E2E Verifiable Borda Count Voting System without Tallying Authorities

Samiran Bag
University of Warwick
Coventry, United Kingdom
samiran.bag@warwick.ac.uk

Muhammad Ajmal Azad
University of Derby
Derby, United Kingdom
m.azad@derby.ac.uk

Feng Hao
University of Warwick
Coventry, United Kingdom
feng.hao@warwick.ac.uk

ABSTRACT

An end-to-end verifiable (E2E) voting system enables candidates, voters and observers to monitor the integrity of an election process and verify the results without relying on trusted systems. In this paper, we propose a DRE-based Borda count e-voting system called DRE-Borda. The proposed system is E2E verifiable without involving any tallying authorities. Furthermore, it outputs only the total score a candidate gets without revealing any other information such as the breakdown of scores with respect to different ranks. This reduces the information leakage from the tallying result to the minimum, hence effectively preventing Italian attacks. When the DRE machine is completely compromised, the integrity of the tallying result is still preserved and what an adversary can learn from a compromised machine is strictly limited to the partial tally at the time of compromise.

KEYWORDS

Borda count election, E-voting, Non-interactive zero knowledge proof, Direct-recording electronic voting machine, End-to-end verifiability.

ACM Reference Format:
Samiran Bag, Muhammad Ajmal Azad, and Feng Hao. 2019. E2E Verifiable Borda Count Voting System without Tallying Authorities. In *Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES 2019) (ARES '19), August 26–29, 2019, Canterbury, United Kingdom.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3339252.3339259

1 INTRODUCTION

The Borda count is a ranked-choice electoral system, which has been used in national assembly elections in Nauru and Slovenia, and also in student union elections in several universities (e.g., Michigan, UCLA, Harvard) [10, 15, 23]. In a Borda count system, each voter ranks the candidates in order. A candidate receives a score depending on its position in the ranking order. For example, a candidate may get 1 point for being the last preference, 2 points for the second last preference and so on. The candidate with the highest aggregated points wins. As compared to the more common plurality voting method where voters are only permitted to choose one candidate from a list (e.g., in "first past the post"), the Borda count system allows taking in more information from voters about their preferences of all candidates. Hence, it tends to elect a candidate who is broadly acceptable rather than simply the most popular[1] [23].

In a traditional Borda count voting system, voters mark their preferences of the candidates on a paper ballot. However, manual counting of the Borda count votes can prove tedious and error-prone especially when there are many cast ballots and many candidates. Often, an electronic device is used to automate the process, e.g., using a Direct Recording Electronic (DRE) voting machine to record the voter's choice directly, and compute the scores for all candidates electronically. Thus, tallying results are instantly available once all voters have cast their votes. However, the use of an election machine introduces a number of security threats: e.g., the machine (or the software inside the machine) may be compromised to give selected candidates certain advantages without the public knowing it.

To address the lack of integrity for a DRE-based voting system, Chaum first proposed a seminal solution based on visual cryptography [11]. Chaum's voting protocol allows voters to visually verify their votes by overlaying two printed transparencies, each being an encrypted share of the vote. Obviously, the operation of physically aligning transparencies is not straightforward for most voters. Nonetheless, Chaum's solution highlights the important notion of *end-to-end (E2E) verifibility*, which encompasses the following requirements: individual voters are able to verify their votes are *cast as intended*, and *recorded as cast*, and any public observers (universal verifiers) are able to verify that all votes are *tallied as recorded*. Following Chaum's work [11], many other E2E verifiable voting systems are proposed to improve Chaum's scheme in various ways, e.g., MarkPledge [21], Prêt à Voter [22], Punchscan [14], Scantegrity [13], Scantegrity II [12], scratch & vote [4], STAR-Vote [6], Adder [20], and Helios [5]. These systems use either mix-net [11] or homomorphic encryption [3], but they all involve a set of trustworthy tallying authorities (TAs) to perform the decryption and tallying process in a publicly verifiable way.

One major obstacle in deploying these E2E verifiable e-voting systems in practice is finding and managing such a set of TAs. The TAs should be selected from different parties of conflicting interests, so they are unlikely to collude. A threshold control scheme should be applied to distribute the trust among the TAs. The threshold should be sufficiently high such that collusion is difficult, but also sufficiently low such that the system is fault-tolerant (e.g., a n/n threshold control gives the highest resistance to collusion, but the

ARES '19, August 26–29, 2019, Canterbury, United Kingdom

ACM ISBN 978-1-4503-7164-3/19/08…$15.00
https://doi.org/10.1145/3339252.3339259

[1] As an example, Donald Trump may be considered the most popular by votes in the 2016 US presidential election, but he was not "broadly acceptable".

whole election data will fail to decrypt if one share is missing). Furthermore, the TAs are assumed to manage their private key shares securely and perform cryptographic operations competently. As reported in the real-world case of using Helios to elect the university president in Université catholique de Louvain (UCL), finding and managing TAs in practice proved to be "one particularly difficult issue" [3].

In 2014, Hao et al. proposed to address the above issue by removing the need for TAs while retaining the E2E verifiability [17]. They called the new approach "self-enforcing e-voting" (SEEV) and proposed a concrete protocol called DRE-i. Their protocol removes the TAs by pre-computing the ciphertexts in such a way that multiplying ciphertexts will cancel out random factors and hence allow anyone to verify the tally. Based on DRE-i, the authors implemented an E2E verifiable Internet voting system, which has been used in campus elections including classroom voting and student prize competitions [16].

In 2016, Shahandashti and Hao proposed an alternative design of the SEEV system called DRE-ip [24]. The DRE-ip system provides the same E2E verifiability as DRE-i without requiring TAs, but in contrast to DRE-i that uses pre-computation, it encrypts ballots in real-time. Because of the real-time computation strategy, DRE-ip is more suitable for polling station voting than DRE-i which requires secure storage of pre-computed ballots. The system removes the need for TAs by keeping an aggregated form of the random factors in the memory, which will only be published in the end of the election to enable public verification of the tallying integrity without involving any TAs. The system provides strong privacy guarantee that when the DRE is completely compromised, an attacker will only be able to learn the partial tally at the time of compromise, which is the minimum information leakage.

Self-enforcing e-voting systems such as DRE-i [17] and DRE-ip [24] significantly simplify the election management by removing the TAs. However, existing SEEV systems only support simple non-ranked voting methods. For example, the basic versions of DRE-i and DRE-ip are designed only for a single candidate election with "Yes" and "No" choices, which can be extended to support multiple candidates as explained in [17] and [24]. The extended systems use approval voting where the candidates are still not ranked. In the past work, there are several E2E voting systems that can support ranked-choice voting such as [8, 9, 18, 19, 22, 25], however all of these systems require a set of tallying authorities. It remains an open question if E2E voting systems that support more complex ranked-choice voting methods without requiring any TAs are possible.

In this paper, we address this question and provide a positive answer for one type of the ranked choice voting system, namely, the Borda count. We present a concrete protocol, called DRE-Borda, which is designed for DRE-based on-site voting in a polling station. Our design is inspired by the real-time computation strategy in DRE-ip but we adapt it to support a full ranked-choice voting method based on the Borda count. The DRE-Borda protocol is E2E verifiable without involving any tallying authorities. As compared to traditional paper-based Borda count voting which displays all paper ballots for counting (after physical mixing), DRE-Borda only reveals the total score that each candidate gets, hence effectively preventing Italian attacks. When the DRE machine is completely compromised, our protocol still guarantees the tallying integrity, and strictly limits the attacker to learn only the partial tally at the time of compromise. Our contributions are summarized below.

