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

Feng Hao, Matthew Nicolas Kreeger
Thales E-Security, Cambridge, U.K.
{feng.hao, matthew.kreeger}@thales-esecurity.com

Abstract

The Direct Recording Electronic (DRE) system commonly uses touch-screen technology to directly record votes. It can provide several benefits in large-scale electronic voting, including usability, accessibility and efficiency. Unfortunately, a lack of tallying integrity in many existing products has largely discredited the entire approach along with its merits. To address this problem, we propose a cryptographic protocol called DRE-i, where i stands for integrity. We take a broad interpretation of the DRE: which includes not only touch-screen machines, as deployed at polling stations, but also remote voting systems conducted over the Internet or mobile phones. In all cases, the system records electronic votes directly, although the implementations are different. Our DRE-i protocol provides a drop-in solution to add integrity assurance to any DRE voting system without altering the voter's intuitive voting experience. It preserves election tallying integrity even if the DRE machine is completely corrupted, although in that case, vote secrecy will be compromised. The protocol requires a medium (e.g., an attached printer, email, or SMS) that the DRE machine can write the commitment data to. In addition, it requires a public bulletin board that everyone can read. Whilst past electronic voting protocols generally assume trusted computing or rely on trustees (i.e., tallying authorities), our proposal depends on neither. The protocol is self-tallying – that is, anyone can tally the votes, without involving tallying authorities at all.

1 Introduction

The Direct Recording Electronic (DRE) voting system commonly adopts touch-screen technology to record the voter's choice directly. The system can provide several compelling benefits in terms of usability, accessibility and efficiency [10]. Voters, including the disabled and the elderly, generally consider a touch screen interface easy-to-use [8]. The electronic display can be conveniently customized to various language options. In addition, DREs can effectively limit voters to select only a specified number of candidates, hence preventing both over- and under-voting [10].

The procedural complexity of a DRE is low. Firstly, the voter authenticates himself at the polling station and obtains a token (typically, a PIN slip or smart card) [2]. The voter enters a private booth, presents the token to the DRE machine, and starts the voting process. Figure 1 shows an example of the selection choices on the touch screen. The voter follows two basic steps to cast a vote: 1) select a candidate; 2) confirm or cancel. If the voter opts to "confirm" the intended vote, the vote is recorded. Otherwise, no vote is recorded, and the screen will again prompt the voter to select the desired candidate.

However, DRE voting systems have a number of inherent limitations. Firstly, a voter's privacy is constrained by the necessary human interaction with the DRE machine, which records the votes directly [24]. Secondly, there is the unavoidable threat of a Denial of Service (DoS) attack by the DRE machine [9]. The machine may suffer, or be subjected to: human error, malice or misfortune [1], which, if such occurred during the middle of an election, may result in the loss of all stored electronic data. This would cause considerable interruption to the election and significant trust reduction in the process itself.

A far more serious concern with any DRE system is the lack of transparency in terms of tallying. To the public, a DRE machine is a "black box", which performs internal vote calculation and produces a tally result. This lack of transparency, in the internal workings of vote tabulation, raises wide-spread doubts on the accuracy of the produced tally [2]. Many DRE machines claim "trustworthiness" due to awarded government certification. Whilst the certification may be necessary, it is not sufficient in building up public confidence and trust. Previous studies have shown that certified machines still contain an abundance of software defects and system vulnerabilities [2, 11].

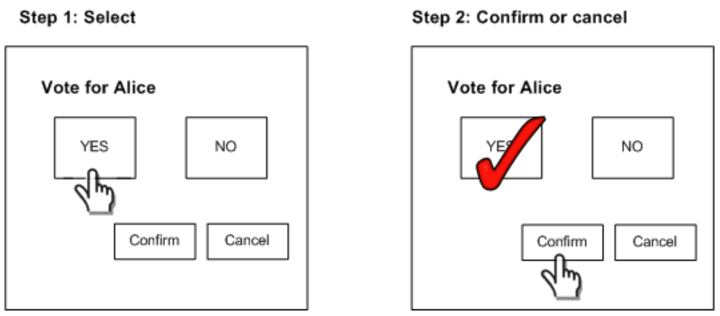

Figure 1: A touch screen based single-candidate DRE voting system

In this paper, we will present a cryptographic voting protocol specifically to address the lack of tallying integrity, which is arguably the most contentious problem with existing DRE-based voting systems. First, we review the related work in the following section.

2 Past Work

There are two categories of e-voting: decentralized and centralized [12]. In the former case, the election is run by voters themselves without involving any trusted third parties. A decentralized e-voting protocol can provide the theoretical best protection of the voter's privacy: the voter does not have to trust anyone but himself [12]. However, such protocols are limited in terms of scalability and are only suitable for small-scale elections [12, 20, 21]. This contrasts with centralized e-voting, where centralized administration ensures a greater level of robustness and is considered more suitable for large-scale elections [8, 25].

DRE is one example of centralized e-voting. As with any centralized system, the DRE machine becomes an attractive target of attack and a single point of failure. If the machine crashes in the middle of election, it may engender great disruptions to the election, effectively causing DoS attacks. On the other hand, due to the centralized nature, it is considerably easier to focus resource on protecting the machine from system failures.

Apart from system robustness, a key issue with a DRE system is whether the software is trustworthy. Government certification of the DRE is one perceived method to instill trust [2], however, such certified systems still contain serious design flaws and software vulnerabilities [2, 11]. It is, therefore, imprudent to rely on certification alone for establishing trust. A more promising approach is to trust-but-verify [29]: where the voting public should have the opportunity to verify whether the DRE software has tallied votes correctly. In other words, the voting system should be "software independent" [33], which is a basic design principle for any e-voting system.

