3.3. Usability

As explained earlier in Section 2.1, there are three types of users in an e-voting system: ordinary voters, auditors and universal verifiers. In the DRE-i protocol, the auditing is voter-initiated, so an ordinary voter is also an auditor. Of course this does not preclude employing dedicated auditors in an election to perform auditing by casting dummy votes. A universal verifier is anyone in the world who has the technical expertise to verify all data on the public bulletin board in a batch operation.

For an e-voting system to be practically useful, it needs to be "usable". However the notion of "usability" can be abstract and elusive. Here, we define a "usable" cryptographic e-voting system as one that can be used independently by ordinary voters and auditors without requiring any cryptographic knowledge or relying on any trusted software. This is because in practice most people have no knowledge of cryptography and cannot distinguish trustworthy software from untrustworthy software.

The DRE-i protocol assumes a minimum technical background about the voter who may wish to audit the system. The auditing process has been seamlessly integrated into the natural confirm/cancel selection. Every voter can easily audit the ballot by simply choosing the "cancel" button. If a ballot is canceled, the voter just needs to verify that the printed candidate choice (in plain text) on the receipt is the same as that she chose previously. If not, she should lodge a protest immediately. This can be done without requiring any cryptographic knowledge. Of course, the voter needs to know how to open a web browser and check the bulletin board. This basic computer skill is also assumed in other verifiable e-voting protocols [Kiayias et al. 2006; Clarkson et al. 2008; Adida 2008; Adida et al. 2009; Chaum et al. 2008b; Chaum et al. 2008a; Adida and Neff 2006; Chaum 2004].

One may be concerned about the authenticity of the receipt and how to verify this. The data on the receipt should be authentic; otherwise, a dishonest voter may modify the receipt to support a protest that the data fail to match that on the bulletin board. Obviously, if we wish to assume the official receipt paper is physically unforgeable and any tampering with the printed data on the receipt will be visibly evident, then such an attack will not work. However, the assumption of the physical

[2]This is estimated based on using a simultaneous computation technique [Menezes et al. 1996].

unforgeability is difficult to realize. In most cases, a digital signature would be needed, as in other e-voting protocols. With DRE-i, the voter does not have to verify the signature cryptographically; all she needs to do is to ensure the data on the receipt matches that on the bulletin board. A universal verifier will be able to verify all data on the bulletin board in a batch. We assume there is a facility provided at the polling station, say before the exit of the station, to allow voters to check the bulletin board. If the data is found not to match, the voter should raise the matter immediately.

3.4. Dependability and fault tolerance

In DRE-i, the integrity of the election tally depends on the accuracy and completeness of the audit data. The DRE machine directly records votes just as the existing practice in real-world DRE deployment. At the end of the election, the machine reports the tally that it counts internally. But unlike the ordinary DRE machines, in DRE-i, the machine must publish additional audit data to allow public verification of the tally. If the audit data is corrupted (say some ballots are lost), then the integrity of the tally will be lost and the universal verification will fail. In that case, the system essentially degenerates to the existing unverifiable DRE-based e-voting. Here, we have considered the assurance of tallying integrity in the most stringent case, ensuring that every vote must be counted.

In a practical election, it is desirable to handle system faults gracefully. When the audit data have been found to be partially corrupted, instead of merely degenerating to unverifiable e-voting, we can extend the DRE-i protocol to provide a fail-safe feature.

Fail-safe DRE-i. Consider a case where a small subset L of ballots are found missing from (or to be corrupted on) the public bulletin board. (The number of the missing ballots should be insufficient to change the election outcome; otherwise, the act of error recovery may not be meaningful.) We assume the DRE machine still maintains the x_i secrets in the protected memory of the tamper-resistant module. To allow the tallying verification to proceed, one trivial solution is to re-publish the cryptograms of the subset L of ballots as if they were "dummy" votes. The no-votes ($g^{x_i y_i}$ for $i \in L$) are then included into the tallying process, hence allowing the tally of the remaining ballots to be verified. However, if a voter holds a receipt of a missing ballot, the secrecy of that ballot will be lost. Hence, instead of publishing individual cryptograms, it is more secure to publish just one aggregate value: namely, $A = \sum_{i \in L} g^{x_i y_i}$ together with some cryptographic proofs to show that A is in the correct format (details can be found in Appendix A). Thus, the information leakage is minimal. An attacker in possession of some (not *all*) receipts cannot learn anything about the missing ballots. In the worst case when the attacker is able to collect *all* receipts of the missing ballots, the only thing he can learn is the tally of the missing ballots, not any individual vote.

Distributed DRE-i. The fail-safe mechanism works on the condition that the x_i secrets are available. If the DRE machine is physically damaged or lost, such an error recovery procedure may no longer be possible. In order to ensure system robustness, it is desirable to implement DRE-i in a distributed way, as we explain below.

Figure 6 shows one possible implementation of the DRE-i system using a distributed client-server architecture. The system consists of touch-screen DRE clients and a back-end server cluster. The DRE client interacts with the voter and records the vote directly as usual. The server cluster consists of n servers and implements a k/n threshold control. The setup phase works based on a proactive secret sharing scheme [Herzberg et al. 1995]. Each server generates a random polynomial of degree $t-1$ and distributes n shares to all servers. All n polynomials are then added up with no single server knowing the aggregate secret. Let the aggregate secret be x_i. The process can be repeated for all x_i where $i = 1, \ldots, n$. Subsequently, the server cluster jointly compute g^{x_i} by performing secret reconstruction on the exponent [Herzberg et al. 1995], such that no single server learns the exponent x_i. To finish the setup phase, the server cluster publishes all the g^{x_i} values on the bulletin board as commitment. During the voting phrase, the DRE client queries the shares from k honest severs in the server cluster through secure channels and reconstructs the x_i secret. With x_i, the client is able to compute the cryptogram and print the receipt accordingly. The DRE client erases the transient x_i secret immediately after its use.