- We propose the first E2E verifiable Borda count voting system without any TAs. Our system is designed for DRE-based voting in a polling station environment.
- We present security proofs to show the security and privacy guarantee of our proposed system, and detailed analysis of the system complexity to show its practical feasibility.

2 PRELIMINARIES

2.1 Trust assumptions

We explain our trust assumptions according to the following requirements.

- **Registration** We assume voter registration is done securely, and only eligible voters are registered with rights to vote.
- **Authentication** When in the polling station, we assume voters are properly authenticated with identifying documents (e.g., ID, passport). Normally, when voting is finished, the polling station should make available a list of voters who have cast their votes[2]. Any public observer will be able to count the number of voters and compare that against the number of votes published on the public bulletin board (which we will explain). The two numbers should match (for preventing ballot stuffing).
- **Anonymity** We assume voting is anonymous. After being authenticated in the polling station, each voter obtains a random credential, which could be a one-time password or a smartcard. This credential allows the voter to log onto the DRE machine in a private voting booth and cast one vote, but the machine does not know the voter's real identity.
- **Integrity** We assume the DRE machine may alter the voter's choice or change the tallying results. We do not distinguish if such deviating changes are made by accident (say bugs) or by malice (say hacks), but we require any such change be detected even when the machine is completely controlled by an adversary.
- **Vote secrecy** When the voter chooses candidates on the touch-screen, the DRE machine learns the voter's choice by definition. This is unavoidable. We assume the DRE machine keeps the voter's choice secret during a voting session, however, we require that when the DRE is completely compromised by an adversary, what the adversary will learn from the machine is strictly limited to only the partial tally at the time of compromise.

2.2 Borda Count Scheme

Borda count is a special rank-based election scheme. In a Borda count election scheme, a voter ranks the candidates in order of her preference. There is a fixed score associated with every rank. If the total number of candidates in an election is c, there are c different scores denoted as $a_1, a_2, \ldots, a_c$. Here, a_1 is the score associated

[2]In the UK, this list is recorded in a *marked register*, which anyone is entitled to inspect according to the Representation of the People (England and Wales) Regulations 2001. In the US, this information is stored in the each state's *voter files*, which are available in different ways across the different states, such as publishing it online or offering lists against a nominal fee.

with the highest rank and a_c is the score associated with the lowest rank. Obviously, $a_1 > a_2 > \cdots > a_c$ i.e. the higher the rank of a candidate, the higher the candidate's score will be. Upon the completion of the election, the total score obtained by a candidate is calculated as the sum of all scores assigned to this candidate by all the voters according to ranks. The candidate who earns the highest score wins the election. Figure 1 shows an example of a Borda count election. In this example, a candidate obtains 4 points for being the first preference, 3 points for being the second preference, and so on. The candidate B, who obtains the highest score 106, wins the election.

It is worth noting that in a traditional paper-based Borda count system, at the end of the election all votes (after physical mixing) will be displayed for counting. Given c candidates, there are $c!$ different ways to rank the candidates. If a voter is coerced to choose an unusual ranking order of the candidates, the coercer can verify if the voter has followed the instruction by checking the displayed votes in the end of the election. This attack is commonly known as the "Italian attack". In our system, we address this attack by limiting the information revealed from the tallying process: only the total score of each candidate will be shown, not any breakdown of tallies according to ranking choices as shown in the left diagram of Fig. 1.

3 SYSTEM DESIGN

In this section, we present an E2E verifiable DRE-based Borda count voting system without involving any TAs.

3.1 System setup

We present the schematic diagram of the system in Figure 2. The DRE machine has a touch screen through which it allows the voter to rank the candidates in order of her preference. The machine is connected to a local printer, which prints voter receipts. It is also connected to a public bulletin board for publishing all user receipts that enable public verifiability of the tallying integrity. Here, the bulletin board acts as an append-only public data record which everyone is able to read [2, 16].

Let there be n voters in this election. The voters are denoted as $V_i : i \in [1, n]$. There are c candidates who are contesting the election. Let p and q be two large primes satisfying $q \mid (p-1)$. Let G_q be a subgroup of $\mathcal{Z}_p^*$ of the prime order q. The decisional Diffie-Hellman (DDH) problem is assumed to be intractable in G_q. Unless specified otherwise, all modular operations in G_q are performed with respect to the modulus p; hence for the rest of the paper we will omit the explicit "mod p" notation for simplicity. Let g and $\tilde{g}$ be two generators of G_q, such that the logarithmic relationship between them is unknown to anyone (e.g., given the first generator, the second generator can be computed by using a one-way hash function [1]). The DRE machine also holds two $1 \times n$ vectors $S = (s_1, s_2, \ldots, s_c)$, and $U = (u_1, u_2, \ldots, u_c)$, each of which is initialized to zero, that is, initially $S = U = (0, 0, \ldots, 0)$.

3.2 Voting Phase

A voter $i : i \in [1, n]$, upon entering a polling station, needs to first authenticate herself with the polling station officials. Upon successful authentication, the voter obtains a random credential (e.g., a password or smartcard) and enters a private voting booth. The credential allows the voter to log on to the DRE machine and cast a vote. The DRE machine provides an interactive interface for her to rank candidates in order of her preference, as shown in Figure 3.

In the Borda count scheme, the scores given for candidates are defined as $a_1, a_2, \ldots, a_c$ in the decreasing order of preference for c candidates. Here, following the most common scenario, we assign the first preference a score c, the second preference a score $c-1$ and so one. Hence, the voter's selection is essentially a permutation of $(1, 2, \ldots, c)$. Although we take specific values for the Borda count score, our solution works with any values for $a_j, j \in [c]$.

We denote the voter's vote permutation as $V_i = (v_{i1}, v_{i2}, \ldots, v_{ic})$. Obviously, $v_{ij} \in \{a_l : l \in [1, c]\}$, and $\bigcup_{j=1}^{c}\{v_{ij}\} \equiv \{a_l : l \in [1, c]\}$. In other words, $v_{ij} = v_{ij'}$, for any $j, j' \in [1, c]$ implies $j = j'$. Once, the voter $V_i; i \in [n]$ has keyed in her choice, the DRE selects random numbers $X_i = (x_{i1}, x_{i2}, \ldots, x_{ic}) \in_R \mathbb{Z}_q^c$ and computes a ballot $\langle B_i, \tilde{X}_i \rangle$ where $B_i = (b_{i1}, b_{i2}, \ldots, b_{ic}), \tilde{X}_i = (\tilde{x}_{i1}, \tilde{x}_{i2}, \ldots, \tilde{x}_{ic})$, and

$$b_{ij} = g^{x_{ij}} g^{v_{ij}}; \forall j \in [1, c]$$

$$\tilde{x}_{ij} = \tilde{g}^{x_{ij}}; \forall j \in [1, c]$$

The DRE machine prints $\langle B_i, \tilde{X}_i \rangle$ on paper as the first part of the receipt which is also digitally signed to prove the authenticity of data. Thereafter, the DRE machine provides two options to the voter: either to audit the ballot or to confirm it, as shown in Figure 4. If voter i opts for auditing the ballot, the DRE machine prints the second part of the receipt, including randomness $X_i = (x_{i1}, x_{i2}, \ldots, x_{ic})$ and the vote $V_i = (v_{i1}, v_{i2}, \ldots, v_{ic})$, together with a digital signature that cover the entire receipt. The machine marks the ballot as an 'audited' one. The DRE also posts the receipt on the bulletin board and marks it as an audited ballot (see Figure 5). Upon receiving the receipt, the voter needs to check if the revealed vote V_i is the same as her choice; if not, she should raise a dispute to the voting officials immediately. Then the DRE starts over from the beginning letting the voter make a fresh choice once again. Each time the DRE has to choose a fresh set of random factors for generating the ballot. Technically, the voter may choose to audit her ballot as many times as she wishes to. In the end, the voter makes a final choice and proceeds to confirm her ballot. If the voter chooses to confirm the ballot, the DRE generates a NIZK proof of well-formedness of the ballot, and prints it on the receipt along with a digital signature. The NIZK proof associated with a confirmed ballot proves that $(v_{i1}, v_{i2}, \ldots, v_{ic})$ is a permutation of $r = (a_1, a_2, \ldots, a_c)$. In order to prove this, it is sufficient that we prove the following c relations.