In general, past solutions come in two flavors: use of a Trusted Computing Base (TCB) [3, 4, 10] or tallying authorities [8, 19]. (Paper-based voting protocols [5–7] are not within the scope of this paper.) The TCB approach tries to bootstrap trust from the integrity of a small piece of hardware and software [3]. However, the existence of the TCB is sometimes called into question [25]. For example, Scytl is a commercial TCB-based solution [32]. It is an external device that can be attached to the DRE machine, permitting voters to verify votes in real-time. Essentially, this solution shifts trusting the DRE to trusting Scytl – voters must completely trust the software of the Scytl device (and trust it does not collude with the DRE) [25].

A second approach is to depend on tallying authorities (also called trustees) [8, 27, 28, 35]. For example, in the VoteBox voting system, all the votes are encrypted by a tallying public key using ElGamal encryption [8]. The private key is shared among a number of authorities using a secret sharing scheme. The homomorphic property of the ElGamal encryption facilitates adding votes in the ciphertext form. The final tally is revealed when a quorum of authorities are reached to reconstruct the decryption key. The Helios voting system works in essentially the same way, although is customized for web-based voting [34, 35]. In a different approach, Chaum proposes to let each trustee possess his own public key, which in turn is used to encrypt the votes [27]. The encrypted votes are run through a series of mix-nets for shuffling and re-encryption before they are finally decrypted. If at least one of the mix-nets is honest, voter anonymity is preserved. Neff has a similar mix-net based protocol [28].

All the above protocols [8, 27, 28, 35] assume the trustees do not collude altogether nor lose their secret keys during the course of election. This seems sound when examined in isolation, but may be problematic from a system's context [9]. In theory, the trustees should represent different parties' interests, and are therefore unlikely to collude. However, in practice, they are probably not computer experts [35]. It is possible that all trustees use the software that hails from a single source. In that case, the software may collude (leaking or deleting the private key) without the trustees' knowledge. Therefore, it is crucial to have strict physical and procedural controls in place to securely manage the keys and software used by the trustees. This, as shown in [35], is not a trivial task.

We therefore set out to address the integrity issue without relying on trusted computing or tallying authorities. Our solution is based on an innovative modification of ElGamal encryption. Many cryptographic voting protocols have adopted the conventional form of ElGamal encryption [8, 27, 28, 34, 35]: where data is encrypted against a *static* (or fixed) public key. By contrast, we propose to apply a modified ElGamal scheme that encrypts data against a *dynamic* public key. A dynamic public key is formed by by combining several other public keys in a highly structured way – a technique called "public key juggling" [13]. This technique has proved useful in designing anonymous veto [14, 15], password authenticated key exchange [13], decentralized e-voting [12], and public-key authenticated key exchange [16] protocols. In this paper, we will apply the modified ElGamal encryption to construct a centralized e-voting protocol and demonstrate that it can significantly reduce the system complexity of the existing cryptographic voting protocols [8, 27, 28, 34, 35].

3 The DRE-i Protocol

In this section, we describe a cryptographic e-voting protocol called DRE-i, where i stands for integrity. For simplicity of discussion, we will mainly explain the protocol in the context of touch-screen based voting, but later show its application in remote e-voting.

The design of our protocol differs from other cryptographic e-voting protocols [8, 27, 28, 34, 35] in two main aspects. Firstly, all the random values used in the encryption of votes are chosen before election with commitments published on the public bulletin board. This serves to greatly limit any room of maneuver by a DRE once it is deployed in the field, where the environment becomes much more adverse. Secondly, the protocol is self-tallying. Everyone can tally the votes without relying on any tallying authorities. This has the benefit of minimizing trust in the protocol. In the following sections, we will explain the DRE-i protocol in more detail, and justify our design strategies.

3.1 Integrity requirements

To ensure integrity, a voting protocol should fulfill the following requirements.

1. **Ballot well-formedness:** The ballot must have the correct format to represent exactly one vote.
2. **Recorded as cast:** The recorded vote must be the same as the one the voter intended to cast.
3. **Tallied as recorded:** The tally must be the same as the sum of the recorded votes.

These requirements are intuitive. The first requirement limits a single ballot to have only one vote. For example, in the single-candidate election, the ballot should contribute either 0 or 1 to the tally, nothing more than that. The Zero Knowledge Proof (ZKP) is a well-established technique to ensure ballot well-formedness [18, 19]. The second requirement states the machine must record the correct input from the voter. A widely adopted solution, which is also used in our protocol, is via voter-initiated auditing [29]. The third one is a crucial requirement [28]. Satisfying this requirement in a simple and elegant way is the primary contribution of this paper. Additional requirements such as coercion resistance can be found in [6, 8, 9]. We will explain in Section 4 that our protocol also fulfills those requirements.

3.2 Three Stages of Voting

The DRE-i protocol consists of three stages: ballot generation, ballot casting and ballot tallying. The following sections explain each stage in detail.

3.2.1 Ballot generation

Let G denote a finite cyclic group of prime order q in which the Decision Diffie-Hellman (DDH) problem is intractable [17]. Let g be a generator in G. The parameters (G, g) are publicly agreed before the election starts.

Let us first consider the single-candidate case. The system generates n ballots where n is significantly larger (say 10 times more) than the total number of the eligible voters. The extra ballots are used for auditing purposes.

For each ballot, the system computes a random public key g^{x_i}, where $x_i \in_R [1, q-1]$. When this is 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 g^{y_i} a restructured public key, because it is constructed by multiplying all the random public keys before i and dividing all the public keys after i. Given that x_i is random, $y_i \neq 0$ holds with an exceedingly overwhelming probability. (If $y_i = 0$, it would be publicly obvious that the machine is misbehaving.) In the following theorem, we assume the machine selects x_i properly at random and keep the values secret. In Section 4, we will discuss the implications if the machine deviates from this assumption.

