Robust Receipt-Free Election System with Ballot Secrecy and Verifiability

Sherman S.M. Chow* schow@cs.nyu.edu
Joseph K. Liu† ksliu@i2r.a-star.edu.sg
Duncan S. Wong‡ duncan@cityu.edu.hk

Abstract

For most elections, receipt-freeness is important – voters are unable to prove to others on how they voted, in order to prevent vote-buying. Many existing receipt-free electronic voting systems are not practical enough as they require voters to participate in the tallying phase (i.e. do not satisfy the vote-and-go requirement), or have no mechanism for the voters to verify whether their votes have been counted (i.e. do not satisfy universal verifiability).

We propose a new way of constructing vote-and-go election system without tamper-resistant hardware, or anonymous channel. Receipt-freeness is guaranteed even if there is only one voting authority (in a distributed setting) being honest. Regarding the correctness, voter alone has no chance to tamper with the validity of the final tally, while any misbehaving authority can be detected (and proven to the public) by the tallying center. Robustness can be achieved by fixing the corrupted vote in a verifiable manner. Ballot secrecy cannot be compromised even if all tallying authorities collude.

Keywords: *electronic voting, receipt-freeness, universal verifiability, robustness, ballot secrecy, homomorphic encryption, escrowed linkable ring signatures*

1. Introduction

Electronic voting (e-voting) generally means the collection and dissemination of people's opinions with the help of some electronic means (e.g. ballots processing and automated verification, etc). This paper studies the cryptographic design of e-voting system. Cryptographers have been proposing constructions for e-voting since the 1980s [12, 18]. After all these years of research effort, we see some real systems being used over the world; examples can be found in the cryptographic literature such as [4, 37].

Numerous functional and security requirements for e-voting system have been formalized. For example, in [26], seven requirements are listed: completeness, soundness (or robustness), privacy (or ballot secrecy), non-reusability (or double voting detection), eligibility, fairness and verifiability. Recent research has proposed some more desirable properties. One is receipt-freeness [10, 33], which prevents a voter from proving to others that s/he has voted for a particular candidate. Such a feature is essential in order to prevent voters from selling their votes.

1.1. Major Approaches

Over years, many different approaches have been proposed for constructing e-voting systems. We review some of the major ones.

1. *Blind signatures* [13, 26, 50, 49]: Schemes based on blind signatures are simple and efficient, which may be suitable for large scale voting. However, voters are required to obtain some tokens before each voting event, which means that a registration phase is necessary. At the same time, universal verifiability may not be possible since the registrar can just add more ballots to bias the tally. An anonymous channel [12, 31, 51] between the voter and the tallying official is also required to keep the identity of the voter confidential at ballot casting. This requirement may be inhibiting and mix-net [12] may be used to realize it.

2. *Mix-nets* [51, 57, 2, 34, 35, 3, 33, 27, 47, 28, 30, 40, 4, 54]: Mix-net is a network of servers (mix-center) which takes a set of ciphertexts as input, and outputs the corresponding plaintexts according to a secret permutation. Schemes based on mix-nets require the ballots to be permuted at some point between leaving the voters and arriving at the tallying officials. They are generally not efficient since computational effort is required for multiple mixers to prove the correctness. In addition, the anonymity protection comes from the

*Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, NY 10012, USA.
†Institute for Infocomm Research (I^2R), Singapore.
‡Department of Computer Science, City University of Hong Kong, Hong Kong, China.

mix-net, the function of which may possibly be performed by the tallying center in a real world scenario. Thus, the tallying center must be trusted. This problem can be *partially* solved by distributing it over many machines so that trust is placed only on a threshold number of them.

3. *Homomorphic encryption* [9, 56, 20, 21, 33, 32, 42, 22, 41, 53, 39, 4]: Voting schemes using homomorphic encryption are efficient in general. Extensive research using this approach had been done, but only a few of them (such as [33, 32]) can achieve receipt-freeness.

4. *Ring signatures* [44]: Roughly speaking, a ring signature is a signer-ambiguous signature. Each voter signs on her/his vote by a ring signature to confirm her/his voting. The ambiguity in the real identity of the signer protects the voter-vote relationship from being known. However, some special mechanism is required to detect double voting. This also motivates the introduction of the linkability in the ring signature paradigm [44], so two ring signatures produced by the same signer can be linked.

The suitability of each of these four types varies with the conditions under which it is to be applied.

1.2. Receipt-Freeness

Generally speaking, receipt-freeness means a voter cannot prove to a third party that he has voted for a particular candidate. In other words, vote-buying is not possible. In an ordinary *mix-net* or *homomorphic encryption* based scheme, an adversary can simply furnish the voter with a ciphertext on a particular candidate, and then verify that the voter has posted a ballot containing that ciphertext. The *blind signature* based schemes suffer from similar vulnerabilities. For *ring signature* based schemes, a voter can simply reveal the randomness of the ring signature he has posted to show that he is the actual signer. Since the randomness is the source of signer-anonymity, doing so can claim the authorship of a particular ballot.

The first scheme claimed to be receipt-free was proposed by Benaloh and Tuinstra [10], although it was later shown to be not [33]. Okamoto [50] proposed a scheme with receipt-freeness that makes use of blind signatures. It assumes the existence of an *anonymous untappable channel* and requires three times of voter and system *interaction* in the course of an election. That is, they require every voter to participate even in the tallying phase. Sako and Kilian [57] proposed an e-voting scheme with multi-authority using mix-net and homomorphic encryption, which is postulated to be receipt-free. Even though its receipt-freeness cannot be guaranteed under the commonly used assumption that only one mix-center is honest, it served as a basis for the later work of Hirt and Sako [33] and a more efficient approach in [8].