$$a_1 \in \{v_{i1}, v_{i2}, \ldots, v_{ic}\}$$

$$a_2 \in \{v_{i1}, v_{i2}, \ldots, v_{ic}\}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots$$

$$a_c \in \{v_{i1}, v_{i2}, \ldots, v_{ic}\}$$

The above c relations prove that $a_i, i \in [1, c]$ is an element of the set $\{v_{i1}, v_{i2}, \ldots, v_{ic}\}$. Given that $\{a_1, a_2, \ldots, a_c\}$ contain distinct values, and it has the same size as the set $\{v_{i1}, v_{i2}, \ldots, v_{ic}\}$, it follows that $\{v_{i1}, v_{i2}, \ldots, v_{ic}\}$ must be a permutation of $\{a_1, a_2, \ldots, a_c\}$. The c relations above are equivalent to the following statement:

$$(b_{i1} = g^{x_{i1}} g^{a_j}) \vee (b_{i2} = g^{x_{i2}} g^{a_j}) \vee \ldots \vee (b_{ic} = g^{x_{ic}} g^{a_j}), \forall j \in [1, c]$$

Number of Voters	14	10	8	4	1
1st choice	A	C	D	B	C
2nd choice	B	B	C	D	D
3rd choice	C	D	B	C	B
4th choice	D	A	A	A	A

$\Rightarrow$

Candidate scores

A: $4 \times 14 + 10 + 8 + 1 = 79$

B: $3 \times 14 + 3 \times 10 + 2 \times 8 + 4 \times 4 + 2 = 106$

C: $2 \times 14 + 4 \times 10 + 3 \times 8 + 2 \times 4 + 4 = 104$

D:$14 + 2 \times 10 + 4 \times 8 + 3 \times 4 + 3 = 81$

Figure 1: An example of a Borda count election

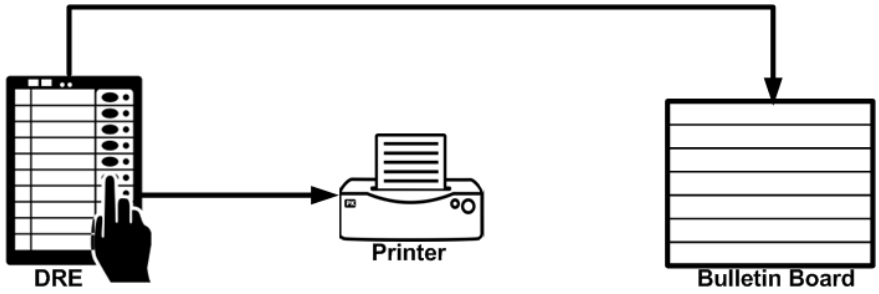

Figure 2: A schematic diagram of the DRE system that includes a printer and the public bulletin board.

We denote the NIZK proof as the following:

$$\Pi_{ij}\left[x_{i1}, x_{i2}, \ldots, x_{ic} : \tilde{X}_i, B_i\right]; \forall j \in [1, c]$$

Here, $\tilde{X}_i = (\tilde{x}_{i1}, \tilde{x}_{i2}, \ldots, \tilde{x}_{ic})$, for all $j \in [1, c]$ the NIZK proof Π_{ij} proves that the following statement is true
$\sigma \equiv (b_{i1} = g^{x_{i1}} g^{a_j} \wedge \tilde{x}_{i1} = \tilde{g}^{x_{i1}}) \vee (b_{i2} = g^{x_{i2}} g^{a_j} \wedge \tilde{x}_{i2} = \tilde{g}^{x_{i2}}) \vee \ldots \vee (b_{ic} = g^{x_{ic}} g^{a_j} \wedge \tilde{x}_{ic} = \tilde{g}^{x_{ic}})$.

The construction of this NIZK proof is detailed in the Appendix. The receipt for the cast ballot is posted on the bulletin board (see Figure 5). The ballot is marked as a confirmed ballot. The DRE machine updates S and U as below $s_k = s_k + x_{ik}, u_k = v = u_k + v_{ik}$ for $k \in [1, c]$, and $S = (s_1, s_2, \ldots, s_c), U = (u_1, u_2, \ldots, u_c)$. After the update, it securely deletes x_{ik} and v_{ik}, $k \in [1, c]$ and ends the voting session for the voter i.

Upon exiting the private voting booth, the voter holds a receipt for a confirmed ballot, and additionally, zero or more receipts for audited ballots. The voter compares the receipt(s) against the bulletin board and verifies that the same content on the receipt(s) is published on the bulletin board (see Fig. 5).

3.3 Tally Phase

Once, the election is over, the DRE machine publishes S and U on the bulletin board. The result is correct if the following equations hold for all $j \in [1, c]$.

$$\prod_{i=1}^{n} \tilde{x}_{ij} = \tilde{g}^{s_j} \tag{1}$$

$$\prod_{i=1}^{n} b_{ij} = g^{s_j} g^{u_j} \tag{2}$$

Note that $U = (u_1, u_2, \ldots, u_c)$ represent the total votes obtained by the candidates where u_j is the total votes obtained by candidate j for all $j \in [1, c]$. The winner of of the election is candidate j, such that $u_j = \max(u_1, u_2, \ldots, u_c)$.

4 CORRECTNESS OF THE SCHEME

We can easily prove that the above scheme is correct. We assume that the actual tally is $U = (u_1, u_2, \ldots, u_c)$ and the actual sum of all randomnesses is $S = (s_1, s_2, \ldots, s_c)$. Hence,

$$\prod_{i=1}^{n} \tilde{x}_{ij} = \tilde{g}^{s_j}; \forall j \in [1, c] \tag{3}$$

and

$$\prod_{i=1}^{n} b_{ij} = g^{s_j} g^{u_j}; \forall j \in [1, c] \tag{4}$$

Let us also assume that after the completion of the election the DRE machine posts a tally vector $U' = (u'_1, u'_2, \ldots, u'_c)$, and the sum of all randomnesses given by $S = (s'_1, s'_2, \ldots, s'_c)$. If these two vectors are to satisfy the verification equations, we must have

$$\prod_{i=1}^{n} \tilde{x}_{ij} = \tilde{g}^{s'_j}; \forall j \in [1, c]$$

and

$$\prod_{i=1}^{n} b_{ij} = g^{s'_j} g^{u'_j}; \forall j \in [1, c]$$

Thus, $\tilde{g}^{s_j} = \tilde{g}^{s'_j}$ and $g^{s_j} g^{u_j} = g^{s'_j} g^{u'_j}$ for all $j \in [1, c]$. From this we get $s_j = s'_j$ and $u_j = u'_j$ for all $j \in [1, c]$. That is, the DRE cannot alter the tally, hence, the scheme is correct.