USENIX Journal of Election Technology and Systems (JETS)
Volume 2, Number 3 • July 2014
www.usenix.org/jets/issues/0203

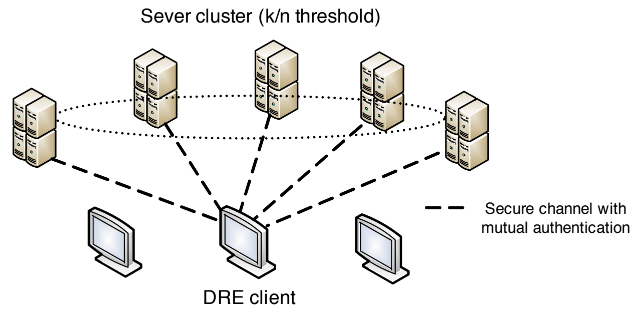
Sever cluster (k/n threshold)
Secure channel with mutual authentication
DRE client

Fig. 6. A distributed implementation of the DRE-i system

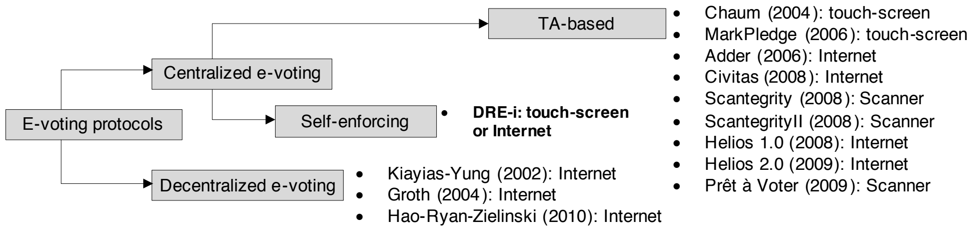
E-voting protocols
Centralized e-voting
TA-based
Self-enforcing
Decentralized e-voting
DRE-i: touch-screen or Internet
Chaum (2004): touch-screen
MarkPledge (2006): touch-screen
Adder (2006): Internet
Civitas (2008): Internet
Scantegrity (2008): Scanner
ScantegrityII (2008): Scanner
Helios 1.0 (2008): Internet
Helios 2.0 (2009): Internet
Prêt à Voter (2009): Scanner
Kiayias-Yung (2002): Internet
Groth (2004): Internet
Hao-Ryan-Zielinski (2010): Internet

Fig. 7. Categorization of e-voting protocols

A further practical strategy in distributing the implementation of DRE-i is to divide the national-scale tallying into a set of smaller-scale tallying processes, each implementing an independent DRE-i system. This is consistent with many real-world elections where tallies are calculated at relatively small (say county or precinct) scales and then added up.

4. RELATED WORK AND COMPARISON

In this section, we compare DRE-i with previous DRE-based voting protocols in a local supervised voting environment.

4.1. Categorization of e-voting

First of all, we take a broad view at the existing e-voting protocols. There are generally two categories of cryptographic voting protocols: decentralized and centralized (see Figure 7). The former includes boardroom voting protocols due to Kiayias-Yung [Kiayias and Yung 2002], Groth [Groth 2004] and Hao-Ryan-Zieliński [Hao et al. 2010]. The latter includes a wide range of E2E verifiable protocols: e.g., Adder [Kiayias et al. 2006], Civitas [Clarkson et al. 2008], Helios [Adida 2008; Adida et al. 2009], Scantegrity [Chaum et al. 2008b], Scantegrity II [Chaum et al. 2008a], Prêt à Voter [Ryan et al. 2009], MarkPledge [Adida and Neff 2006] and Chaum's visual cryptographic scheme [Chaum 2004]. Existing E2E verifiable voting protocols are often designed to use different voting interfaces: e.g., a web browser [Kiayias et al. 2006; Adida 2008; Adida et al. 2009], an optical scanner [Chaum et al. 2008b; Chaum et al. 2008a; Ryan et al. 2009], and a touch-screen DRE [Adida and Neff 2006; Chaum 2004]. They are also designed to suit two different scenarios: local voting [Chaum et al. 2008b; Chaum et al. 2008a; Ryan et al. 2009; Adida and Neff 2006; Chaum 2004] and remote voting [Kiayias et al. 2006; Clarkson et al. 2008; Adida 2008; Adida et al. 2009]. All these E2E verifiable protocols require external tallying authorities to decrypt and tally the submitted votes. Hence, they belong to the category of "TA-based e-voting" (see Figure 7).

The proposed DRE-i protocol provides the same E2E verifiability, but *without* involving any external tallying authorities. This puts DRE-i in a new category, which we call "self-enforcing e-voting".

Table IV. Comparison between DRE-i and ordinary (black-box) e-voting in local DRE-based voting.

	DRE-i	**Ordinary (black-box) DRE machine**
External tallying authorities	Not required	Not required
Ballot casting assurance	Voter-initiated auditing	**No assurance**
Transmission integrity	Check receipt with Public Bulletin Board	**No assurance**
Tallying integrity	Accurate audit data	**No assurance**
Ballot secrecy	Voting interface, **setup** and **DRE not leaking random factors (or pre-computed cryptograms)**	Voting interface
Voter privacy	Anonymity	Anonymity
Receipt	Yes, but cannot be used for coercion	**No receipt**
Availability	Dependent on system robustness	Dependent on system robustness
Tamper-resistant module	**Needed for key management**	Not required
Crypto-awareness of voter	Not required	Not required
Crypto-awareness of auditor	Not required	**Public auditing is impossible**
Crypto-awareness of verifier	Required	**Universal verification is impossible**

Notes: Major differences are highlighted in bold face.