Receipt-freeness comes with some assumptions. All of the above but [50] have the receipt available if the adversary can corrupt even one of the tallying authorities in a distributed setting. As noted in [37], systems in [33] and [8] still retain receipt-freeness when such a corruption takes place, but only under a stronger assumption that the voter knows which tallying authorities have been corrupted. Juels, Catalano and Jakobsson [37] proposed a receipt-free voting scheme, which is one of the most efficient and practical schemes to date. It only requires one of the tallying authorities to be honest for providing receipt-freeness while the ballot secrecy depends on the anonymous channel and the honesty of at least one among the group of the registrars (entities responsible for registration) and tallying authorities. In addition, it can even provide a stronger form of receipt-freeness, called as *coercion-resistance*, which defends against randomization, forced-abstention and simulation attacks (details can be found in [37]). However, coercion-resistance does not allow the voters to verify whether their votes have been counted (i.e., no universal verifiability).

Besides the above schemes, most of the other schemes such as [42, 38, 41, 31, 40, 45] require either tamper-resistant hardware or a randomizer to provide receipt-freeness.

1.3. Our Contribution

In this paper, we make use of the *escrowed* linkable ring signature [17] to get the robustness and the receipt-freeness simultaneously. The unforgeability (in an appropriate sense in the context of escrowed linkable ring signature) gives the robustness, so the misbehavior in preparing an invalid ballot can later be attributed to some one (either a voting center or a voter) in a non-repudiable manner. Receipt-freeness comes from the signer-ambiguity and the private linkability of the ring signature.

Even though it has been previously suggested that linkable ring signature can be used to construct e-voting system [44], we believe that our study provides a better and more comprehensive solution. We advocate using identity-based linkable ring signatures to address the problem of requiring all potential voters to get their public keys. We provide more than the double voting detection mechanism brought by *plain* linkable ring signatures. Indeed, our application of escrowed linkable ring signatures is fundamentally different from the use of linkable ring signatures or linkable credentials in previous simple e-voting systems [44, 45].

Our proposal enjoys the following major properties:

- RECEIPT-FREENESS: Our scheme provides receipt-

freeness without using any randomizer or tamper-resistant device. We only make the basic assumption that at least one of *arbitrary* voting authorities remains honest. In our scheme, the honest authority can be *unknown* to the voters. All they need to believe is at least one of them (no matter which) is honest, in contrast with the schemes in [33, 8].

- UNIVERSAL VERIFIABILITY AND CORRECTNESS: Our scheme provides universal verifiability in contrast with Juels *et al.*'s scheme [37] which is not universal verifiable. Moreover, their scheme relies on the assumption that the adversary can only corrupt a minority of tallying authorities. We observe that corrupting all of them means the encrypted credential posted by voters in the bulletin board can be decrypted. With such a credential, an adversary can pretend to be a voter to cast a vote and affect the result. (This observation does not contradict to the security result claimed in [37].)

  Regarding the correctness of our scheme, voter alone has no chance to tamper with the validity of the final tally. Any misbehaving party who introduces corrupted votes can be detected by tallying authorities. At the same time, tallying authorities can prove to the public about this. Our scheme is also robust in the sense that the corrupted votes can be fixed, again in a publicly verifiable manner.

- BALLOT SECRECY: In our scheme, voter contributes to the randomness introduced in the encryption of choice, which means even the collusion of all voting authorities cannot reveal a particular ballot. Since a threshold number of tallying authorities can come together and open any encrypted ballot, the voter-vote relationship is under protection if the majority of tallying authorities or all voting authorities are honest.

- NO REGISTRATION/AUTHORIZATION STAGE: Unlike many other schemes, especially those using anonymous channel (such as [50, 37]), our scheme does not require any registration stage. In these schemes, the voter is assumed to have a private/public key pair (more generally, a credential and its public representation [37]) for proving eligibility to the registrar in the registration stage. In other words, by saying registration-free we do not mean the voters are free from getting a public key certificate from a public key infrastructure (PKI).

  Our scheme employs linkable ring signatures to remove the registration stage. Nevertheless, the generation of such signatures involves the public keys of all legitimate voters. This requirement can be removed if identity-based (ID-based) infrastructure (in particular, ID-based linkable ring signature scheme [6, 17]) is used instead, such that those who want to vote will not be affected by those who do not bother to get the private key of their identities. Of course, just like the commonly used trust assumption that the certificate authority of PKI would not launch an active impersonate attack, we make a similar assumption for our ID-based case. Advantages of ID-based ring signatures over traditional PKI based one are discussed in [16].

- VOTE-AND-GO: Once the voter has cast her/his vote, our system requires no further action from the voter to get the final voting result, which is a nice feature since voters may find additional duties tedious. For example, the voter is required to open the trapdoor commitment in [50], or the "encrypted" ballot cannot be opened. On the other hand, our scheme is publicly verifiable, so a voter who wants to verify the correctness of the result can participate in a public audit after the final tallying phase, which may make the voters feel that their participation is important or dutiful.

In addition, our scheme supports double voting detection, and more than two candidates (i.e. not restricted to a yes/no vote).

Table 1 summarizes the comparison of different receipt-free voting schemes without using any tamper-resistance device or randomizer.

Organization. In Section 2, we describe the framework of an e-voting system. In Section 3, we describe the primitives from which our scheme is built, before the description of our e-voting scheme in Section 4. Finally, some concluding remarks are made at the end of the paper.

2. Voting Model

2.1. Entities

There are three types of entities in the system.

Voters: Voters are the people who are authorized to vote. In the rest of the paper, we use U to denote a particular voter who is about to vote. Let (x_U, y_U) be the private/public key pair of U.