5 E2E VERIFIABILITY

Ballot is cast as intended: As discussed before, the scheme relies on voter initiated auditing [7, 24] to provide assurance of this property. By the time the DRE displays the screen containing both the options 'Audit' and 'Confirm', the DRE needs to have printed the ballot on paper. Now, if the voter chooses to audit her ballot, the DRE will have to reveal the randomness which would enable decryption of the ballot. If the DRE generates a wrong ballot that does not correspond to the right choice that the voter has

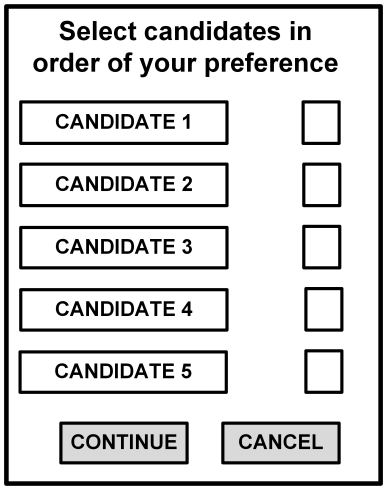

Figure 3: Screenshot of a DRE machine after the voter has started voting. The DRE requires the voter to choose candidates in order of her preference.

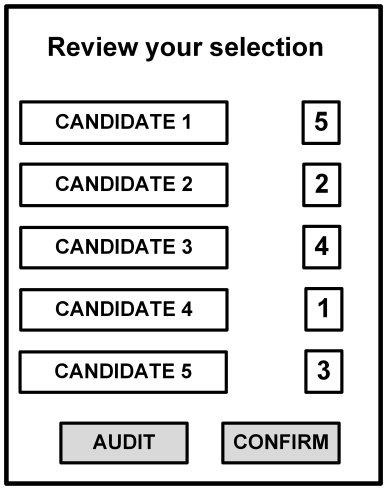

Figure 4: Screenshot of a DRE machine after a voter has ranked all candidates. At this stage the voter should either audit or confirm the ballot.

Ballot Type: Audited
Ballot Id
$(b_{i1}, \tilde{x}_{i1}), (b_{i2}, \tilde{x}_{i2}), \ldots, (b_{ic}, \tilde{x}_{ic})$
Randomnesses: $x_{i1}, x_{i2}, \ldots, x_{ic}$ Votes: $v_{i1}, v_{i2}, \ldots, v_{ic}$
Ballot Type: Confirmed
Ballot Id
$(b_{i1}, \tilde{x}_{i1}), (b_{i2}, \tilde{x}_{i2}), \ldots, (b_{ic}, \tilde{x}_{ic})$
NIZK Proofs: $\Pi_{i1}, \Pi_{i2}, \ldots, \Pi_{ic}$

Figure 5: Structure of Audited and Confirmed ballots posted on the bulletin board with digital signatures omitted.

made, it will get caught if the voter chooses to audit the ballot in the following step. The voter gets to audit as many ballots as she wishes to. Since, the DRE cannot predict beforehand whether the voter is going to audit or confirm a ballot, it cannot act maliciously to change votes at any significant scale without being detected.

Ballot is recorded as cast: Every voter can match the confirmed ballot she was given, printed on her receipt, with the one posted on the bulletin board. The bulletin board which acts as the storage of voter receipts is publicly accessible, making this verification possible.

Ballot is tallied as recorded: Anyone can access the public bulletin board and verify the digital signatures, NIZK proofs and the verification equations about the tally as described in Section 3.3.

6 SECURITY ANALYSIS

In this section, we show that our scheme is secure against all probabilistic polynomial time adversaries that try to deduce the vote of a particular voter. We show that the public bulletin does not reveal any additional information regarding how a voter voted except for what the tally normally reveals. If the attacker colludes with some l number of voters, then she will learn the partial tally of the rest of the $n-l$ voters, but nothing beyond that. Note than anyone who knows how the colluding voters voted can find this partial tally by subtracting the overall tally by the partial tally of the colluding voters. Also in Lemma 6.6, we prove that if an adversary can make momentary access to the DRE machine at a certain time t, she will get to learn the partial tally of all votes cast from the start of the election to time t and from t till the end of the election. We show that the security properties proven in this paper depend on the intractability of DDH problem in the group G. If the Decisional Diffie Hellman problem is hard in the group G, the scheme is secure.

In this proof, we consider an abridged bulletin board where the zero knowledge proofs are simulated. If the zero knowledge proof systems are secure, the adversary should have negligible advantage while dealing with them instead of the real zero knowledge proofs. For the rest of this section, we will not be mentioning the existence of simulated NIZK proofs, but the reader should keep in mind that they exist and are provided by the challenger in each case.

Assumption 1. *Given $g, g^a, g^b \in G$, and a challenge $\Omega \in \{g^{ab}, R\}$, it is hard to decide whether $\Omega = g^{ab}$ or $\Omega = R$.*

Assumption 2. *Given $g, g^a, g^{b_1}, g^{b_2}, \ldots, g^{b_k} \in G$, and a challenge $\Omega \in \{\Omega_1, \Omega_2\}$, where $\Omega_1 = (g^{ab_1}, g^{ab_2}, \ldots, g^{ab_k})$ and $\Omega_2 = (R_1, R_1, \ldots, R_k)$, it is hard to decide whether $\Omega = \Omega_1$ or $\Omega = \Omega_2$.*

Lemma 6.1. *Assumption 1 implies assumption 2. That is assumption 2 will hold true in the group G if assumption 1 holds true in G.*

Proof. We show that if there exists a PPT adversary $\mathcal{A}$ against assumption 2, then we could use the same to construct another adversary $\mathcal{B}$ against assumption 1. $\mathcal{B}$ works as follows. It receives as inputs $g, A = g^a, B = g^b$ and the challenge $\Omega \in \{g^{ab}, R\}$. It selects random $r_1, r_2, \ldots, r_k \in_R \mathbb{Z}_q$, and $s_1, s_2, \ldots, s_k \in_R \mathbb{Z}_q$, and computes $B_i = g^{b_i} = B^{r_i} * g^{s_i}$, for all $i \in [1, k]$. Since, $s_i \in_R \mathbb{Z}_p$,

we have $B_i \in_R G$, for all $i \in [1,k]$. Again, let $\omega_i = \Omega^{r_i} * A^{s_i}$, for all $i \in [1,k]$. If $\Omega = g^{ab}$, then $\omega_i = DH(A, B_i)$, for all $i \in [1,k]$. Alternatively if $\Omega \in_R G$, then since, q is a prime number, $\Omega^{r_i} \in_R G$ for all $i \in [1,k]$. Thus, if $\Omega \in_R G$, then $\omega_i \in_R G$, for all $i \in [1,k]$. Now, $\mathcal{B}$ sends $A, (B_1, B_2, \ldots, B_k)$, and $(\omega_1, \omega_2, \ldots, \omega_k)$ to $\mathcal{A}$. Note that if $\Omega = g^{ab}$, then $(\omega_1, \omega_2, \ldots, \omega_k) = (g^{ab_1}, g^{ab_2}, \ldots, g^{ab_k})$, else if $\Omega = R$, then $(\omega_1, \omega_2, \ldots, \omega_k) = (R_1, R_2, \ldots, R_k)$. If $\mathcal{A}$ can distinguish between these two values, $\mathcal{B}$ can identify the correct value of Ω. It is easy to see that $Adv(\mathcal{A}) \leq Adv(\mathcal{B})$. □