4.2. Comparison with unverifiable DRE

We first compare DRE-i with the unverifiable (or black-box) DRE machines that have been widely deployed around the world. The results of the comparison are summarized in Table IV. We explain the main differences below.

Integrity. The primary advantage of DRE-i lies in the additional "-i" in the name: i.e., its integrity. In DRE-i, a voter can verify that her ballot is recorded to the correct candidate through voter-initiated auditing (i.e., *ballot casting integrity*). She can further verify that the recorded ballot is correctly transmitted to the tallying unit by checking the receipt against the public bulletin board (i.e., *transmission integrity*). Finally, every voter is able to verify the integrity of the tally based on the public audit data published on the bulletin board (i.e., *ballot tallying integrity*). These essential verification procedures are missing in the currently deployed DRE machines.

Ballot secrecy and voter privacy. In both systems, the touch-screen interface can violate the secrecy of the vote. However, it does not know the voter's real identity. Hence, the voter's privacy is protected through anonymity. In DRE-i, the system requires an additional setup phase, which pre-fixes random factors used for encryption. The secrecy of the random factors needs to be securely protected, as well as the pre-computed cryptograms (if the pre-computation option is enabled).

Receipt. In DRE-i, the machine prints out a receipt, which the voter can verify against a public bulletin board. The receipt does not reveal how a voter had voted, but allows the voter to check if her vote has indeed been included into the tallying process. By contrast, the ordinary DRE machine does not provide any receipt. If the ballot is missing or miscounted, the voter would not be able to know.

Tamper-resistant module. In DRE-i, a tamper-resistant module (e.g., smart card or TPM chip) is needed to securely manage sensitive key material, including the private signing key, the x_i secrets and pre-computed cryptograms (if any). This follows the standard industry practice for key management [Anderson 2008]. However, an ordinary DRE machine normally does not require a tamper-resistant module, as no cryptography is used.

Usability. As compared to the ordinary DRE, the usability in DRE-i degrades slightly due to the additional opportunity provided to the voter to check the receipt against the bulletin board. On the other hand, the receipts allow public verification of the tallying integrity, which is not possible with ordinary DRE machines. Hence, the trade-off seems worthwhile for the improved assurance on integrity.

4.3. Comparison with previous DRE-based E2E verifiable schemes

Next, we compare DRE-i with two previous DRE-based E2E verifiable voting protocols: Mark-Pledge [Adida and Neff 2006] and Chaum's visual crypto scheme [Chaum 2004]. The results of this comparison are summarized in Table V.

Table V. Comparison between DRE-i and related E2E verifiable voting protocols for local DRE-based voting

	DRE-i	**Local DRE-based protocols** [Adida and Neff 2006; Chaum 2004]
External tallying authorities	Not required	**Required**
Ballot casting assurance	Voter-initiated auditing	Voter-initiated auditing
Transmission integrity	Check receipt with Public Bulletin Board	Check receipt with Public Bulletin Board
Tallying integrity	Accurate audit data	Accurate audit data, and **TA not losing keys**
Ballot secrecy	Voting interface, setup, DRE not leaking random factors (or **pre-computed cryptograms**)	Voting interface, setup, DRE not leaking random factors and **TA not leaking private keys**
Voter privacy	Anonymity	Anonymity
Receipt-freeness	Yes	Yes
Availability	Dependent on system robustness	Dependent on system robustness and **TA not losing keys**
Tamper-resistant module	Needed for key management	Needed for key management
Crypto-awareness of voter	Not required	**Required**
Crypto-awareness of auditor	Not required	**Required**
Crypto-awareness of verifier	Required	Required

Note: Major differences are highlighted in bold face.

Integrity. DRE-i provides the same E2E verifiability as MarkPledge [Adida and Neff 2006] and Chaum's scheme [Chaum 2004], but without involving any external tallying authorities. To guarantee the tallying integrity, all three protocols require the audit data as published on the bulletin board be accurate and complete. In MarkPledge [Adida and Neff 2006] and Chaum's scheme [Chaum 2004], when the election is finished, the audit data published on the bulletin board must be first decrypted by external tallying authorities before any verification is possible. This requires that the tallying authorities' private keys be available at the decryption and tallying phase; otherwise, the tally cannot be verified.

Ballot secrecy and voter privacy. In all three protocols, if the voting interface is corrupted, the secrecy of the ballot is lost. In addition, if the setup process (be it the pre-computation procedure in DRE-i or the secret sharing setup in TA-based e-voting) is compromised, the secrecy of the ballot is lost too. In DRE-i, all random factors are pre-determined before the election with commitment published on the bulletin board. The secrecy of the pre-determined random factors needs to be securely protected, which can be realized by storing them in the secure memory of a tamper-resistant module [Anderson 2008]. In MarkPledge [Adida and Neff 2006] and Chaum's scheme [Chaum 2004], the random factors are generated by the DRE machine on the fly during the encryption of ballots. Similarly, the secrecy of those random factors needs to be protected. It is critically important that the random factors are generated honestly from a secure random number generator. If the random number generator is corrupted, all random factors are effectively leaked. Consequently, the secrecy of all encrypted votes is trivially lost (which is orthogonal to the security of the TAs' private keys). In DRE-i, the choice of pre-computing random factors before the election is based on the assumption that the environment in the setup phase is more controllable than that in the field deployment on the election day, hence the random number generator is less likely to be corrupted. Finally, MarkPledge [Adida and Neff 2006] and Chaum's scheme [Chaum 2004] assume the external TAs do not leak their private keys; otherwise, the secrecy of the votes is compromised.