Voting Center/Voting Authorities: We use the term voting *center* to refer to the logical entity responsible for organizing the voting. Its task may be separated into $N_V > 1$ voting *authorities* $V_1, \ldots, V_{N_V}$.

Tallying Center/Tallying Authorities: Tallying center is responsible for tallying and publishing voting results. Let (x_T, y_T) be the private/public key pair of the center. Again, the task of a tallying center can be distributed to N_T authorities, denoted by $T_1, \ldots, T_{N_T}$.

Schemes	Receipt-free (only one tallying authority is honest)	Universal Verifiable	Without Anonymous Channel	Ballot secrecy (all tallying authorities collude)	Without Registration Stage	Vote-and-Go
Okamoto [50]	✓	✓	✘	✓	✘	✘
Sako-Kilian [57]	✘	✓	✘	✓	✓	✓
Hirt-Sako [33]	✓	✓	✓	✘	✓	✓
Baudron *et al.* [8]	✓	✓	✓	✘	✓	✓
Juels *et al.* [37]	✓	✘	✘	✓	✘	✓
Our scheme	✓	✓	✓	✓	✓	✓

Table 1. Properties of different receipt-free voting schemes

Besides the entities above, we also assume the existence of the following object.

Bulletin Board: It can be considered as a broadcast channel with memory, such that (1) all entities can read the content of the board; (2) any entity can append information onto the board; and (3) no one can remove any information that has already been written to the board. It is commonly used for achieving universal verifiability, e.g. [21].

The bulletin board can be modeled formally by a stateful oracle with two queries post and get. The query post takes a message $m \in \{0,1\}^*$ and stores m in its state $\mathcal{BB}$, without giving any return value. The query get takes no input and returns $\mathcal{BB}$.

2.2. Functions

Suppose there are n election candidates (choices) and they are indexed from 1 to n. Let L be the list of public keys of all the voters. Our voting protocol proposed in the subsequent section is composed of the following functions.

Setup: $param \leftarrow \mathsf{Setup}(1^k)$ is a randomized algorithm that takes a security parameter k and outputs the system parameter that will be used by all the other algorithms. In particular, $param$ includes a unique event identifier eID, the cryptographic security parameter k and the description of all cryptographic algorithms to be used. For brevity, we will omit the inclusion of system parameter as part of the input.

Key Generation: $(x, y) \leftarrow \mathsf{Gen}()$ is a randomized algorithm that outputs a private key x and the corresponding public key y. For simplicity of the discussion, we omit the details of master key generation and the user secret key generation in the ID-based case.

Voting: $ballot_U(\beta) \leftarrow \mathsf{Vote}(x_U, y_T, L, \beta)$ is a randomized algorithm that takes U's private key x_U, the public key of the tallying center y_T, the list of public keys of all legitimate voters L and a candidate selection $\beta \in \{1, \cdots, n\}$; outputs some binary string $ballot_U(\beta)$ to be posted on the bullet board $\mathcal{BB}$.

The above description captures the essential characteristics of the voting phase of any election system. In our scheme, voting is done via an interactive protocol between the voter and the voting center, where the voting center uses its private key as its secret input, and the voter also takes the public key of the voting center as the public input.

Tallying: $(\mathcal{X}, P) \leftarrow \mathsf{Tallying}(x_T, L, \mathcal{BB})$ is a deterministic algorithm, which takes the private key of the tallying center x_T, a list of public keys of all legitimate voters L and the full contents of the bulletin board $\mathcal{BB}$; outputs a vote tally $\mathcal{X}$, along with a non-interactive proof P that the tally was correctly computed.

Verification: $0/1 \leftarrow \mathsf{TallyVerify}(y_T, L, \mathcal{BB}, \mathcal{X}, P)$ is a deterministic algorithm which takes the public key of the tallying center y_T, a list of public keys of all legitimate voters L, the full contents of the bulletin board $\mathcal{BB}$, the vote tally $\mathcal{X}$ and the associated non-interactive proof P for correct tallying. Its output is either 0 or 1, meaning $invalid$ or $valid$ respectively. In our scheme, the public key of the voting center is also involved.

2.3. Real World Deployment Assumptions

We assume that, once the communication between the voter and the voting center starts, a vote-buyer (or any third party) is not allowed to communicate with the voter. That is, we assume that the voting center is really communicating with the purported voter, not a vote-buyer or other adversary. We also assume that the vote-buyer is not allowed to view or video record the whole process of the voting phase (for example, he stands behind the voter and watches what he has typed in the computer). Similar to [33], we assume untappable channels between the voters and the voting authorities, which is also related to a recent result [14] that one cannot get universal verifiability and receipt-freeness

simultaneously unless the voting process involves interactions between voters and possibly the voting authority.

3. Cryptographic Primitives

Our scheme uses the following cryptographic primitives.

3.1. Proof of Knowledge

A Σ-protocol for an NP-relation R is a two-party three-round protocol, such that for every input $(x, y) \in R$ to a prover $\mathcal{P}$ and y to a verifier $\mathcal{V}$, $\mathcal{P}$ firstly sends a commitment message a, $\mathcal{V}$ replies with a random challenge c and $\mathcal{P}$ concludes with a response message z. At the end $\mathcal{V}$ outputs a $0/1$ value locally, depending on y and the communication transcript (a, c, z) only; a transcript is valid if the output of $\mathcal{V}$ is 1. In addition, a Σ-protocol must satisfy the following two properties.

(Special Soundness.) There exists a probabilistic polynomial time (PPT) extractor takes an input y and two valid transcripts with the same commitment (a, c, z) and (a, c, z'), outputs x such that $(x, y) \in R$.