Theorem 1. *Under the Decision Diffie-Hellman assumption, provided $y_i \neq 0$, the term $g^{x_i y_i}$ is indistinguishable from a random non-identity element in the group G.*

Proof. By the protocol definition, $x_i \in_R [1, q-1]$ and $y_i = \sum_{i<j} x_j - \sum_{i>j} x_j$. The y_i is random over Z_q and is unrelated to x_i. Since, $y_i \neq 0$, we have $y_i \in_R [1, q-1]$. To obtain a contradiction, we assume there is a polynomial-time algorithm (an oracle) to distinguish $g^{x_i \cdot y_i}$ from a random non-identity element in the group G. Without loss of generality, we only discuss the case that the oracle distinguishes $g^{x_1 y_1}$ from random.

Given g^a, g^b, g^r, g^{ab} where $a, b, r \in_R [1, q-1]$, the DDH assumption states that g^{ab} is indistinguishable from g^r [17]. We now show how the assumed oracle can break this assumption. Let the DRE machine generate $g^{x_1} = g^a$, $g^{x_2} = g^{-b-\sum_{i>2} x_i}$, and the rest random g^{x_i} for $i > 2$. Note the machine does not know the values of x_1 and x_2 but does for $x_3, x_4, \ldots, x_n$. We can generate $g^{y_i} = \prod_{j<i} g^{x_j} / \prod_{j>i} g^{x_j}$ for every ballot. By definition, $y_1 = -\sum_{i>1} x_i = b$. We use the assumed oracle to efficiently tell $g^{x_1 y_1}$ apart from g^r. Hence, we obtain $g^{x_1 y_1} = g^{ab}$ (which is a given value). For $i = 2$, $g^{x_2 y_2} = g^{(-b-\sum_{i>2} x_i)(a-\sum_{i>2} x_i)}$. Clearly, the machine can easily compute this value, and similarly the value of $g^{x_3 x_3}$ until $g^{x_n y_n}$. Thus, the machine completes simulating the entire electronic ballots. To sum up, by simulating the ballot generation and assuming that $g^{x_1 y_1}$ is distinguishable from random, we have solved the DDH problem by distinguishing g^{ab} from random. This contradicts the assumption that the DDH problem is intractable. The same argument applies if any $g^{x_i y_i}$ $(i \neq 1)$ is distinguishable from random.

$\square$

The "Yes"/"No" value in each ballot is encoded in the form of as $C_i = g^{x_i y_i} \cdot g^{v_i}$ where $v_i = 0$ for "No" and 1 for "Yes". Theorem 1 shows that the no-vote, $g^{x_i y_i}$, is indistinguishable from random. 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, the correlation between the two will make it trivially obvious which is "No" and which is "Yes".

In addition, the system needs to compute a 1-out-of-2 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 the two: 0 and 1. We adopt the standard 1-out-of-n ZKP technique (also known as the CDS technique) presented in [18]. Here, we use $n = 2$.

Given an ElGamal encryption $(x, y) = (g^{x_i}, h^{x_i} m)$, the CDS technique demonstrates that m is either m_0 or m_1 without revealing which[1]. This is achieved by proving the following OR statement:

$$\log_g x = \log_h (y/m_0) \quad \vee \quad \log_g x = \log_h (y/m_1)$$

Figure 2 shows a 3-move interactive protocol using the CDS technique, with $m_0 = g^0$, $m_1 = g^1$ and $h = g^{y_i}$. Applying the Fiat-Shamir's heuristics makes the protocol non-interactive [18], by letting $c = H(i, x, y, a_1, b_1, a_2, b_2)$ where H is a publicly secure hash function. In summary, the 1-out-of-2 ZKP for C_i contains: $(w, a_1, b_1, a_2, b_2, d_{1,} d_{2,})$. Additional information on the 1-out-of-n Knowledge Proof can be found in [18, 19].

As shown in Table 1, we define the cryptograms for the yes/no votes as follows. 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. At the end of the ballot generation, the random public keys are published on the bulletin board, while

[1]In our case, the public key h equals g^{y_i} and is dynamically constructed by combining other public keys. We use the symbol h for simplicity. The context should make the meaning clear.

$$\begin{array}{l|lcl}
\multicolumn{2}{c}{\text{Voter}} & & \text{Verifier} \\
v = 1 & v = 0 & & \\
\hline
w, r_1, d_1 \in_R \mathbb{Z}_q & w, r_2, d_2 \in_R \mathbb{Z}_q & & \\
x \leftarrow g^{x_j} & x \leftarrow g^{x_j} & & \\
y \leftarrow h^{x_j} \cdot g & y \leftarrow h^{x_j} & & \\
a_1 \leftarrow g^{r_1} x^{d_1} & a_1 \leftarrow g^{w} & & \\
b_1 \leftarrow h^{r_1} y^{d_1} & b_1 \leftarrow h^{w} & & \\
a_2 \leftarrow g^{w} & a_2 \leftarrow g^{r_2} x^{d_2} & & \\
b_2 \leftarrow h^{w} & b_2 \leftarrow h^{r_2} (y/g)^{d_2} & \xrightarrow{x, y, a_1, b_1, a_2, b_2} & \\
d_2 \leftarrow c - d_1 & d_1 \leftarrow c - d_2 & \xleftarrow{c} & c \in_R \mathbb{Z}_q \\
r_2 \leftarrow w - x_j d_2 & r_1 \leftarrow w - x_j d_1 & \xrightarrow{d_1, d_2, r_1, r_2} & c \overset{?}{=} d_1 + d_2 \\
 & & & a_1 \overset{?}{=} g^{r_1} x^{d_1} \\
 & & & b_1 \overset{?}{=} h^{r_1} y^{d_1} \\
 & & & a_2 \overset{?}{=} g^{r_2} x^{d_2} \\
 & & & b_2 \overset{?}{=} h^{r_2} (y/g)^{d_2}
\end{array}$$

Figure 2: Proof of Validity: the ballot (x, y) is either $(g^{x_j}, h^{x_j} \cdot g)$ or (g^{x_j}, h^{x_j}) where $h = g^{y_j}$.