Lemma 6.2. *Given $g, \tilde{g} \in G$, $\mathcal{G} = \{\tilde{g}^{x_i} : i \in [\eta]\}, s = \sum_{i=1}^{\eta} x_i$, A and B are indistinguishable, if $\sum_{i=1}^{\eta} v_i = \sum_{i=1}^{\eta} v'_i$, and $v_i, v'_i \in \mathbb{Z}_q$.*

$g^{x_1} g^{v_1}$	$g^{x_1} g^{v'_1}$
$g^{x_2} g^{v_2}$	$g^{x_2} g^{v'_2}$
...	...
$g^{x_\eta} g^{v_\eta}$	$g^{x_\eta} g^{v'_\eta}$
A	B

Proof. Let us assume $\sum_{i=1}^{\eta} v_i = \sum_{i=1}^{\eta} v'_i = v$. Since, $s = \sum_{i=1}^{\eta} x_i$, we may write $g^{x_\eta} = \frac{g^s}{g^{\sum_{i=1}^{\eta-1} x_i}}$. Note that, the discrete logarithm of $\tilde{g}$ with respect to g is unknown. We assume that $a = \log_{\tilde{g}} g$. So, $\tilde{g}^a = g$. Hence, we can rewrite A and B as follows: $A = (\tilde{g}^{ax_1} g^{v_1}, \tilde{g}^{ax_2} g^{v_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v_{\eta-1}}, \frac{g^s g^{v_\eta}}{\tilde{g}^{\sum_{i=1}^{\eta-1} ax_i}}) = (\tilde{g}^{ax_1} g^{v_1}, \tilde{g}^{ax_2} g^{v_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v_{\eta-1}}, \frac{g^s g^v}{\prod_{i=1}^{\eta-1} \tilde{g}^{ax_i} g^{v_i}})$. Similarly, $B = (\tilde{g}^{ax_1} g^{v'_1}, \tilde{g}^{ax_2} g^{v'_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v'_{\eta-1}}, \frac{g^s g^v}{\prod_{i=1}^{\eta-1} \tilde{g}^{ax_i} g^{v'_i}})$.

Now, in order to prove the above lemma, it is sufficient to show that $(\tilde{g}^{ax_1} g^{v_1}, \tilde{g}^{ax_2} g^{v_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v_{\eta-1}}) \overset{c}{\approx} (\tilde{g}^{ax_1} g^{v'_1}, \tilde{g}^{ax_2} g^{v'_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v'_{\eta-1}})$ First, we show that, $(\tilde{g}^{ax_1} g^{v_1}, \tilde{g}^{ax_2} g^{v_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v_{\eta-1}}) \overset{c}{\approx} (R_1, R_2, \ldots, R_{\eta-1})$. We show that if there exists an adversary $\mathcal{B}$ that can distinguish between these two items, then we can construct another adversary $\mathcal{A}$ against assumption 2. $\mathcal{A}$ receives as input the following items: $\tilde{g}, g = \tilde{g}^a, (\tilde{g}^{x_1}, \tilde{g}^{x_2}, \ldots, \tilde{g}^{x_{\eta-1}})$, and the challenge $(\omega_1, \omega_2, \ldots, \omega_{\eta-1})$, where either $\omega_i = \tilde{g}^{ax_i}$, for all $i \in [1, \eta-1]$ or $\omega_i = R_i \in_R G$, for all $i \in [1, \eta-1]$. $\mathcal{A}$ computes $\omega'_i = \omega_i * g^{v_i}$, for all $i \in [1, \eta-1]$. Now, $\mathcal{A}$ invokes $\mathcal{B}$ with the following inputs: $\tilde{g}, \tilde{g}^a, (\tilde{g}^{x_1}, \tilde{g}^{x_2}, \ldots, \tilde{g}^{x_{\eta-1}}), (v_1, v_2, \ldots, v_{\eta-1})$, and $(\omega'_1, \omega'_2, \ldots, \omega'_{\eta-1})$. Note that, if $\omega_i \in_R G$, then $\omega'_i \in_R G$. else if $\omega_i = \tilde{g}^{ax_i}$, then $\omega'_i = \tilde{g}^{ax_i} g^{v_i}$. If $\mathcal{B}$ can distinguish between them, then $\mathcal{A}$ can identify the challenge correctly. Thus, $(\tilde{g}^{ax_1} g^{v_1}, \tilde{g}^{ax_2} g^{v_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v_{\eta-1}}) \overset{c}{\approx} (R_1, R_2, \ldots, R_{\eta-1})$. Similarly, $(\tilde{g}^{ax_1} g^{v'_1}, \tilde{g}^{ax_2} g^{v'_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v'_{\eta-1}}) \overset{c}{\approx} (R_1, R_2, \ldots, R_{\eta-1})$ also holds. Since, computational indistinguishability is a transitive relation, we may write, $(\tilde{g}^{ax_1} g^{v_1}, \tilde{g}^{ax_2} g^{v_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v_{\eta-1}}) \overset{c}{\approx} (\tilde{g}^{ax_1} g^{v'_1}, \tilde{g}^{ax_2} g^{v'_2}, \ldots, \tilde{g}^{ax_{\eta-1}} g^{v'_{\eta-1}})$. Hence, the result holds. □

Lemma 6.3. *Let us assume that $\mathbf{A} = \{a_1, a_2, \ldots, a_c\}$, and $\kappa = \sum_{i=1}^{c} a_i$. Given $g, \tilde{g} \in G$ and $\tilde{X}_i = (\tilde{g}^{x_{i1}}, \tilde{g}^{x_{i2}}, \ldots, \tilde{g}^{x_{ic}}), \forall i \in [\eta]$, the two bulletin boards viz. A and B are indistinguishable where,*

1. $v_{ij}, v'_{ij} \in \mathbf{A}$
2. $\bigcup_{j=1}^{c} v_{ij} = \bigcup_{j=1}^{c} v'_{ij} = \mathbf{A}$.
3. $\sum_{j=1}^{c} v_{ij} = \sum_{j=1}^{c} v'_{ij} = \kappa, \forall i \in [\eta]$
4. $\sum_{i=1}^{\eta} v_{ij} = \sum_{i=1}^{\eta} v'_{ij}, \forall j \in [c]$
5. $s_j = \sum_{i=1}^{\eta} x_{ij}, \forall j \in [c]$

Proof. Follows from Lemma 6.2. □

Lemma 6.4. *Let us assume that $\mathbf{A} = \{a_1, a_2, \ldots, a_c\}$, and $\kappa = \sum_{i=1}^{c} a_i$. Given $g, \tilde{g} \in G$ and $\tilde{X}_i = (\tilde{g}^{x_{i1}}, \tilde{g}^{x_{i2}}, \ldots, \tilde{g}^{x_{i\kappa}}), \forall i \in [\eta]$, the two bulletin boards A and B are indistinguishable where,*

1. $v_{ij}, v'_{ij} \in \mathbf{A}, \forall i \in [n-\eta], j \in [c]$
2. $\bigcup_{j=1}^{c} v_{ij} = \bigcup_{j=1}^{c} v'_{ij} = \mathbf{A}$
3. $\sum_{j=1}^{c} v_{ij} = \sum_{j=1}^{c} v'_{ij} = \kappa, \forall i \in [n-\eta]$
4. $\sum_{i=1}^{n-\eta} v_{ij} = \sum_{i=1}^{n-\eta} v'_{ij} = u_j, \forall j \in [c]$
5. $s_j = \sum_{i=1}^{\eta} x_{ij}, \forall j \in [c]$

and A and B are as in the table 2

Proof. Select votes $v_{ij} \in \mathbf{A}, \forall i \in \{n-\eta+1, \ldots, n\}, j \in [c]$ satisfying $\bigcup_{j=1}^{c} \{v_{ij}\} = \mathbf{A}, \forall i \in \{n-\eta+1, \ldots, n\}$. Set $v'_{ij} = v_{ij}, \forall i \in \{n-\eta+1, \ldots, n\}, j \in [c]$. Now, $\sum_{i=1}^{n} v_{ij} = \sum_{i=1}^{n} v'_{ij} = u_j + \sum_{i=n-\eta+1}^{n} v_{ij}, \forall j \in [c]$. If an attacker can distinguish between the two bulletin boards, she will disprove Lemma 6.3. □