Availability. All three protocols depend on the robustness of hardware and software to ensure availability of functionality. In MarkPledge [Adida and Neff 2006] and Chaum's scheme [Chaum 2004], the tallying process is entirely reliant on the external tallying authorities. All data is encrypted under the authorities' keys and there is usually no mechanism of directly recording votes by the machine. However, this dependence on external authorities may lead to an additional, in fact a catastrophic, failure mode. Human nature being what it is, when a security system critically depends on a few selected human beings as authorities, they may form the weakest link in the system [Anderson 2008]. Suppose that when the national voting is finished, tallying authorities claim that their private keys are lost [Karlof et al. 2005] (e.g., as victims of targeted attacks or as the authorities claim such is the case). All the data on the bulletin board will be useless, and the whole election may have to be aborted as a result. (Recall that in the Helios election [Adida et al. 2009], all the tallying authorities' private keys were centrally backed up at a trusted third party to ensure availability.)

Tamper-resistant module. In DRE-i, a tamper-resistant module is required to securely manage sensitive key material, including the private signing key, the pre-determined random factors and the pre-computed cryptograms (if any). In MarkPledge [Adida and Neff 2006] and Chaum's scheme [Chaum 2004], a tamper-resistant module is also required, for safeguarding the private signing key, and additionally, for protecting the ephemeral random factors that are generated as part of the encryption process. Because of the pre-generation of random factors, DRE-i requires more memory in the tamper-resistant module than the other two schemes.

Usability. In MarkPledge, the voter needs to supply a "short-string challenge" [Adida and Neff 2006], which demands special cryptographic knowledge. To address this limitation, the designers of the MarkPledge system suggest having a trusted third party at the polling station to issue the challenges on the voters' behalf. Unfortunately, this means a voter will not be able to *independently* perform auditing. In Chaum's visual crypto scheme [Chaum 2004], the voter needs to choose one of the two transparencies for auditing. However, this implicitly assumes that voters understand how visual cryptography works. In practice, not many voters can grasp the concept of visual cryptography [Karlof et al. 2005]. As explained in Section 3.3, by design, DRE-i is free from these issues. In all three protocols, a universal verifier who has necessary computing expertise is required to verify the audit data published on the bulletin board in one batch operation.

4.4. Comparison with alternative designs

The design of the DRE-i protocol is motivated by the observation that since the touch-screen DRE learns the voter's choice directly and generates random factors for encryption on its own, the involvement of external tallying authorities does not seem strictly necessary for realizing the E2E verifiability. It is worth stressing that there are several ways to construct a "self-enforcing e-voting" protocol and DRE-i is just one of them. While it is beyond the scope of this paper to discuss all possible alternative designs, we will briefly describe one scheme and then compare it with DRE-i.

In order to avoid the involvement of external tallying authorities, one straightforward solution is to adapt the existing TA-based e-voting protocols by merging the functions of the DRE with those of the TAs. For instance, the system may use a single TA and keep the private key in the protected memory of the tamper-resistant module in the DRE machine. All votes are encrypted under the TA's public key on the fly using the standard ElGamal encryption [Kiayias et al. 2006; Clarkson et al. 2008; Adida 2008; Adida et al. 2009] with ciphertext printed on the receipt and also published on the bulletin board. At the end of the election, the DRE machine decrypts the published ciphertext in a verifiable way.

The DRE-i protocol is better than the above alternative design in two main aspects. The first is efficiency. In DRE-i, the ciphertext for the no-vote ($g^{x_i y_i}$) and the yes-vote ($g^{x_i y_i} \cdot g$) consists of a single group element. It takes merely one exponentiation to compute it. By comparison, using the standard ElGamal encryption, it takes two exponentiations to encrypt a vote and the resultant ciphertext consists of two group elements. Second, in DRE-i, all the random factors used in the encryption are fixed before the election with commitment published on the bulletin board, while they are determined on the fly during voting in the alternative design. Our assumption is that the environment in the setup phase is more controllable than that in the field deployment on the election day. Furthermore, the publication of all random public keys (g^{x_i}) before the election gives the public an opportunity to verify the distribution of the values, gaining some measure about the randomness. Another practical advantage of pre-fixing the random factors is to allow pre-computing the cryptograms, thus reducing the latency in voting.

5. CONCLUSION

E2E verifiable e-voting protocols have been extensively studied in the past twenty years, but the real-world deployment of those protocols has been limited. Our hypothesis is that a key obstacle to the practical deployment is the existing E2E verifiable voting protocols' universal dependence on a set of trustworthy tallying authorities to administer the tallying process. Previous trial experience has shown that implementing such authorities is not an easy task in practice. In this paper, we focus on studying local touch-screen DRE-based elections. First of all, we observe that since the DRE machine learns the voter's choice directly and generates its own random factors for encryption, the involvement of external tallying authorities does not seem strictly necessary for achieving the E2E verifiability. Based on this observation, we propose a self-enforcing e-voting protocol called DRE-i, which provides the same E2E verifiability as previous schemes but without involving any tallying authorities. By comparing DRE-i with related voting systems, we demonstrate encouraging improvements in several aspects, including security, efficiency and usability. This shows that "self-enforcing e-voting", as a new paradigm, has promising potential for further research. In future research, we plan to extend our study to remote e-voting and also to accommodate more complex voting schemes, such as Single Transferable Vote (STV).

ACKNOWLEDGMENTS

We would like to thank the editors and the anonymous reviewers of USENIX JETS for constructive and pertinent comments. We also thank Ross Anderson, Joseph Bonneau and other members of the security group at the Computer Lab, University of Cambridge, for helpful discussions at the early stage of this research work.