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

Table 1: Ballot generation. The table, except the last two columns, is published on a public bulletin board before the election starts.

the cryptograms are kept secret by the machine. At this stage, the x_i secret values become technically redundant and will not be needed for the rest of the protocol execution[2].

3.2.2 Ballot casting

While the ballot generation was performed before the election in a controlled environment, ballot casting occurs at the polling stations on the election day. The environment at the field deployment of the DRE becomes more adverse. However, note that all the random values used in the computation of the cryptograms have been chosen before election and the random public keys have been published on the public bulletin board (see Table 1). The ballot casting basically involves very simple operations to print out the pre-computed cryptograms depending on the voter's choice, as we explain below.

As before, we assume the eligible voter has been properly authenticated before entering the private voting booth and that machine does not know the real identity of the voter. The voter presents the authentication token to the DRE machine and sees the same "select and confirm" interface on the touch screen (Figure 3). The ballot no i may be incremental or randomly assigned – there is no significant difference from the protocol's perspective. To cast the ballot, the voter follows the same two steps.

In step one, the voter selects a choice on the screen. Meanwhile, the machine prints the following commitment data on the paper: the ballot no i, the cryptogram of the selected choice (i.e., $g^{x_i y_i} \cdot g^{v_i}$ where $v_i = 0$ or 1 for "No"/"Yes" choice correspondingly, and a 1-out-of-2 Zero Knowledge Proof to prove that v_i is indeed one of the two values $\{0, 1\}$). The data on the receipt is digitally signed by the machine to prove the authenticity.

In step two, the voter either confirms or cancels the selection. If he chooses to confirm, the system will print a

[2]From the protocol's perspective, the x_i values are no longer needed. However, in practice, there may be reasons to (securely) retain these values in the DRE during the course of election, as they form the most compact backup data. Note that leaking x_i is effectively equivalent to revealing the pre-computed cryptograms – in either case, the leakage will present itself to the public as clear evidence of the machine's misbehavior.

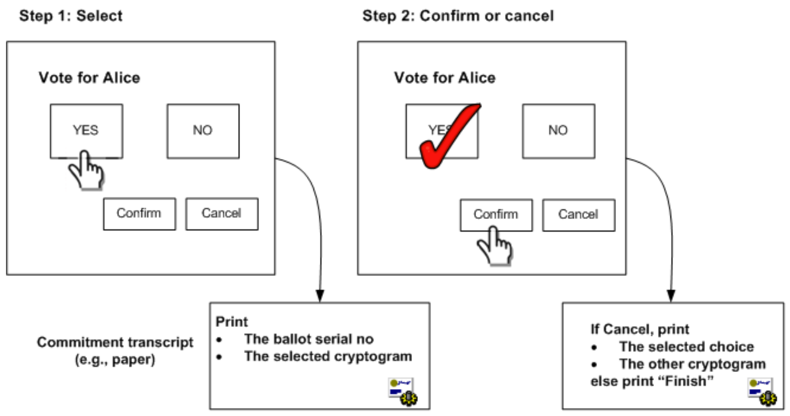

Figure 3: A DRE with integrity (DRE-i) voting system . A confirmed ballot is termed a "valid" vote while a canceled one is referred to as a "dummy" vote.

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

Table 2: Ballot tallying. 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 is to multiply all the V_i values (only including the "No" votes for the dummy case).

"finish" message on the paper. It will also publish the selected cryptogram on a public bulletin board (see Ballot No. 1 and 2 in Table 2 for the example of "No" and "Yes" votes respectively).

However, if the voter chooses to cancel, the DRE machine will print the selected choice, and reveal the other cryptogram onto the paper. It will also publish both cryptograms on the public bulletin board and declare the ballot as "dummy" (see Ballot No. 3 in Table 2). The touch screen will return to the "select candidate" step. A voter is entitled to cast as many dummy votes as he wishes[3], but is allowed to cast only one valid vote.

The cancel option serves for auditing. Note that the 1-out-of-2 ZKP ensures that the formats of "No"/"Yes" votes are in the form of $g^{x_i y_i} \cdot g^{v_i}$, $v_i \in \{0, 1\}$, but it does not guarantee the correct assignment of "0"/"1" to "No"/"Yes". The voter-initiated auditing addresses this. Auditing can be performed by any voter during any stage of the election. When all the voters have cast their votes, the system will reveal the remaining ballots as "dummy" and publish them on the public bulletin board (displayed as if canceled by the voters).

The paper receipt for ballot i contains the printed data from both steps, signed by the machine's private signing key. The digital signature proves the authenticity of the data on the receipt. The voter is free to take home the receipt and verify it against the public bulletin board that his vote has been indeed included. The receipt does not reveal whom the voter has voted for, therefore preventing potential coercion and voter-buying.