(Special Honest-Verifier Zero-Knowledge.) There exists a PPT simulator which takes input y and any challenge c, output a pair (a, z) such that (a, c, z) is a valid transcript and is distributed according to the probability distribution of one returned by the interaction of $\mathcal{P}$ and $\mathcal{V}$ via $\mathcal{P}(x, y) \leftrightarrow \mathcal{V}(y)$, for any y where $(x, y) \in R$.

A Σ-protocol can be generalized to an 1-out-of-n witness indistinguishable proof [19] as illustrated in the Appendix A. Moreover, it can be made non-interactive. By the Fiat-Shamir heuristics [25], every three rounds proof of knowledge protocols that is honest-verifier zero-knowledge can be turned into a non-interactive proof-of-knowledge protocol by setting the challenge to the hash value of the commitment a together with other auxiliary data involved.

3.2. Designated Verifier Proof of Knowledge

Using the Fiat-Shamir technique to make the proof-of-knowledge non-interactive makes the resulting "transcript" in itself a transferable proof, the correctness of which can be verified by anyone. This universal verifiability may not be desirable in some occasions.

Jakobsson *et al*. [36] constructed a non-interactive designated verifier proof. The designated verifier can always use his trapdoor to simulate a transcript for any statement. Any other party cannot distinguish a valid proof of a true statement from a simulated proof.

Designated verifier proof is also used in some existing e-voting system (e.g. [33]).

3.3. (Escrowed) Linkable Ring Signatures

The idea of ring signatures was formalized by Rivest *et al*. [55]. Ring signature schemes enable one to sign on a message in a way that anyone can ensure the signature is generated by some one from a group of signers which includes the real signer, but do not know exactly who. The diversion group of signers is formed spontaneously, which means ones may be totally unaware that they are involved in a ring signature. Several ring signature schemes have been proposed [23, 11, 15]. Many of them enjoy unconditional anonymity– no one can later reveal the actual signer, other than the actual signer herself/himself.

Note that a two-party ring signature can be used as a designated verifier signature, i.e. only the designated verifier gets convinced that the signature is valid, but cannot prove to anyone else.

Traditional ring signature schemes support unlinkability: it is not possible to decide whether two signatures have been produced by the same group member. The notion of linkability in ring signature schemes was first introduced by Liu, Wei and Wong [44]. In addition to the 1-out-of-n signer-anonymity, it also allows anyone to determine whether two signatures have been produced by the same signer. The linkability is determined by a linkability tag that must present in a valid ring signature. Given an event identifier and the user's private signing key, the linkability tag is uniquely determined. In this way, ring signatures for the same event from the same user can be linked.

The first scheme in [44] uses a single type of public key while the scheme in [60] allows different types to be used together. Schemes in ID-based [17, 6] and certificate-based [7] settings are proposed later on. Schemes in [44, 60] have signature size linear in the size of the diversion group. Examples of constant-size linkable scheme include the ID-based scheme in [17, 6] and the PKI-based scheme in [5].

Chow, Susilo and Yuen made a further refinement of the anonymity of ring signatures [17] such that the linkability can be *escrowed*, i.e. the linking can only be done by a linking authority. While the linkability tag is still a deterministic one as in [17], it is encrypted by a probabilistic encryption algorithm, and it is computationally impossible to prove the non-authorship of someone else's signature.

Our construction uses the escrowed linkable ring signatures to realize the designated verifier signatures, so only the voter can verify and only the linking authority (in which the function can be performed by the tallying center in our system) can later reveal who is the actual signer, which help providing the receipt-freeness and the verifiability.

Looking ahead, linkable ring signatures will help in double voting detection, and escrowed linkable ring signatures ensure the detection of misbehaving voting authority and the robustness (by signing the re-randomization introduced

in an "untransferable" manner, i.e. the signature is only convincing to the designated verifier but no one else).

3.4. Functions Notation for Signatures

We define a set of signature primitive functions here.

- $\sigma \leftarrow \mathsf{Sign}_V(m)$ takes an input of a message m and the private key of the voting center V, returns a normal signature σ signed by V.
- $0/1 \leftarrow \mathsf{SigVerify}_V(m, \sigma)$ takes an input of message/signature pair (m, σ), and the public key of V, returns 1 if the message/signature pair is valid and 0 otherwise.
- $\sigma^* \leftarrow \mathsf{LRingSign}_{L,eID}(m)$ takes an input of a message m, the public keys of all members in a group L, a unique event identifier eID, and a private key of one of the members in the group L, returns a linkable ring signature σ^*.
- $0/1 \leftarrow \mathsf{LRingVerify}_{L,eID}(m, \sigma^*)$ takes an input of message/signature pair (m, σ^*), the public keys of all members in the group L, a unique event identifier eID, returns 1 if σ^* is a signature on m signed by someone in the group L for the event eID, and 0 otherwise.
- $\sigma^* \leftarrow \mathsf{ELRingSign}_{L,eID}(m, pk_{\ell a})$ takes an input of a message m, the public keys of all members in a group L, a unique event identifier eID, a private key of one of the members in the group L, and the public key of the linking authority $pk_{\ell a}$, returns an escrowed linkable ring signature σ^*.
- $0/1 \leftarrow \mathsf{ELRingVerify}_{L,eID}(m, \sigma^*, pk_{\ell a})$ takes an input of message/signature pair (m, σ^*), the public key of the linking authority $pk_{\ell a}$, the public keys of all members in the group L, and a unique event identifier eID, returns 1 if σ^* is a signature on m signed by someone in the group L and the linkability tag is verifiably encrypted to ℓa, 0 otherwise.
- $0/1 \leftarrow \mathsf{Link}_{eID}(\sigma_0, \sigma_1[, sk_{\ell a}])$ takes an input of two valid signatures (σ_0, σ_1), a unique event identifier eID, and possibly the secret key of the linking authority $sk_{\ell a}$, returns 1 if both signatures are issued by the same signer for the same event eID, 0 otherwise.