References

Ben Adida. 2008. Helios: web-based open-audit voting. In *Proceedings of the 17th USENIX Security Symposium*. USENIX Association, 335–348.

Ben Adida, Olivier De Marneffe, Olivier Pereira, and Jean-Jacques Quisquater. 2009. Electing a university president using open-audit voting: Analysis of real-world use of Helios. In *Proceedings of the Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE)*. USENIX Association.

Ben Adida and C Andrew Neff. 2006. Ballot casting assurance. In *Proceedings of the Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE)*. USENIX Association.

R Michael Alvarez, Gabriel Katz, and Julia Pomares. 2011. The impact of new technologies on voter confidence in Latin America: evidence from e-voting experiments in Argentina and Colombia. *Journal of Information Technology & Politics* 8, 2 (2011), 199–217.

Ross Anderson. 2008. *Security engineering: a guide to building dependable distributed systems* (second edition ed.). John Wiley & Sons.

Josh Benaloh. 1987. *Verifiable secret-ballot elections*. Ph.D. Dissertation. Yale University.

Josh Benaloh. 2007. Ballot casting assurance via voter-initiated poll station auditing. In *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology (EVT)*. USENIX Association.

Jarrett Blanc. 2007. Challenging the norms and standards of election administration: electronic voting. In *International Foundation For Electoral Systems Report*. 11–19.

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

David Chaum, Richard Carback, Jeremy Clark, Aleksander Essex, Stefan Popoveniuc, Ronald L Rivest, Peter YA Ryan, Emily Shen, and Alan T Sherman. 2008a. Scantegrity II: end-to-end verifiability for optical scan election systems using invisible ink confirmation codes. In *Proceedings of the USENIX/ACCURATE Electronic Voting Workshop (EVT)*. USENIX Association, 1–13.

David Chaum, Aleks Essex, Richard Carback, Jeremy Clark, Stefan Popoveniuc, Alan Sherman, and Poorvi Vora. 2008b. Scantegrity: End-to-end voter-verifiable optical-scan voting. *IEEE Security & Privacy* 6, 3 (2008), 40–46.

David Chaum and Torben Pryds Pedersen. 1993. Transferred cash grows in size. In *Advances in Cryptology – EUROCRYPT'92 (Lecture Notes in Computer Science)*, Vol. 658. Springer, 390–407.

Michael R Clarkson, Stephen Chong, and Andrew C Myers. 2008. Civitas: Toward a secure voting system. In *Proceedings of IEEE Symposium on Security and Privacy*. IEEE, 354–368.

Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. 1994. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology – CRYPTO'94 (Lecture Notes in Computer Science)*, Vol. 839. Springer, 174–187.

Ronald Cramer, Matthew Franklin, Berry Schoenmakers, and Moti Yung. 1996. Multi-authority secret-ballot elections with linear work. In *Advances in Cryptology – EUROCRYPT'96 (Lecture Notes in Computer Science)*, Vol. 1070. Springer, 72–83.

Stephanie Delaune, Steve Kremer, and Mark Ryan. 2006. Coercion-resistance and receipt-freeness in electronic voting. In *Proceedings of the 19th IEEE Computer Security Foundations Workshop (CSFW)*. IEEE, 28–39.

Saghar Estehghari and Yvo Desmedt. 2010. Exploiting the client vulnerabilities in Internet e-voting systems: Hacking Helios 2.0 as an example. In *Proceedings of Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE)*. USENIX Association.

Amos Fiat and Adi Shamir. 1987. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology – CRYPTO'86 (Lecture Notes in Computer Science)*, Vol. 263. Springer, 186–194.

Jens Groth. 2004. Efficient maximal privacy in boardroom voting and anonymous broadcast. In *Proceedings of Financial Cryptography (Lecture Notes in Computer Science)*, Vol. 3110. Springer, 90–104.

Feng Hao, Dylan Clarke, and Carlton Shepherd. 2013. Verifiable classroom voting: Where cryptography meets pedagogy. In *Proceedings of the 21st Security Protocols Workshop (SPW) (Lecture Notes in Computer Science)*, Vol. 8263. Springer, 245–254.

Feng Hao, Brian Randell, and Dylan Clarke. 2012. Self-enforcing electronic voting. In *Proceedings of the 20th Security Protocols Workshop (SPW) (Lecture Notes in Computer Science)*, Vol. 7622. Springer, 23–31.

Feng Hao, Peter YA Ryan, and P Zieliński. 2010. Anonymous voting by two-round public discussion. *IET Information Security* 4, 2 (June 2010), 62–67.

Feng Hao and Piotr Zieliński. 2006. A 2-round anonymous veto protocol. In *Proceedings of the 14th International Workshop on Security Protocols (Lecture Notes in Computer Science)*, Vol. 5087. Springer, 202–211.

Amir Herzberg, Stanisław Jarecki, Hugo Krawczyk, and Moti Yung. 1995. Proactive secret sharing or: How to cope with perpetual leakage. In *Advances in Cryptology – CRYPT0'95 (Lecture Notes in Computer Science)*, Vol. 963. Springer, 339–352.

David Jefferson, A Rubin, Barbara Simons, and David Wagner. 2004. A security analysis of the secure electronic registration and voting experiment (SERVE). http://servesecurityreport.org. (2004).

Chris Karlof, Naveen Sastry, and David Wagner. 2005. Cryptographic Voting Protocols: A Systems Perspective.. In *Proceedings of the 14th USENIX Security Symposium*, Vol. 5. USENIX Association, 33–50.