3.2.3 Ballot tallying

Tallying the ballots is a case of multiplying the published cryptogram V_i (for dummy votes, only the *no*-value) altogether (See Table 2). 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 $\sum x_i y_i = 0$, which we refer to as the "cancelation formula" (see Proposition 1 and also [12, 14, 15]). The term $\sum_i v_i$ is the total number of the "yes" votes. Since it is a relatively small number, it is feasible to compute it by exhaustive search. However, this exhaustive search is not entirely necessary. Since the machine records the ballots

[3]Obviously, this is bounded by n and, in practice, a reasonable limit would be enforced.

	x_1	x_2	x_3	x_4	x_5
x_1		$-$	$-$	$-$	$-$
x_2	$+$		$-$	$-$	$-$
x_3	$+$	$+$		$-$	$-$
x_4	$+$	$+$	$+$		$-$
x_5	$+$	$+$	$+$	$+$	

Table 3: A simple illustration of $\sum_{i=1}^{n} x_i y_i = 0$ for $n = 5$. The sum $\sum_{i=1}^{n} x_i \left(\sum_{j=1}^{i-1} x_j - \sum_{j=i+1}^{n} x_j\right)$ is the addition of all the cells, where $+, -$ represent the sign. They cancel each other out.

directly, it can announce the count of "yes" votes, β, right after the election. Everyone can verify whether g^{β} and $g^{\sum_i v_i}$ are equal. This takes only one exponentiation. Also, everyone can count the number of dummy votes from the bulletin board, which we denote as λ. Thus, the tally of "no" votes is $\alpha = n - \beta - \lambda$.

Proposition 1. *For the x_i and y_i as defined in the protocol, $\sum_i x_i y_i = 0$.*

Proof. By definition $y_i = \sum_{j<i} x_j - \sum_{j>i} x_j$, hence

$$\begin{aligned}
\sum_i x_i y_i &= \sum_i \sum_{j<i} x_i x_j - \sum_i \sum_{j>i} x_i x_j \\
&= \sum_{j<i} \sum x_i x_j - \sum_{i<j} \sum x_i x_j \\
&= \sum_{j<i} \sum x_i x_j - \sum_{j<i} \sum x_j x_i \\
&= 0.
\end{aligned}$$

Table 3 illustrates this equality in a more intuitive way. □

3.3 Extension to multiple candidates

There are several ways to extend a single-candidate election to multiple candidates [12]. A preferred method is attributed to Cramer *et al.* [19]: suppose that we have n votes, choose m so that m is the smallest integer such that $2^m > n$. Now the vote for candidate 1 is encoded as 2^0, for candidate 2 as 2^m, for candidate 3 is 2^{2m}, and so on. In other words, redefine the encoding value v_i within the cryptogram definition $C_i = g^{x_i y_i} \cdot g^{v_i}$ as:

$$v_i = \begin{cases} 2^0 & \text{if vote for candidate 1} \\ 2^m & \text{if vote for candidate 2} \\ \cdots & \cdots \\ 2^{(k-1)m} & \text{if vote for candidate } k \end{cases}$$

Tabulation is much as before: $\prod_i g^{x_i y_i} g^{v_i} = g^{\sum_i v_i}$. The votes are summed and the super-increasing nature of the encoding ensures that the total can unambiguously be resolved into the totals for the candidates. Hence, $\sum_i v_i = 2^0 \cdot c_1 + 2^m \cdot c_2 + \ldots + 2^{(k-1)m} \cdot c_k$, where c_1 to c_k are the counts of votes for the k candidates correspondingly. As before, the machine will announce the counts of votes right after the election. Anyone can verify the counts against $g^{\sum_i v_i}$, which takes a single exponentiation.

3.4 Remote e-voting

We now take a broader interpretation of the DRE voting system: which not only includes on-site, touch-screen machines, but also remote voting systems via the Internet or mobile phones. In all cases, the system records the votes directly, although the security environments are different.

The DRE-i protocol is generically applicable to both on-site and remote e-voting scenarios. The protocol remains basically the same although the implementations are quite different. For example, in web-based Internet voting, the DRE machine may commit data by sending a signed email (as opposed to printing on paper). Similarly, if a mobile phone is used to vote, a signed Short Message Service (SMS) may be sent.

However, we need to stress that on-site and remote voting applications have distinct voting environments, each with an impact on security. For instance, in a remote setting, we lose some of the procedural and physical protections that are available when conducting an on-site election. This opens up a number of new attacks – for example, a voter may be cajoled in disclosing their vote via a "bogus" website; the actual vote may be conducted under the duress of a coercer; voting credentials may become an item of profitable trade et cetera. Also, any independent observation of "counting valid voters" at the polling station will not be possible. Nevertheless, remote e-voting may still prove useful in some specifically identified scenarios, where the concerns on coercion and voter privacy are low [34, 35].

4 DRE-i Analysis

Electronic voting is a complex problem. It requires more than just technical considerations as it crosses the disciplines of politics, human psychology, security economics and sociology et cetera [1]. Within this paper we only focused on solving a technical problem: tallying integrity. However, it is unlikely that any single technical protocol alone can guarantee a secure election. Correct implementation of the protocol is crucial. Also, there are realistic threats that our protocol alone cannot address.

In the following sections, we perform a comprehensive analysis of the proposed DRE-i protocol: explaining the technical properties of the protocol, discussing protocol implementations, highlighting practical threats concerning deployment as well as suggesting possible mitigation strategies.

4.1 Technical Properties of the Protocol

The DRE-i protocol fulfils the three integrity requirements as described in Section 3.1. The use of the CDS technique (i.e., the 1-out-of-n zero knowledge proof) ensures the correct format of the ballot [18], thus fulfilling the first requirement. The second requirement is satisfied by the auditing function. Any voter can be an auditor by simply pressing the "cancel" button. Our primary contribution within this paper is to fulfil the third requirement in a simple and elegant way. This permits anyone to easily verify the tally, based on the encrypted data displayed on the public bulletin board, without relying on any tallying authority.