3.5. Homomorphic Encryption

For the ease of understanding, we use ElGamal encryption in our scheme, which is reviewed in Appendix B. We remark there are other homomorphic encryption schemes like Paillier encryption [22].

3.6. Encoding of Candidates

Recall $|L|$ is the number of legitimate voters, suppose candidate i gets c_i votes from these $|L|$ voters, let $\gamma \in G$ be a generator chosen randomly and ℓ be another integer such that $\forall i \in \{1, \cdots, n\} : c_i < \ell$ (e.g. $\ell = |L| + 1$); we use an idea from [20] to encode a choice of candidate as $\gamma^{\ell^{i-1}}$. For examples, candidate 1 is encoded as γ, while candidate n is encoded as $\gamma^{\ell^{n-1}}$.

Since we encode the choices in the exponent and we are going to aggregate the ciphertexts by the homomorphic property, it is crucial that parameters are chosen so that $\ell^n < q$, where q is the order of the group $\mathbb{G}$, to ensure the final tally is uniquely represented in $\mathbb{Z}_q$.

4. Proposed Construction

4.1. Initial Attempts

We first discuss how naïve attempts of combining cryptographic schemes (even each of them has some nice properties) would not work.

Homomorphic encryption makes the final decryption by tallying center efficient since only one decryption of the "combined" encrypted vote is needed. Unfortunately, any stateless encryption scheme with semantic security (informally, it means the ciphertexts encrypting two known messages respectively are indistinguishable) involves randomness in the encryption. The random factor introduced in the encryption process makes the encryption of a particular message reproducible, and hence it can be used by the voter as a piece of evidence that a certain vote has been cast.

To provide receipt-freeness, the encryption can be done in the other way round, i.e. a voting center encrypts on the voter's behalf. Instead of revealing the randomness, the center can produce a proof of knowledge that the encrypted vote is a valid one for the voter's choice. It is also possible to strengthen the security by introducing more voting authorities to perform re-randomization. However, the ballot secrecy is lost.

For the voter to hide the voting choice, a natural solution is to ask for the ciphertext of all candidates, and submitting only one of them finally, which is clearly inefficient. Moreover, the collusion of all voting authorities can reveal the voter's choice by a comparison between the submitted vote and the re-encryption transcript, even this voter uses an anonymous channel to obtain those ciphertexts.

One may suggest requiring both the voter and the voting center jointly contribute the randomness in the encryption process. Nevertheless, not to forget universal verifiability and correctness. For a correct combined decryption of the final result, each ciphertext must be well-formed. This requires the voting center to prove to the voter that its con-

tribution of the ciphertext is well-formed, without revealing the randomness. Zero-knowledge challenge-response protocol seems to be useful here, but it is entirely possible that the voter has brought the challenge furnished by the vote-buyer to participate in the protocol. In other words, the ability for anyone to audit such a process may enable a voter to convince the vote-buyer that a certain vote has been cast.

4.2. High Level Idea

Now we describe the main idea to solve all the aforementioned problems. Our e-voting scheme combines the useful properties from homomorphic encryptions, (witness-indistinguishable) zero-knowledge proof (of equality of discrete logarithm), designated verifier signature and (escrowed) linkable ring signatures as follows.

A voting center first prepares an encryption of a multiplicative identity message (i.e. $m = 1$), denoted as C_D, sends to the voters and uses the zero-knowledge proof (ZKP) protocol to convince the voter that it is a valid one. The voter, now encrypts her/his choice and multiplies the resulting ciphertext with C_D to get C_E. As C_D is an encryption of a multiplicative identity, C_E is just an encryption of the voter's choice. Clearly, the voting center cannot decrypt since it does not know the randomness introduced by the voter (nor the private key of the tallying center). The validity of the ciphertext can be checked by the center using a witness-indistinguishable ZKP that division of C_E by C_D is a valid encrypted vote of a certain candidate.

Informally, the receipt-freeness is achieved from the following facts.

1. The voter does not know the randomness introduced by the voting center.
2. In the single voting authority version, the voting authority never signs on C_D.
3. In the multiple voting authorities version, the authorities only sign on C_D using an escrowed linkable ring signature, which means the voter can come up with the same signature.
4. Thus, the voter can always make the corresponding C_D to claim that C_E is a valid encrypted vote of a certain candidate, even it is not the case (also from the properties of the designated verifier proof-of-knowledge protocol).

With only one voting center, a temporary break-in (leaking the secret random factor) makes receipt available. We can increase the security level by introducing a series of voting authorities to perform the voting center's function. To ensure all authorities have been involved in the re-randomization, they communicate among themselves in such a way that one authority only signs on the partial ciphertext of the voter if a valid signature from a previous authority is presented. The final encrypted vote is considered to be invalid if a valid signature from the last authority is absent. This gives a proof for the occurrences of interactions with all voting authorities, but this cannot give us robustness, since the signatures never go to the voter's hands.

To give the voter some piece of evidence in case a voting authority produced an invalid encrypted vote, the voting authorities must give some forms of signature to the voter. However, care must be taken on how to sign. Signing C_D makes receipt available since vote-buyer would know C_D is not constructed by the voter but the authority. On the other hand, using a designated verifier signature to sign C_D, with the voter designated as the verifier, makes it impossible to identify whether the voting authority or the voter misbehaved (when C_D corresponds to some inconsistent re-randomization, there is equal chance that the authority or the voter gives such a signature). That is why something in between, i.e. a linkable ring signature, should be used.