Aggelos Kiayias, Michael Korman, and David Walluck. 2006. An Internet voting system supporting user privacy. In *Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC)*. 165–174.

Aggelos Kiayias and Moti Yung. 2002. Self-tallying elections and perfect ballot secrecy. In *Proceedings of Public Key Cryptography (PKC) (Lecture Notes in Computer Science)*, Vol. 2274. Springer, 141–158.

Tadayoshi Kohno, Adam Stubblefield, Aviel D Rubin, and Dan S Wallach. 2004. Analysis of an electronic voting system. In *Proceedings of IEEE Symposium on Security and Privacy*. IEEE, 27–40.

Robert Krimmer, Stefan Triessnig, and Melanie Volkamer. 2007. The development of remote e-voting around the world: A review of roads and directions. In *Proceedings of the 1st International Conference on E-voting and Identity (VOTE-ID)*. 1–15.

Tomasz Küsters, Ralf andTruderung and Andreas Vogt. 2010. Accountability: definition and relationship to verifiability. In *Proceedings of the 17th ACM conference on Computer and communications security (CCS)*. ACM, 526–535.

Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. 1996. *Handbook of applied cryptography*. CRC press.

Rebecca T Mercuri. 2001. *Electronic vote tabulation checks and balances*. Ph.D. Dissertation. University of Pennsylvania.

Wolter Pieters. 2011. How devices transform voting. In *Innovating Government*. Vol. 20. Springer, 439–452.

Stefan Popoveniuc, John Kelsey, Andrew Regenscheid, and Poorvi Vora. 2010. Performance requirements for end-to-end verifiable elections. In *Proceedings of the 2010 international conference on Electronic voting technology/workshop on trustworthy elections (EVT/WOTE)*. USENIX Association, 1–16.

Peter YA Ryan, David Bismark, James Heather, Steve Schneider, and Zhe Xia. 2009. Prêt à voter: a voter-verifiable voting system. *IEEE Transactions on Information Forensics and Security* 4, 4 (2009), 662–673.

Alan T Sherman, Aryya Gangopadhyay, Stephen H Holden, George Karabatis, A Gunes Koru, Chris M Law, Donald F Norris, John Pinkston, Andrew Sears, and Dongsong Zhang. 2006. An examination of vote verification technologies: Findings and experiences from the Maryland Study. In *Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop (EVT)*. USENIX Association.

Michael Steiner, Gene Tsudik, and Michael Waidner. 1996. Diffie-Hellman key distribution extended to group communication. In *Proceedings of the 3rd ACM conference on Computer and communications security (CCS)*. ACM, 31–37.

Douglas R Stinson. 2006. *Cryptography: theory and practice* (third edition ed.). CRC press.

VoteHere. 2002. Network Voting System Standards (NVSS). (2002). Public Draft 2.

Scott Wolchok, Eric Wustrow, J Alex Halderman, Hari K Prasad, Arun Kankipati, Sai Krishna Sakhamuri, Vasavya Yagati, and Rop Gonggrijp. 2010. Security analysis of India's electronic voting machines. In *Proceedings of the 17th ACM conference on Computer and communications security (CCS)*. ACM, 1–14.

Richard Wolf. 2008. Voting equipment changes could get messy on Nov. 4. (29 October 2008). Available at http://www.usatoday.com/news/politics/election2008/2008-10-28-votingequipment_N.htm#table.

A. FAIL-SAFE DRE-I AND SECURITY PROOFS

We use the same domain parameters, (p,q,g), as those defined in Section 2. Assume at the end of the election, a subset L of ballots are found to be missing (or corrupted) on the bulletin board. To allow the public to verify the tally of the remaining ballots, the DRE publishes $A = g^{\sum_{i\in L} x_i y_i}$ and proves non-interactively that A is in the right format without revealing the secrecy of each individual $g^{x_i y_i}$ term as follows.

(1) DRE chooses $r \in_R [1, q-1]$ and publishes $X_i = (g^{x_i})^r$ and $Z_i = (g^{x_i y_i})^r$ for all $i \in L$.
(2) DRE publishes ZKPs of Equality (based on Chaum-Pedersen's technique [Chaum and Pedersen 1993]) for all $i \in L$ to prove that the discrete logarithm of X_i with respect to base g^{x_i} is equal to the discrete logarithm of X_j with respect to base g^{x_j}, where j is the index in L immediately greater than i. These ZKPs guarantee that for any $i, j \in L$ $(i \neq j)$, $X_i = (g^{x_i})^r$ and $X_j = (g^{y_i})^r$ have the same exponent r.
(3) DRE publishes ZKPs of Equality [Chaum and Pedersen 1993] for all $i \in L$ to prove that (X_i, g^{y_i}, Z_i) forms a DDH tuple. This is equivalent to proving that the discrete logarithm of Z_i with respect to base g^{y_i} is equal to the discrete logarithm of X_i with respect to base g. These ZKPs guarantee that for all $i \in L$, $Z_i = (g^{x_i y_i})^r$ has the same exponent r.
(4) DRE publishes a ZKP of Equality [Chaum and Pedersen 1993] to prove that the discrete logarithm of $\prod_{i\in L} Z_i$ with respect to base A is equal to the discrete logarithm of an arbitrary X_i $(i \in L)$ with respect to base g^{x_i}. It suffices to choose i to be the first index in L. This ZKP guarantees that A is indeed represented in the form of $g^{\sum_{i\in L} x_i y_i}$.