In addition, the protocol protects the secrecy of the valid votes. The published value for a valid vote, $g^{x_i y_i} \cdot g^{v_i}$ for $v_i = 0$ or 1, is indistinguishable from random (see Theorem 1) and the associated 1-out-of-2 ZKP reveals nothing more than the statement: the v_i is either 0 or 1 [18]. If the vote is dummy, both cryptograms will be revealed. A dummy vote requires no secrecy since it does not add to the tally. As with any DRE system, the machine naturally learns the value of each vote (i.e., "Yes" or "No" for a single-candidate election). Our technical protocol cannot prevent a corrupted machine leaking the secret values, but there are further physical and procedural means to protect the voter. For example, as we stated earlier, the voting officials shall ensure: 1) the voting booth is private; 2) the DRE machine does not know the voter's real identity; and 3) the published ballots do not show any linkage to the voters. These measures serve to decouple the voter's identity from each cast ballot, and hence to preserve the voter's anonymity.

The paper receipt in our protocol is coercion free. As we detailed earlier, if the voter chooses to confirm the vote, the receipt does not leak any information about whom he had voted for. This prevents potential coercion and vote-buying. If, however, the voter opts to cancel the vote, the receipt will reveal the selected choice, but the vote has been declared dummy. A dummy vote is useless to the coercer.

4.2 Estimating the Computation Cost

We begin by examining ballot generation. This stage involves computationally intensive operations. For a typical scenario, let us assume $n = 10^5$ (which is 10 thousand voters at a polling station times a safety factor of 10 for auditing). Also, we assume a typical cyclic group setting where p is 1024-bit and q is 160-bit.

As shown in Table 1, we need to compute g^{y_i} for each ballot. At first glance, this is very expensive, taking approximately $n = 10^5$ multiplications to compute g^{y_1}. 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 $i = 2, 3, \dots, n$, 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 to compute the 1-out-2 ZKP for each no/yes vote, totalling ten exponentiations. Each exponentiation takes approximately 5 milliseconds on a 2.33-GHz MacBook laptop [13]. Therefore, pre-computing all ballots on a single laptop takes $0.005 \times 10 \times n = 5 \times 10^3$ seconds = 1.4 hours. (We have not factored in the use of any form of optimization technique e.g. caching.)

In the ballot casting stage, the computational cost incurred by the DRE machine is nearly negligible – the machine merely needs to print out the pre-computed cryptogram according to the voter's choice. This is in contrast to other cryptographic voting protocols [8,27,28,34], which require the machine to perform intensive cryptographic operations in the field deployment on the election day[4].

The ballot tallying involves multiplying n group elements to obtain $g^{\sum v_i}$. One exponentiation requires an average of $1.5 \times \log_2 q = 240$ multiplications. The multiplication will take approximately $n/240 \times 0.005 = 2.08$ seconds. Verifying the tally against the count accounted for by the DRE requires one additional exponentiation: that is, another 0.005 seconds.

In addition, before an election, anyone can verify that the published random public keys g^{x_i} lie within the prime-order group, are randomly distributed 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 as shown in Figure 2, taking roughly 0.02 seconds on a laptop.

4.3 Voter Verified Paper Audit Trail

The retaining of a paper record for each electronically cast ballot was first suggested by Mercuri in 2000 as the Voter Verified Paper Audit Trail (VVPAT) [30]. This permits independent tallying in the form of manually counting the paper records. The DRE-i protocol itself does not have this feature.

However, it is possible to add the VVPAT feature to the DRE-i protocol. When the voter is asked to confirm the candidate selection, the DRE machine prints a paper ballot under a layer of glass (permitting visual inspection) before being transferred to a secure location. The voter leaves the booth with a DRE-i receipt. The addition of the VVPAT may inconvenience some users, especially the disabled, decreasing the system usability and accessibility.

Aside from the obvious benefit as a "fall back" mechanism (in case the machine crashes and all electronically stored data is lost), the use of VVPAT may introduce new problems. For example: what if the electronic tally and the manual recount do not match? Which one is more accurate? The data published on the public bulletin board is publicly verifiable and re-computation will return exactly the same result. This can be contrasted with the time-consuming, error-prone process of a manual recount. In addition, different attempts of the manual recount may well return different tallying results [31]. Therefore, the limitation of manual recounting needs to be fully acknowledged and understood before VVPAT is added.

4.4 Subliminal Channels

A subliminal channel is concerned with information leakage about a cast ballot to a third-party. Karlof, Sastry and Wagner describe possible subliminal channel attacks against cryptographic e-voting protocols such as Neff's and Chaum's [9]. The threat of subliminal channels also applies to DRE-i.

During the ballot generation, we assume that the DRE machine generates the x_i values randomly and secretly. However, a corrupted machine may deviate from this assumption. It may choose the x_i values from a low-entropy source, so the value $g^{x_i y_i}$ will no longer be indistinguishable from random. In this case, the public keys g^{x_i} will not be randomly distributed and, therefore, subject to detection. Hence, more likely, the corrupted machine will choose x_i at random, but leak the values to the coercer. This is an inherent threat with any DRE-based voting system due to the machine directly recording the vote and learning the voter's secret choice. After the election, all the x_i values together with the unused cryptograms should be permanently deleted. In any event, if any of these values becomes known to the public, this will present itself as clear evidence about the machine's misbehavior.

[4]One benefit of our design strategy becomes evident in web-based e-voting. For example, Helios [35] requires Java plug-in installed on the voter's browser to perform the needed cryptographic operations. The use of Java is considered one major limitation with Helios according to the IACR e-voting trial report [36]. Clearly, our protocol is free from this issue.