The following lemma is the main security result of our paper. It shows that in our scheme an adversary cannot learn anything about the secret votes of uncompromised voters if the tally does not allow her to learn it. Consider a case where the adversary colludes with a number of voters. The adversary will know the secret votes of the colluders, she will also know the tally of all the votes as it is made public by the end of the election process. Thus, she can compute the partial tally of the colluding voters which she can then subtract from the overall tally to obtain the partial tally of all uncompromised voters. Now, Lemma 6.5 proves that if there are multiple possible voting patterns corresponding to the same partial tally, the adversary will not be able to distinguish between them. In other words, if multiple sets of votes result in the same partial tally, the adversary will not be able to differentiate between them. That is, the votes of a particular voter will remain secret as long as it cannot be inferred from the final tally. The transcripts of the protocol available with the bulletin board (other than the overall tally) does not provide any additional advantage to the adversary towards learning a particular voter's secret vote.

Lemma 6.5. *In our proposed Borda count scheme, an attacker $\mathcal{A}$ who colludes with $\eta < n$ voters shall only learn the partial tally of $n-\eta$ uncompromised voters, not the individual votes of uncompromised voters.*

Proof. Without loss of generality, we can assume that the indices of the honest voters are $\{1, 2, \ldots, n-\eta\}$ and that of the colluding voters are $\{n-\eta+1, n-\eta+2, \ldots, n\}$. The attacker will learn the votes of the colluding voters but the randomness used to compute the ballots will remain secret. Each ballot is $\langle B_i, \tilde{X}_i \rangle$,

$g^{x_{11}}g^{v_{11}}$	$g^{x_{12}}g^{v_{12}}$	...	$g^{x_{1c}}g^{v_{1c}}$
$g^{x_{21}}g^{v_{21}}$	$g^{x_{22}}g^{v_{22}}$	...	$g^{x_{2c}}g^{v_{2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{\eta 1}}g^{v_{\eta 1}}$	$g^{x_{\eta 2}}g^{v_{\eta 2}}$	...	$g^{x_{\eta c}}g^{v_{\eta c}}$

$g^{x_{11}}g^{v'_{11}}$	$g^{x_{12}}g^{v'_{12}}$	...	$g^{x_{1c}}g^{v'_{1c}}$
$g^{x_{21}}g^{v'_{21}}$	$g^{x_{22}}g^{v'_{22}}$	...	$g^{x_{2c}}g^{v'_{2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{\eta 1}}g^{v'_{\eta 1}}$	$g^{x_{\eta 2}}g^{v'_{\eta 2}}$	...	$g^{x_{\eta c}}g^{v'_{\eta c}}$

Table 1: Bulletin Board A (on the left) & B (on the right)

$g^{x_{11}}g^{v_{11}}$	$g^{x_{12}}g^{v_{12}}$	...	$g^{x_{1c}}g^{v_{1c}}$
$g^{x_{21}}g^{v_{21}}$	$g^{x_{22}}g^{v_{22}}$	...	$g^{x_{2c}}g^{v_{2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n-\eta 1}}g^{v_{n-\eta 1}}$	$g^{x_{n-\eta 2}}g^{v_{n-\eta 2}}$	...	$g^{x_{n-\eta c}}g^{v_{n-\eta c}}$
$g^{x_{n-\eta+11}}$	$g^{x_{n-\eta+12}}$	...	$g^{x_{n-\eta+1c}}$
$g^{x_{n-\eta+21}}$	$g^{x_{n-\eta+22}}$	...	$g^{x_{n-\eta+2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n1}}$	$g^{x_{n2}}$	...	$g^{x_{nc}}$

$g^{x_{11}}g^{v'_{11}}$	$g^{x_{12}}g^{v'_{12}}$	...	$g^{x_{1c}}g^{v'_{1c}}$
$g^{x_{21}}g^{v'_{21}}$	$g^{x_{22}}g^{v'_{22}}$	...	$g^{x_{2c}}g^{v'_{2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n-\eta 1}}g^{v'_{n-\eta 1}}$	$g^{x_{n-\eta 2}}g^{v'_{n-\eta 2}}$	...	$g^{x_{n-\eta c}}g^{v'_{n-\eta c}}$
$g^{x_{n-\eta+11}}$	$g^{x_{n-\eta+12}}$	...	$g^{x_{n-\eta+1c}}$
$g^{x_{n-\eta+21}}$	$g^{x_{n-\eta+22}}$	...	$g^{x_{n-\eta+2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n1}}$	$g^{x_{n2}}$	...	$g^{x_{nc}}$

Table 2: Bulletin Board A(on the left) & B (on the right)

where B_i is of the form $(g^{x_{i1}}g^{v_{i1}}, g^{x_{i2}}g^{v_{i2}}, \ldots, g^{x_{ic}}g^{v_{ic}})$, for all $i \in \{n-\eta+1, \ldots, n\}$. Since, $v_{i1}, v_{i2}, \ldots, v_{ic}$ are known, the attacker can compute $(g^{x_{i1}}, g^{x_{i2}}, \ldots, g^{x_{ic}})$ by dividing each B_{ij} by $g^{v_{ij}}$. Hence, the attacker's view of the bulletin board will be same as what is depicted on Table 3.

$g^{x_{11}}g^{v_{11}}$	$g^{x_{12}}g^{v_{12}}$	...	$g^{x_{1c}}g^{v_{1c}}$
$g^{x_{21}}g^{v_{21}}$	$g^{x_{22}}g^{v_{22}}$	...	$g^{x_{2c}}g^{v_{2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n-\eta 1}}g^{v_{n-\eta 1}}$	$g^{x_{n-\eta 2}}g^{v_{n-\eta 2}}$	...	$g^{x_{n-\eta c}}g^{v_{n-\eta c}}$
$g^{x_{n-\eta+11}}$	$g^{x_{n-\eta+12}}$	...	$g^{x_{n-\eta+1c}}$
$g^{x_{n-\eta+21}}$	$g^{x_{n-\eta+22}}$	...	$g^{x_{n-\eta+2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n1}}$	$g^{x_{n2}}$	...	$g^{x_{nc}}$

Table 3: Attacker's view of bulletin board in Lemma 6.5

Now, according to Lemma 6.4, the two bulletin boards A and B of Table 4 will be indistinguishable. The two bulletin boards of Table 4 have identical partial tally of the honest voters. Hence, if they are indistinguishable to the attacker, the lemma holds.