These published data guarantee that A is in the correct representation. Therefore A can be subsequently included into the tallying process to rectify the effects of missing ballots. Among the published data, the ZKP of Equality does not leak anything more than one bit information about the truth of the statement: the two discrete logarithms are equal [Chaum and Pedersen 1993]. However, the process also involves publishing additional data: X_i and Z_i for all $i \in L$. In the following, we will prove that the X_i and Z_i values will not affect the secrecy of each individual $g^{x_i y_i}$. In other words, the result in Theorem 3.4 still holds. We consider the extreme case when the available data to an adversary is the maximum: i.e., L is a whole set rather than a subset (obviously, $|L| > 1$). We will prove Theorem 3.4 holds even in this extreme case. First of all, we define a variant of the DDH assumption as below.

ASSUMPTION 2 (3DDH VARIANT). *For a generator g and randomly chosen a, b, and c, given a tuple $(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc}, C)$ in which C is either g^{ab} or g^{ab+1}, it is hard to decide whether $C = g^{ab}$ or $C = g^{ab+1}$.*

LEMMA A.1. *Assumption 2 is implied by the DDH assumption.*

PROOF. First, note that Steiner, Tsudik, and Waidner [Steiner et al. 1996] have proven that DDH is equivalent to the generalized DDH assumption. An instance of the generalized DDH assumption is the three-party DDH assumption (3DDH) which states that for a generator g and randomly chosen a, b, and c, given a tuple $(g, g^a, g^b, g^c, g^{ab}, g^{ac}, g^{bc})$, it is hard to distinguish g^{abc} from random. An equivalent formulation of the 3DDH assumption is as follows: for a generator g and randomly chosen a, b, and c, given a tuple $(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc})$, it is hard to distinguish g^{ab} from random. This can be easily seen by considering g^c as the generator in the original formulation.

Now we prove that the latter formulation of 3DDH implies Assumption 2. Similar to the proof of Lemma 3.1, consider the following tuples:

$$\begin{aligned}&(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc}, g^{ab}),\\&(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc}, R),\\&(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc}, R'g), \quad \text{and}\\&(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc}, g^{ab}g),\end{aligned}$$

for random a, b, c, R, and R'. 3DDH guarantees that the first and second tuples are indistinguishable. The second and third tuples have the exact same distribution and hence are indistinguishable. 3DDH also guarantees that the third and fourth tuples are indistinguishable. Hence, the first and fourth tuples, i.e. $(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc}, g^{ab})$ and $(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc}, g^{ab+1})$ are indistinguishable. $\square$

LEMMA A.2. *Consider two* failsafe *DRE-i elections in which all the votes are exactly the same except for two votes v_i and v_j which are swapped between the two elections. Under Assumption 2, the bare bulletin boards of the above two elections are indistinguishable to an adversary that determines an arbitrary number of the votes other than v_i and v_j.*

PROOF. Let us assume w.l.o.g. that $i < j$. If $v_i = v_j$, the lemma holds trivially. In the following we give a proof for $v_i \neq v_j$.

Let us assume there is an adversary $\mathcal{A}$ that first chooses an arbitrary number of the votes other than v_i and v_j, and eventually distinguishes the two elections. We construct an algorithm $\mathcal{S}$ that uses $\mathcal{A}$ to break Assumption 2.

Given a tuple $(g, g^a, g^b, g^c, g^{ac}, g^{bc}, g^{abc}, C)$, where C equals either g^{ab} or g^{ab+1}, $\mathcal{S}$ sets up the bulletin board with the generator g as follows. Let $I = \{1, \ldots, n\} \setminus \{i, j\}$.

g^{x_k} and g^{y_k} are set up in the same way as the proof of Lemma 3.3. First, $\mathcal{S}$ chooses $n-2$ random values x_k for all $k \in I$. $\mathcal{S}$ sets $g^{x_i} \leftarrow g^a$, $g^{x_j} \leftarrow g^b$, and calculates g^{x_k} for all $k \in I$. Note that we implicitly have $x_i = a$ and $x_j = b$. Let $s_1 = \sum_{k<i} x_k$, $s_2 = \sum_{i<k<j} x_k$, and $s_3 = \sum_{k>j} x_k$. $\mathcal{S}$ also calculates s_1, s_2, and s_3 and then computes $\sigma_i = s_1 - s_2 - s_3$ and $\sigma_j = s_1 + s_2 - s_3$.

Now given all g^{x_k}, all g^{y_k} can be computed accordingly. Note that we implicitly have:

$$y_i = \sum_{k<i} x_k - \sum_{k>i} x_k = s_1 - (s_2 + b + s_3) = \sigma_i - b$$

$$y_j = \sum_{k<j} x_k - \sum_{k>j} x_k = (s_1 + a + s_2) - s_3 = \sigma_j + a$$

Next, $\mathcal{S}$ simulates $g^{x_k r}$ and $g^{x_k y_k r}$ as follows. It sets $g^{x_i r} \leftarrow g^{ac}$ and $g^{x_j r} \leftarrow g^{bc}$; that is, we implicitly have $r = c$. For all $k \in I$, it sets $g^{x_k r} \leftarrow (g^c)^{x_k}$. Then it sets

$$g^{x_i y_i r} \leftarrow \left(g^{ac}\right)^{\sigma_i} / g^{abc} = g^{a(\sigma_i - b)c} \quad \text{and} \quad g^{x_j y_j r} \leftarrow \left(g^{bc}\right)^{\sigma_j} \cdot g^{abc} = g^{b(\sigma_j + a)c} \ .$$