4.5 Denial of Service

The electronic data published by the DRE machine must be precise. If a single bit of the vote gets flipped, it will break the well-formedness of the vote, thus bringing it to the public attention. Also, all votes must be included within the tallying process. If a single vote goes missing, rather than going unnoticed, the tallying process at the particular precinct will fail – essentially, a publicly evident Denial of Service attack. This may appear, at first glance, as a weakness. On the contrary, this is a strength of the protocol, ensuring every vote is counted. This effectively demands that DRE vendors must follow stringent engineering practice to ensure hardware and software robustness (which is generally required in most security-critical systems [1], e.g., payment solutions in the banking industry).

4.6 More Powerful Coercion

We previously highlighted that the DRE-i receipt is coercion free. However, there may be more powerful coercion scenarios: e.g., threatening voters not to vote at all, forcing voters to film everything they do in the private booth, or colluding with the DRE machine to discover the voters' secret votes et cetera [9]. It is beyond our protocol to address these threats. If such powerful coercion becomes wide spread, the value of the election itself may be called into question.

4.7 Voter Enrollment and Authentication

Voter enrollment and authentication are two important pre-conditions for our protocol. The former helps to determine the volume of ballots to generate. The latter is crucial to ensure *one-man-one-vote*. The election staff at the polling station must keep a reliable record of how many voters have voted on the election day. This number can then be compared with the total count of valid votes published on the public bulletin board. It is possible, that once inside the booth, the voter fails to cast a vote, or only casts dummy votes. This would be evident by the paper receipt or the returned authentication token.

4.8 Social engineering attacks

Typically, threats have centered on how to identify a misbehaving DRE machine, however, we also need to consider the case where the DRE is honest, but the voter is misbehaving. We call this a "social engineering attack", as the attack is not technical in nature, but can sometimes be very effective if countermeasures are lacking. We highlight a few attacks below. Note that they are also applicable to other e-voting protocols in general [27, 28, 34, 35].

As an example, suppose the voter selects "yes" in the first step, and then chooses to cancel. The machine dutifully prints a receipt to reveal the "yes" selection and declares the vote as dummy. The voter may now report to an election staff that he actually selected "no". We assume the purpose of this attack is to discredit the machine.

In such a situation, is the voter misbehaving or the machine? For the election staff, it is not easy to tell the difference. If the voter raises the dispute, one practical resolution is to invite several independent observers to supervise the voting. The independent observers do not need to learn the voter's secret. For example, the voter casts several dummy votes under independent observation until being happy that the machine is acting in accordance with his wishes.

In another example, the voter might dispute that he chose "cancel", but the screen displays "your vote has been confirmed". Again, there is no easy way to tell who is lying. One resolution may be to have several voting officials jointly agree to mark the ballot as "disputed" and allow the voter to cast another vote. When the election finishes, the "disputed" ballots, together with the unused ballots, will be revealed by the DRE and declared "dummy". The key to resolution is to ensure that the handling of the "disputed" votes is transparent to the public. In any event, the total number of the valid votes published on the public bulletin board shall match the number of the voters who actually cast their votes on the election day.

4.9 Receipt Verification

The generated paper receipt provides the voter the ability to verify whether his vote has been correctly included in the final tally. The receipt itself does not reveal any information about whom the voter has voted for. Still, it is important for voters to verify the receipts, so that DRE fraud, if any, can be detected.

However, in general, we should assume that many voters will not endeavour to verify the receipts by themselves. This is a general problem for many cryptographic e-voting protocols [27, 28], and not specific to our protocol. It is therefore crucial for the election officials to establish incentive schemes, that encourage verification of receipts, and to provide voters with all the necessary facilities and assistance to do so (say near the exit of the polling station). The successful verification of the receipt will help build up public trust in the election.

5 Conclusion

In this paper, we proposed the DRE-i protocol to tackle arguably the most contentious problem with the DRE voting system: the lack of tallying integrity. Our solution adds cryptographic assurance of tallying integrity to the DRE system, without altering the voter's intuitive voting experience; the auditing is voter-initiated and has been seamlessly integrated into the natural voting process; the protocol is generically applicable to both on-site and remote e-voting; the election is self-tallying, so the public can tally the votes without relying on trusted computing or tallying authorities. The integrity of an election underlies the integrity of democracy. With the DRE-i protocol, "every vote counts" is no longer a mere slogan.

Acknowledgment

This paper is built upon the previous published works authored in collaboration with Dr. Piotr Zieliński and Prof. Peter Ryan. They also made contributions into this paper. We also like to thank Ross Anderson, Joseph Bonneau and other members of the security groups at the Computer Laboratory, University of Cambridge, for many useful comments.

References

[1] R.J. Anderson, *Security Engineering : A Guide to Building Dependable Distributed Systems*, Second Edition, New York, Wiley 2008.

[2] T. Kohno, A. Stubblefield, A.D. Rubin, and D.S. Wallach, "Analysis of an Electronic Voting System," Proceedings of the 25th IEEE Symposium on Security and Privacy, May, 2004.

[3] R. W. Gardner, S. Garera, A.D. Rubin, "Designing for Audit: A Voting Machine with a Tiny TCB," Proceedings of the 14th Financial Cryptography and Date Security, January, 2010.

[4] S. Garera and A.D. Rubin, "An Independent Audit Framework for Software Dependent Voting Systems," Proceedings of the 14th ACM Conference on Computer and Communications Security, October 2007.

[5] R.L. Rivest, W.D. Smith, "Three Voting Protocols: ThreeBallot, VAV, and Twin," Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology, 2007.

[6] D. Chaum, R. Carback, J. Clark, A. Essex, S. Popoveniuc, R. Rivest, P. Ryan, E. Shen, A. Sherman, "Scantegrity II: End-to-End Verifiability for Optical Scan Election Systems Using Invisible Ink Confirmation Codes," Proceedings of the USENIX/ACCURATE Electronic Voting Workshop, 2008.