□

Lemma 6.6. *Let, the DRE starts the election process at time $t = t_0$ and finishes at time $t = t_e$. If an attacker gains temporary access of the DRE at time $t = t_1 \in [t_0, t_e]$, she will only learn the partial tally of all the confirmed votes cast between t_0 and t_1 and between t_1 and t_e, but not the individual votes.*

Proof. The DRE machine only stores two variables, the sum of all randomnesses S used to compute the ballots and the tally t that includes all confirmed and audited ballots cast so far. Since, both S and t are initialized to 0 at the beginning, If the attacker has temporary access to the DRE at time t_1, it will learn the partial tally and the partial sum of randomnesses used to compute the ballots between time t_0 and t_1 and time t_1 and t_e. Through a similar approach as was taken in Lemma 6.5, we can show that the attacker will not learn the individual votes cast between $t = t_0$ and $t = t_1$ or between t_1 and t_e. □

7 PERFORMANCE ANALYSIS

In this section, we analyze the computation and the communication cost incurred on a DRE machine. Since, exponentiation is the most expensive operation in our scheme, we measure the computation cost in terms of the total number of exponentiations performed by the DRE. There are two types of ballots, namely audited ballot and confirmed ballot. Recall that a voter can cast only a single confirmed ballot. However, there is no restrictions on the number of ballots a voter can audit.

$g^{x_{11}}g^{v_{11}}$	$g^{x_{12}}g^{v_{12}}$	...	$g^{x_{1c}}g^{v_{1c}}$
$g^{x_{21}}g^{v_{21}}$	$g^{x_{22}}g^{v_{22}}$	...	$g^{x_{2c}}g^{v_{2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n-\eta 1}}g^{v_{n-\eta 1}}$	$g^{x_{n-\eta 2}}g^{v_{n-\eta 2}}$	...	$g^{x_{n-\eta c}}g^{v_{n-\eta c}}$
$g^{x_{n-\eta+11}}$	$g^{x_{n-\eta+12}}$	...	$g^{x_{n-\eta+1c}}$
$g^{x_{n-\eta+21}}$	$g^{x_{n-\eta+22}}$	...	$g^{x_{n-\eta+2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n1}}$	$g^{x_{n2}}$	...	$g^{x_{nc}}$

$g^{x_{11}}g^{v'_{11}}$	$g^{x_{12}}g^{v'_{12}}$	...	$g^{x_{1c}}g^{v'_{1c}}$
$g^{x_{21}}g^{v'_{21}}$	$g^{x_{22}}g^{v'_{22}}$	...	$g^{x_{2c}}g^{v'_{2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n-\eta 1}}g^{v'_{n-\eta 1}}$	$g^{x_{n-\eta 2}}g^{v'_{n-\eta 2}}$	...	$g^{x_{n-\eta c}}g^{v'_{n-\eta c}}$
$g^{x_{n-\eta+11}}$	$g^{x_{n-\eta+12}}$	...	$g^{x_{n-\eta+1c}}$
$g^{x_{n-\eta+21}}$	$g^{x_{n-\eta+22}}$	...	$g^{x_{n-\eta+2c}}$
...	...	...	...
...	...	...	...
...	...	...	...
$g^{x_{n1}}$	$g^{x_{n2}}$	...	$g^{x_{nc}}$

Table 4: Bulletin Board A(on the left) & B (on the right)

	Computation Cost				Communication Cost			
Type	**Ballot**	**Secret Key**	**NIZKP**	**Total**	**Ballot**	**Secret Key**	**NIZKP**	**Total**
Audited	$2c$	0	–	$2c$	$2c$	c	–	$3c$
Confirmed	$2c$	0	$4c^2 - 2c$	$4c^2$	$2c$	-	$4c^2$	$4c^2 + 2c$

Table 5: The computation and communication cost for the proposed DRE-Borda scheme.

7.1 Theoretic Estimate

In order to generate a ballot, the DRE needs to do $2c$ exponentiations. Again, there are c NIZK proofs per ballot. Each NIZK proof requires $4c - 2$ exponentiations. Hence, all c NIZK proofs require $4c^2 - 2c$ exponentiations. All together, each confirmed ballot needs $4c^2$ exponentiations. Note that the audited ballots do not require any proof of well-formedness as all the secret randomnesses used to compute are printed . So, if there are β audited ballots, the total number of exponentiations needed will be $4nc^2 + 2\beta c$. Also, if a single user makes at most $\tilde{\beta}$ audits, then the total number exponentiations that needs to be performed for all the $1 + \tilde{\beta}$ ballots will be $4c^2 + 2c\tilde{\beta}$. For each of the first $\tilde{\beta}$ audited ballots cast by the voter, the DRE has to perform $2c$ exponentiations.

A modern DRE system can perform $2c$ exponentiations much faster than a human can specify her choice by ranking the candidates displayed on the touchscreen of a DRE system. The costliest operation is the computation of the NIZK proof that corresponds to the final confirmed ballot of the voter. This requires $4c^2 - 2c$ exponentiations. However, this step is performed once per voter. Also, the voter's receipt does not include the NIZK proof. Hence, the voter can simply take her receipt(having her ballots printed on it) and can be allowed to leave the polling station while the DRE computes the NIZK proof and uploads them on the bulletin board. The size of each ballot is $2c$. The size of each of the c NIZK proofs is $4c$. Thus, the total amount of storage required per confirmed ballot is $4c^2 + 2c$.

The audited ballots only have the secret keys instead of the NIZK proofs. Each secret key is of size c. The total amount of space required for one audited ballot is given by $3c$. If there are n voters, there will be n confirmed ballots. Thus, the total space required to store all the n confirmed ballot and β audited ballots is given by $4nc^2 + 2nc + 3\beta c$. Also, let $\tilde{\beta}$ be the maximum number audits done by a single user, the total number of bandwidth required per voter will not exceed $4c^2 + 2c + 3c\tilde{\beta}$.

8 CONCLUSION

In this paper, we propose DRE-Borda, a DRE-based end-to-end verifiable Borda count e-voting protocol. Our solution is the first that provides E2E verifiability for a ranked-choice voting system such as Borda count without involving any tallying authorities. Besides the guarantee on the tallying integrity, our system also preserves the privacy of votes: in the case that a DRE machine is completely compromised, what an attacker can learn is limited to only the partial tallying at the time of compromise.

Future work includes extending the scheme to support other more complex ranked choice voting systems such as Single Transferable Vote (STV), which involves not only ranking but also transferring the votes between candidates.

ACKNOWLEDGEMENT

This work is supported by ERC Starting Grant, No. 306994 and Royal Society International Collaboration Award, ICA\R1\180226.

REFERENCES

[1] Michel Abdalla and David Pointcheval. 2005. Simple password-based encrypted key exchange protocols. In *Cryptographers' track at the RSA conference*. Springer, 191–208.

[2] Ben Adida. 2008. Helios: Web-based Open-audit Voting. In *Proceedings of the 17th Conference on Security Symposium (SS'08)*. USENIX Association, Berkeley, CA, USA, 335–348. http://dl.acm.org/citation.cfm?id=1496711.1496734

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

[4] Ben Adida and Ronald L. Rivest. 2006. Scratch & Vote: Self-contained Paper-based Cryptographic Voting. In *Proceedings of the 5th ACM Workshop on Privacy in Electronic Society (WPES '06)*. ACM, New York, NY, USA, 29–40. https://doi.org/10.1145/1179601.1179607

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

[6] Susan Bell, Josh Benaloh, Michael D. Byrne, Dana Debeauvoir, Bryce Eakin, Philip Kortum, Neal McBurnett, Olivier Pereira, Philip B. Stark, Dan S. Wallach, Gail Fisher, Julian Montoya, Michelle Parker, and Michael Winn. 2013. STAR-Vote: A Secure, Transparent, Auditable, and Reliable Voting System. In *2013 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 13)*. USENIX Association, Washington, D.C. https://www.usenix.org/conference/evtwote13/workshop-program/presentation/bell

[7] Josh Benaloh. 2006. Simple Verifiable Elections. *EVT* 6 (2006), 5–5.