An escrowed linkable ring signature should be used instead of a plain linkable ring signature; otherwise, the voter can just produce another linkable ring signature and convince the vote-buyer that the voting center must have signed (given there are only two possible signing entities) since the public linkability gives evidence that these two signatures are produced by two different signers. As discussed before, even it is still possible to show the authorship of a newly-created escrowed linkable ring signature, one cannot prove to other the non-authorship of one produced by someone else. So the voter has no way to show the vote-buyer who is the real signer of the one presented by a voting authority.

Now a valid receipt-free encrypted vote is finally obtained, the voter can declare it is her/his choice by using linkable ring signature. This is also the idea suggested in [44] that how a linkable ring signature helps in a simple e-voting system. No one knows who the real vote issuer is, but one can still get convinced that there is no double voting by the public linkability checking.

4.3. Single Voting Authority Version

4.3.1 Voting Phase

1. The voter U randomly chooses $r \in \mathbb{Z}_q$, and sends $R = g^r$ to V.
2. The voting authority V encrypts an identity message as follows.
   (a) Randomly chooses $s \in \mathbb{Z}_q$.
   (b) Computes $(a, b) = (g^s, {y_T}^s)$.

   (Note: y_T is the public key of the Tallying Authority.

3. V proves to U, using non-interactive designated verifier proof-of-knowledge, that (a, b) is indeed an encryption of 1, by proving the discrete logarithm of a to the base g is the same as the discrete logarithm of b to the base y_T.

4. After verifying the proof, U encrypts her/his choice of candidate k, and multiplies the resultant ciphertext with the ciphertext (a, b) s/he just obtained, as

$$(A_U, B_U) = (a \cdot g^r, b \cdot m_k \cdot {y_T}^r)$$

where m_k represents the encoding of the choice of candidate k.

5. U and V engages in a witness indistinguishable proof to show that U performs the steps faithfully, by proving the discrete logarithm of $R = A_U/a$ to the base g is the same as the discrete logarithm of $B_U/b/m_k$ to the base y_T for an valid encoding m_k. (We remark that the proof is carried out in a zero knowledge sense that V learns nothing about the value of $B_U/b/m_k$, or m_k. Details of the construction of this protocol can be found in the Appendix.)

6. If V accepts U's proof, V gives a signature $\sigma_V = \mathsf{Sign}(A_U, B_U)$ to U.

7. If the signature σ is valid, U generates a linkable ring signature $\sigma_U^* = \mathsf{LRingSign}_{L,eID}((A_U, B_U))$, and posts (A_U, B_U), σ_V and σ_U^* to the bulletin board.

Note that some messages of two proof-of-knowledge protocols can be piggy-backed – the challenge of U and the ciphertext generated by U can be sent together, and the response of V for the first proof and the challenge of V for the second proof can also be sent in a single message.

4.3.2 Vote Tallying Phase

The tallying center (i.e. each of the tallying authorities $T_1, \ldots, T_{N_T}$) verifies each ballot, computes the tally and publishes the result as follows. Note that all verifications below can also be done by the public.

1. Each vote $((A_U, B_U), \sigma_V, \sigma_U^*)$ is firstly verified by $\mathsf{LRingVerify}_{L,eID}((A_U, B_U), \sigma_U^*)$. The votes are discarded if the corresponding linkable ring signatures are invalid or linked (i.e. voted twice). For scheme using a deterministic linkability tag (uniquely determined by the actual signer's private key and an event identifier), such a checking can be done using the linkability tag as a key to a hash table and collision detection.

2. For those votes passed the previous checks, the tallying center verifies σ_V is a valid signature on (A_U, B_U), and discard any invalid votes.

3. Anyone among the tallying authorities multiplies all valid ciphertexts to obtain $(\mathfrak{A}, \mathfrak{B}) = (\prod A_U, \prod B_U)$.

4. Each tallying authority T_i (with share x_i) publishes $w_i = \mathfrak{A}^{x_i}$ and proves its correctness in zero-knowledge. Let Λ denote some set of t tallying authorities. The tally can be computed as $v = \mathfrak{B}/\prod_{j\in\Lambda} w_j^{\lambda_j}$, where λ_j is the Lagrange coefficient polynomials (as defined in equation (1) in Appendix B), and this result v is publicly verifiable since anyone can access $\{w_j\}_{1\le j\le N_T}$ and recover v. For details, please refer to Appendix B.

5. The tallying authorities get the result $\mathcal{R} = c_1 + c_2\ell + c_3\ell^2 + \cdots + c_n\ell^{n-1}$ by computing the discrete logarithm of v to the base γ and publish $\mathcal{R}$. The most straightforward way to do this is to try every possible $\mathcal{R}$ to see if it is the correct one, which takes $O(\ell^n)$ operations. We can speed up this process to $O(\sqrt{\ell^n})$ by using Shank's baby step/giant step algorithm (please refer to Appendix B.2). After we get $\mathcal{R}$, we can regard it as a number to the radix ℓ and obtain $c_1, c_2, \ldots, c_n$.

We remark that if Paillier encryption [22] is used instead, computation of discrete logarithm is easy and Shank's algorithm is not needed. For a comparison of using ElGamal and Paillier in homomorphic encryption based e-voting system, one may consult [53].