[7] D. Chaum, P.Y. Ryan, and S.A. Schneider, "A Practical, Voterverifiable, Election Scheme," ESORICS'05, LNCS 3679, pp. 118-139, 2005.

[8] D.R. Sandler, K. Derr, and D.S. Wallach, "VoteBox: A Tamper-Evident, Verifiable Electronic Voting System," Proceedings of the 17th USENIX Security Symposium, July 2008.

[9] C. Karlof, N. Sastry, D. Wagner, "Cryptographic Voting Protocols: A Systems Perspective," Proceedings of the 14th USENIX Security Symposium, pp. 33-50, August, 2005.

[10] R.A. Fink, A.T. Sherman, R. Carback, "TPM Meets DRE: Reducing the Trust Base for Electronic Voting Using Trusted Platform Modules," *IEEE Transactions on Information Forensics and Security*, No. 4. Issue. 4, pp. 628-637, 2009.

[11] A.J. Feldman, J.A. Halderman, E.W. Felten, “Security Analysis of the Diebold AccuVote-TS Voting Machine,” Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology, 2007.

[12] F. Hao, P. Ryan, P. Zieliński, “Anonymous Voting by 2-Round Public Discussion”, *IET Information Security*, in press, 2010.

[13] F. Hao, P. Y. A. Ryan, “Password Authenticated Key Exchange by Juggling,” Proceedings of the 16th International Workshop on Security Protocols, Cambridge, UK, April 2008.

[14] F. Hao, P. Zieliński, “A 2-Round Anonymous Veto Protocol,” Proceedings of the 14th International Workshop on Security Protocols, Cambridge, UK, 2006.

[15] F. Hao, P. Zieliński, “The Power of Anonymous Veto in Public Discussion,” *Sprigner Transactions on Computational Sciences IV*, pp. 41-52, 2009

[16] F. Hao, “On Robust Key Agreement Based on Public Key Authentication,” The 14th International Conference on Financial Cryptography and Data Security’10, to appear in LNCS, Tenerife, Spain, 2010.

[17] D. Stinson, *Cryptography: Theory and Practice*, Third Edition, Chapman & Hall/CRC, 2006.

[18] R. Cramer, I. Damgård, B. Schoenmakers, “Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols,” Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology, LNCS, vol. 839, pp. 174-187, 1994.

[19] R. Cramer, M. Franklin, B. Schoenmakers and Moti Yung, “Multi-Authority Secret-Ballot Elections with Linear Work,” EUROCRYPT ’96, LNCS, vol. 1070, pp. 72-83, 1996.

[20] A. Kiayias, M. Yung, “Self-tallying elections and perfect ballot secrecy,” Public Key Cryptography ’02, LNCS, vol. 2274, pp. 141-158, 2002.

[21] J. Groth, “Efficient maximal privacy in boardroom votisng and anonymous broadcast,” Financial Cryptography ’04, LNCS, vol. 3110, pp. 90-104, 2004.

[22] D. Chaum, “The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability,” *Journal of Cryptology*, vol. 1, no. 1, pp. 65-67, 1988.

[23] D. Chaum, “Untraceable Electronic Email, Return Addresses, and Digital Pseudonyms,” *Communications of the ACM*, Vol. 24, No. 2, pp. 84–88, 1981.

[24] A.M. Keller, D. Mertz, J.L. Hall, A. Urken, “Privacy issues in an electronic voting machine,” Proceedings of the 2004 ACM workshop on Privacy in the electronic society, pp. 33-34, 2004.

[25] A.T. Sherman, A. Gangopadhyay, S.H. Holden, G. Karabatis, A.G. Koru, C.M. Law, D.F. Norris, J. Pinkston, A. Sears, and D. Zhang, “An Examination of Vote Verification Technologies: Findings and Experiences from the Maryland Study,” Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop, 2006.

[26] B. Bederson, B. Lee, R. Sherman, P. Herrnson, “Electronic Voting System Usability Issues, ” Proceedings of CHI in Human Factors in Computing Systems, 2003.

[27] D. Chaum, “Secret-Ballot Receipts: True Voter-Verifiable Elections,” *IEEE Security and Privacy*, vol. 2, no. 1, pp. 38-47, Jan. 2004.

[28] C.A. Neff, “A Verifiable Secret Shuffle and Its Application to E-Voting,”Proceedings of the 8th ACM conference on Computer and Communications Security, pp. 116-125, 2001.

[29] J. Benaloh, “Ballot Casting Assurance via Voter-Initiated Poll Station Auditing,” Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology, 2007.

[30] R. Mercuri, “Electronic Vote Tabulation Checks & Balances,” Ph.D thesis, University of Pennsylvania, 2000.

[31] G.C. Edwards III, “The 2000 U.S. Presidential Election,” Taiwan Journal of Democracy, Vol. 2, N. 1, pp. 37-50, 2006.

[32] The commerical Syctl voting solution website: `http://www.scytl.com`

[33] R.L. Rivest and J.P. Wack, “On the notion of Software Independence in Voting Systems,” 2006. Available at `http://vote.nist.gov/SI-in-voting.pdf`

[34] B. Adida, “Helios: web-based open-audit voting,” Proceedings of the 17th conference on Security symposium, pp. 335-348, 2008.

[35] B. Adida, O. de Marneffe, O. Pereira, and J.J. Quisquater, Proceedings of the Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, 2009.

[36] S. Haber, J. Benaloh, S. Halevi, “The Helios e-Voting Demo for the IACR,” June, 2010. Available at `http://www.iacr.org/elections/eVoting/heliosDemo.pdf`