In general, for any $k = 1, \ldots, n$, we define $\sigma_k = \sum_{\substack{\ell<k\\ \ell\neq i,j}} x_\ell - \sum_{\substack{\ell>k\\ \ell\neq i,j}} x_\ell$. Now we have:

$$\forall k \in I: \quad y_k = \sum_{\ell<k} x_\ell - \sum_{\ell>k} x_\ell = \pm a \pm b + \sum_{\substack{\ell<k\\ \ell\neq i,j}} x_\ell - \sum_{\substack{\ell>k\\ \ell\neq i,j}} x_\ell = \pm a \pm b + \sigma_k \ ,$$

where depending on k, we have either a plus or a minus sign in front of a and b and σ_k is known. Hence $\{g^{x_k y_k r}\}_{k\in I}$ can be simulated as

$$g^{x_k y_k r} \leftarrow \left(g^{\pm ac} g^{\pm bc} \left(g^c\right)^{\sigma_k}\right)^{x_k} = g^{x_k(\pm a \pm b + \sigma_k)c} \ .$$

Table VI. The simulated bare bulletin board in the proof of Lemma A.2

$\mathbf{k}$	$\mathbf{g^{x_k}}$	$\mathbf{g^{y_k}}$	$\mathbf{g^{x_k r}}$	$\mathbf{g^{x_k y_k r}}$	$\mathbf{g^{x_k y_k} g^{v_k}}$
1	g^{x_1}	$1/\prod_{k>1} g^{x_k}$	$(g^c)^{x_1}$	$\left((g^c)^{\sigma_1} / \left(g^{ac} g^{bc}\right)\right)^{x_1}$	$(g^{x_1})^{y_1} g^{v_1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	g^a	$\prod_{k<i} g^{x_k} / \prod_{k>i} g^{x_k}$	g^{ac}	$(g^{ac})^{\sigma_i} / g^{abc}$	$(g^a)^{\sigma_i} \cdot g/C$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
j	g^b	$\prod_{k<j} g^{x_k} / \prod_{k>j} g^{x_k}$	g^{bc}	$\left(g^{bc}\right)^{\sigma_j} \cdot g^{abc}$	$\left(g^b\right)^{\sigma_j} \cdot C$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	g^{x_n}	$\prod_{k<n} g^{x_k}$	$(g^c)^{x_n}$	$\left(g^{ac} g^{bc} (g^c)^{\sigma_n}\right)^{x_n}$	$(g^{x_n})^{y_n} g^{v_n}$

$\mathcal{S}$ sets up the last column of the bare bulletin board similar to the proof of Lemma 3.3. $\mathcal{A}$ chooses a set of votes $\{v_k\}_{k \in I_{\mathcal{A}}}$ for the set of indexes $I_{\mathcal{A}} \subseteq I$. Let us consider some arbitrary set of votes $\{v_k\}_{k \in I \setminus I_{\mathcal{A}}}$. $\mathcal{S}$ can calculate $g^{x_k y_k}$ for all $k \in I$, since it knows x_k and g^{y_k}. Hence, it can calculate $g^{x_k y_k} g^{v_k}$ for all $k \in I$. For $k = i, j$, $\mathcal{S}$ sets

$$g^{x_i y_i} g^{v_i} \leftarrow (g^a)^{\sigma_i} \cdot g/C \quad \text{and} \quad g^{x_j y_j} g^{v_j} \leftarrow \left(g^b\right)^{\sigma_j} \cdot C \, .$$

Now the calculation of the entire bare bulletin board is complete. Table VI shows the simulated bare bulletin board.

In the case that $C = g^{ab}$, we have:

$$g^{x_i y_i} g^{v_i} \leftarrow (g^a)^{\sigma_i} \cdot g/C = (g^a)^{\sigma_i} \cdot g/g^{ab} = g^{a(\sigma_i - b)} g = g^{x_i y_i} g \quad \text{and}$$

$$g^{x_j y_j} g^{v_j} \leftarrow \left(g^b\right)^{\sigma_j} \cdot C = \left(g^b\right)^{\sigma_j} \cdot g^{ab} = g^{b(\sigma_j + a)} = g^{x_j y_j} \, ,$$

which means that in our bare bulletin board $v_i = 1$ and $v_j = 0$.

In the case that $C = g^{ab+1}$, we have:

$$g^{x_i y_i} g^{v_i} \leftarrow (g^a)^{\sigma_i} \cdot g/C = (g^a)^{\sigma_i} \cdot g/g^{ab+1} = g^{a(\sigma_i - b)} = g^{x_i y_i} \quad \text{and}$$

$$g^{x_j y_j} g^{v_j} \leftarrow \left(g^b\right)^{\sigma_j} \cdot C = \left(g^b\right)^{\sigma_j} \cdot g^{ab+1} = g^{b(\sigma_j + a)} g = g^{x_j y_j} g \, ,$$

which means that in our bare bulletin board $v_i = 0$ and $v_j = 1$.

$\mathcal{S}$ then gives $\mathcal{A}$ the constructed bare bulletin board as input. If $\mathcal{A}$ is able to distinguish which of the above two cases the given bare bulletin board corresponds to, $\mathcal{S}$ will be able to successfully distinguish the two cases for C and hence break Assumption 2. □

THEOREM A.3 (MAIN THEOREM). *Under the DDH assumption and that the ZKP primitives used in the protocol are secure, the* failsafe *DRE-i bulletin board does not reveal anything about the secrecy of the votes other than the tally of non-adversarial votes to an adversary that determines an arbitrary number of votes.*

PROOF. Similar to the proof of Theorem 3.4, whereas now we rely on Lemmas A.1 and A.2 instead. □

A corollary of the above theorem can be stated as below for a passive adversary that does not determine any votes, but only observes the bulletin board.

COROLLARY A.4 (PRIVACY AGAINST PASSIVE ADVERSARIES). *Under the assumptions that DDH is intractable and the ZKP primitives used in the protocol are secure, the failsafe DRE-i bulletin board does not reveal anything about the secrecy of the votes other than the tally of the votes to a passive adversary.*