4.4. Full Security Version

The scheme above assumes the voting center will not collude with the voter. If such collusion occurs, the receipt-freeness is gone since all random factors are known. The correctness of the tally can be easily broken even if only one vote is invalid. Such collusion also breaks the correctness since all the proof of knowledge is only done between them and no proof is given to the tallying center or the public.

Below gives a strengthened version with N_V voting authorities performing the voting center function. A self-regulating mechanism is used – verification will be done by an authority for checking the participation of another authority, to counter the above attacks.

Note the notation of σ to denote normal signatures and the $*$ superscript in σ^* to denote ring signatures which the real signer is hidden.

4.4.1 Voting Phase

1. The voter U randomly chooses $r \in \mathbb{Z}_q$, sends $a_0 = g^r$ and a pseudonym P_U to V_1.

2. U also computes $b_0 = {y_T}^r m_k$ where m_k is her/his choice of candidate.

   For i = 1 to $|N_V|$ Do:

3. A voting authority V_i randomly chooses $s_i \in \mathbb{Z}_q$ and computes $(a_i, b_i) = (g^{s_i}, {y_T}^{s_i})$.

4. V_i generates an escrowed linkable ring signature $\sigma^*_{V_i} = \mathsf{ELRingSign}_{(V_i,U),eID}((a_i, b_i, P_U), y_T)$ and gives (a_i, b_i) together with signature $\sigma^*_{V_i}$ to the voter and the tallying center. This provides evidence that how V_i has participated in the re-randomization.

5. U verifies the escrowed linkable ring signature provided by V_i is valid. Then V_i proves to U by a non-interactive designated verifier proof that (a_i, b_i) is indeed an encryption of 1.

6. If the proof is invalid, V_i reports to the tallying center by signing $\sigma^*_U = \mathsf{ELRingSign}_{(V_i,U),e}(\text{“report”}, y_T)$ and presenting both σ^*_U and $\sigma^*_{V_i}$.

   (The tallying center then gets the linkability tags by using its secret key, and checks if they are different.)

7. If $i = 1$, U and V_1 engages in a witness indistinguishable proof for showing the voter has prepared a valid vote, by proving the discrete logarithm of a_0 to the base g is the same as the discrete logarithm of b_0/m_k to the base y_T for a valid encoding m_k representing the choice of the candidate k.

   If $i > 1$, V_i receives $(A_{i-1} = g^r \prod_{j<i} a_j, B_{i-1} = {y_T}^r m_k \prod_{j<i} b_j)$, and the signatures $\{\sigma_{V_j}, j < i\}$ from V_{i-1}, where m_k is the choice made by the voter. V_i proceeds only if verifications pass. Otherwise, re-request for signatures or report to the tallying center.

8. If verification goes through (either checking U for $i = 1$ or checking V_{i-1} for $i > 1$), V_i computes $(A_i = a_i \cdot A_{i-1}, B_i = b_i \cdot B_{i-1})$ and sends to U.

9. If $i < N_V$, V_i generates a signature $\sigma_{V_i} = \mathsf{Sign}_{V_i}((A_i, B_i, P_U))$, and sends (A_i, B_i) with the signatures $\{\sigma_{V_j}, j \leq i\}$ to V_{i+1} via a secure channel.

   If $i = N_V$, V_{N_V} gives a signature $\sigma_{V_{N_V}} = \mathsf{Sign}_{V_{N_V}}((A_{N_V}, B_{N_V}, P_U))$ to U.

   This step is for the “transition of trust”. For efficiency of the next authority’s verification, this signature can be replaced by a message authentication code (MAC) with secret key shared between all authorities.

10. U verifies that $A_i = a_i \cdot A_{i-1}$ and $B_i = b_i \cdot B_{i-1}$.

    End of For Loop

11. U posts $m_U = (A_{N_V}, B_{N_V})||\sigma_{V_{N_V}}||P_U$ together with the linkable ring signature $\sigma^*_U = \mathsf{LRingSign}_L(m_U)$ to the bulletin board.

Note that even the number of voting authorities has increased, the voter only needs to involve in the proof-of-knowledge protocol (as a prover) once (with V_1). Only a mild computational burden is incurred.

This also means that if the first voting authority V_1 colludes with the voter, no proof has been made about the well-formness of the encrypted vote. However, note that V_1 has signed on the encrypted vote, the corrupted part is “marked” and we can always cancel it and frame V_1 later.

4.4.2 Vote Tallying Phase

Vote tallying proceeds as in the basic version.

4.5. Security Analysis

Receipt-freeness. Since the protocol starts by requiring the voter to “commit” firstly to the random factor to be used in the ciphertext, this random factor cannot be chosen in a way depending on the other random factors to be introduced by the voting authorities. The final ciphertext posted on the bulletin board is $(A = g^{r+\sum_j s_j}, B = {y_T}^{r+\sum_j s_j} m_k)$ where m_k is the choice of the candidate. When an arbitrary voting authority (e.g. V_ℓ) is honest, U does not know the randomness $R = r + \sum_j s_j$ since s_ℓ is unknown. The best s/he can do is to show the knowledge $r' = R - s_\ell$ such that $log_{y_T}(B/B'/m_k) = r'$, where B' is supposed to be ${y_T}^{r'}$ that “cancel” all the components that s/he does not know the corresponding exponent.

However, the validity of B' cannot be shown with decisional Diffie-Hellman assumption. For any m_k, one can just randomly selects $r' \in \mathbb{Z}_p$ and computes B' satisfying the equation $(B/B'/m_k) = y^{r'}$. Such a property makes “extraction” of receipt is impossible.

As discussed before, if the authority V_i signs $(a_i = g^{s_i}, b_i = {y_T}^{s_i})$ receipt can be made. This is where escrowed linkable ring signature comes to play. By generating a ring signature with the signer group being the voter and the authority, both the voter and the authority can generate such a signature. The voter cannot convince any other about the “validity” of B'.