[8] J. Benaloh, T. Moran, L. Naish, K. Ramchen, and V. Teague. 2009. Shuffle-Sum: Coercion-Resistant Verifiable Tallying for STV Voting. *IEEE Transactions on Information Forensics and Security* 4, 4 (Dec 2009), 685–698. https://doi.org/10.1109/TIFS.2009.2033757

[9] Josh Benaloh, Tal Moran, Lee Naish, Kim Ramchen, and Vanessa Teague. 2009. Shuffle-sum: coercion-resistant verifiable tallying for STV voting. *IEEE Transactions on Information Forensics and Security* 4, 4 (2009), 685–698.

[10] Jonah Berger. 2018. Undergraduate Council Adopts New Voting Method for Elections. https://www.thecrimson.com/article/2018/9/10/uc-voting-system/

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

[12] David Chaum, Richard Carback, Jeremy Clark, Aleksander Essex, Stefan Popoveniuc, Ronald L. Rivest, Peter Y. A. Ryan, Emily Shen, and Alan T. Sherman. 2008. Scantegrity II: End-to-end Verifiability for Optical Scan Election Systems Using Invisible Ink Confirmation Codes. In *Proceedings of the Conference on Electronic Voting Technology (EVT'08)*. USENIX Association, Berkeley, CA, USA, Article 14, 13 pages. http://dl.acm.org/citation.cfm?id=1496739.1496753

[13] D. Chaum, A. Essex, R. Carback, J. Clark, S. Popoveniuc, A. Sherman, and P. Vora. 2008. Scantegrity: End-to-End Voter-Verifiable Optical- Scan Voting. *IEEE Security & Privacy* 6, 3 (May 2008), 40–46. https://doi.org/10.1109/MSP.2008.70

[14] Kevin Fisher, Richard Carback, and Alan T. Sherman. 2006. Punchscan: Introduction and System Definition of a High-Integrity Election System. In *Workshop on Trustworthy Election. 2006*.

[15] Jon Fraenkel and Bernard Grofman. 2014. The Borda Count and its real-world alternatives: Comparing scoring rules in Nauru and Slovenia. *Australian Journal of Political Science* 49, 2 (2014), 186–205.

[16] Feng Hao, Dylan Clarke, Brian Randell, and Siamak F Shahandashti. 2018. Verifiable Classroom Voting in Practice. *IEEE Security & Privacy* 16, 1 (2018), 72–81.

[17] Feng Hao, Matthew N. Kreeger, Brian Randell, Dylan Clarke, Siamak F. Shahandashti, and Peter Hyun-Jeen Lee. 2014. Every Vote Counts: Ensuring Integrity in Large-Scale Electronic Voting. In *2014 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 14)*. USENIX Association, San Diego, CA. https://www.usenix.org/conference/evtwote14/workshop-program/presentation/hao

[18] James Heather, Peter Y. A. Ryan, and Vanessa Teague. 2010. Pretty Good Democracy for More Expressive Voting Schemes. In *Proceedings of the 15th European Conference on Research in Computer Security (ESORICS'10)*. Springer-Verlag, Berlin, Heidelberg, 405–423. http://dl.acm.org/citation.cfm?id=1888881.1888913

[19] Jason Keller and Joe Kilian. 2008. A Linked-List Approach to Cryptographically Secure Elections Using Instant Runoff Voting. In *Advances in Cryptology - ASIACRYPT 2008*, Josef Pieprzyk (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 198–215.

[20] A. Kiayias, M. Korman, and D. Walluck. 2006. An Internet Voting System Supporting User Privacy. In *2006 22nd Annual Computer Security Applications Conference (ACSAC'06)*. 165–174. https://doi.org/10.1109/ACSAC.2006.12

[21] C. Andrew Neff. 2004. Practical High Certainty Intent Verification for Encrypted Votes.

[22] P. Y. A. Ryan, D. Bismark, J. Heather, S. Schneider, and Z. Xia. 2009. Prêt á voter Voter: a Voter-Verifiable Voting System. *IEEE Transactions on Information Forensics and Security* 4, 4 (Dec 2009), 662–673. https://doi.org/10.1109/TIFS.2009.2033233

[23] Siamak F Shahandashti. 2016. Electoral Systems Used around the World. In *Real-World Electronic Voting (Eds. Hao, Ryan)*. CRC Press, 93–118.

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

[25] Vanessa Teague, Kim Ramchen, and Lee Naish. 2008. Coercion-Resistant Tallying for STV Voting. In *EVT*.

APPENDIX

NIZK Proof

We show the construction of the NIZK proof $\Pi_{il}[x_{i1}, x_{i2}, \ldots, x_{ic} : \tilde{X}_i, B_i]$. We shall use Fiat-Shamir heuristic to convert a standard interactive zero knowledge proof system into a non-interactive one. This NIZK proof is constructed by the DRE machine. The secret input of the DRE is $x_i = (x_{i1}, x_{i2}, \ldots, x_{ic})$. The publicly known inputs are $\tilde{X}_i = (\tilde{x}_{i1}, \tilde{x}_{i2}, \ldots, \tilde{x}_{ic})$ and $B_i = (b_{i1}, b_{i2}, \ldots, b_{ic})$ The statement the DRE needs to prove here is the following:

$$\sigma \equiv (b_{i1} = g^{x_{i1}} g^{a_l}) \vee (b_{i2} = g^{x_{i2}} g^{a_l}) \vee \ldots \vee (b_{ic} = g^{x_{ic}} g^{a_l})$$

This is an OR-statement that consists of c sub-statements. Exactly one of these sub-statements has to be true. Let us assume that the k'th sub-statement is true. That is $b_{ik} = g^{x_{ik}} g^{a_l}$. The prover (DRE) chooses random r_k and computes $com_k = \tilde{g}^{r_k}$ and $com'_k = g^{r_k}$. The prover also chooses $res_1, res_2, \ldots, res_{k-1}, res_{k+1}, res_{k+2}, \ldots, res_c \in_R \mathbb{Z}_q$ and $ch_1, ch_2, \ldots, ch_{k-1}, ch_{k+1}, ch_{k+2}, \ldots, ch_c \in_R \mathbb{Z}_q$ and computes

$$com_j = \tilde{g}^{res_j} \tilde{x}_{ij}^{ch_j}$$

$$com'_j = g^{res_j} (b_{ij}/g^{a_l})^{ch_j}$$

for all $j \in [1, c] \setminus \{k\}$. Let the grand challenge be ch, where

$$ch = Hash(i, l, \tilde{X}_i, B_i, com_1, com_2, \ldots, com_c, com'_1, com'_2, \ldots, com'_c)$$

Here, $Hash(\cdot)$ is a secure hash function. The prover computes $ch_k = ch - \sum_{j \in [1,c] \setminus \{k\}} ch_j$ and $res_k = r_k - ch_k * x_{ik}$.

The verification equations are as below

$$\tilde{g}^{res_j} \stackrel{?}{=} \frac{com_j}{\tilde{x}_{ij}^{ch_j}}; \forall j \in [1, c]$$

$$g^{res_j} \stackrel{?}{=} \frac{com'_j}{(b_{ij}/g^{a_l})^{ch_j}}; \forall j \in [1, c]$$

If these $2c$ equations satisfy, the verification is successful and the proof is correct.

The above NIZK proof comprises c challenges, c responses and $2c$ commitments. Thus, the total space complexity of this NIZK proof is $4c$. The prover needs to perform $4c - 2$ exponentiations, whereas the verifier needs to perform $4c$ exponentiations.