With the real world deployment assumptions we have made, physical recording and masquerade are not possible. The designated-verifier proof-of-knowledge protocols are executed in an interactive manner, which convinces no one else about the validity of the statements to be proven.

Two more points to note. The signature on the pseudonym provides no clue about the candidate choice. Knowledge of any randomness involved in the

linkable ring signature cannot prove a vote on a particular candidate is cast.

Completeness and Universal Verifiability. In the voting phase, the voter proves to each authority that the encryption s/he produces, after canceling the factors contributed by the authorities, is indeed the encryption of a valid encoding of one of the candidates.

For the part contributed by the authorities, note that each authority V_i uses escrowed linkable ring signature to sign (a_i, b_i, P_{U_k}) for each voter U_k. The tallying center can easily check whether V_i has cheated in any re-randomization process by checking if $\prod\{a_i\}^{x_T} = \prod\{b_i\}$.

Either the voting authority or the voter has generated this signature. In case of the dispute, the concerned voter issues an escrowed linkable ring signature as well. The tallying center can confirm who the cheater is by the linkability. Any cheating behavior can be shown to the public by a non-interactive proof.

Given the data on the bulletin board, it is possible for anyone to verify which encrypted votes are valid: those that are signed by the voting authorities (as shown by a valid signature by the last voting authority) which means the re-randomization processes have been taken place, and with a linkable ring signature produced by a voter who has not signed any other votes.

Note the voter should post the diversion signer group of the signature if the voter decided not to include all the legitimate voters in the signers group by any reason. This is necessary since the verification algorithm also takes the signers group as an input.

Given all the encrypted votes, anyone can aggregate them and compute the final encrypted tally. Using the tally center's private key, the tally center is able to produce a non-interactive proof that the result it publishes is the correct decryption of the encrypted tally.

Robustness. Locating invalid re-randomization can be done by a binary search [46]. For the correction of the tally, the corrupted vote is firstly located by using the pseudonym as the index of the vote. Then, the problematic a_i and b_i can be removed from $\mathfrak{A}$ and $\mathfrak{B}$ respectively. If it is the user who contributed a corrupted vote (possibly by a collusion with some voting authority), that vote is simply discard.

Ballot Secrecy. With the use of linkable ring signature, no one can reveal the actual identity of the signer. Voter contributes to the randomness introduced in the encryption of choice, which means even the collusion of all voting authorities cannot reveal a particular ballot.

However, we remark that under a strong attack which compromises both the tallying center and a voting authority can reveal the voter-choice relationship.

Fairness. Every vote and any partial tally, is encrypted using the public key of the tallying center. No voter is able to learn about the outcome of the election before the final result is published.

Double Voting Detection. Any double voting can be detected using the linkability tags of the linkable ring signatures. Anonymity would not be affected if the same key is used in multiple elections since the linking can only be done with respect to the same voting event. The signatures from the same signer for different events remain unlinkable. Finally, even if the voter fails to include some of the legitimate voters in the signers group of the signature for whatever reason, double voting detection is still possible since linking is based on the event identifier but not the signers group.

4.6. Efficiency Analysis

Each vote on the bulletin board contains the following signatures and the corresponding messages: one normal signature signing a constant-size ciphertext, and one constant-size linkable ring signature signing a constant-size ciphertext, The size of the whole bulletin board is thus linear in the number of voters $|L|$. Note that the $O(|L| \times N_V)$ numbers of escrowed linkable ring signatures are not necessary to be posted. They are just used to locate corrupted votes.

5. Concluding Remarks

We apply escrowed linkable ring signature [17] in e-voting system to get the robustness and the receipt-freeness simultaneously. Our application is fundamentally different from the previously suggested use of linkable ring signature in simple e-voting systems, and brings a new tool to the set of cryptographic techniques enabling e-voting systems.

We use an identity-based solution [17] such that the public keys for those who are not interested in voting are implicitly defined, and thus other voters are free to vote even if there exist some legitimate voters who do not bother to get a key at all. It may be interesting to see if other identity-based primitives with privacy concerns can help better e-voting systems. For examples, one may study if the blind key extraction protocol in [29] can be used to issue an anonymous credential for e-voting, and the relationship between

the receiver-anonymity of some identity-based encryption schemes [1] and the realization of anonymous channels.

Our work aims to achieve receipt-freeness but not a stronger notion of coercion-resistance [37]. In particular, it may be interesting to devise some "fake-key" generation algorithm corresponding to some identity-based schemes to fight against simulation attacks, in which an attacker coerces a voter to divulge the private key. It is also interesting to study if the techniques here and those from [37] can be combined to achieve a nice set of security properties simultaneously.

Finally, we acknowledge that many issues still need to be addressed; for examples, development issues such as secure implementation and testing; and even more deployment issues like legislation policy, education and training, voter-eligibility checks, physical security, network and computer security issues like maintaining backup and availability. We hope to offer a comprehensive solution of using ring signatures in e-voting, and alternatives choices of cryptosystems that may be useful for a real-world full-blown system.

References

[1] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. *Journal of Cryptology*, 2008. To appear.

[2] M. Abe. Universally verifiable mix-net with verification work independent of the number of mix-servers. In *EUROCRYPT 98*, pages 437–447. Springer-Verlag, 1998. LNCS 1403.

[3] M. Abe. Mix-networks in permutation networks. In *ASIACRYPT 99*, pages 258–273. Springer-Verlag, 1999. LNCS 1716.

[4] R. Aditya. *Secure Electronic Voting with Flexible Ballot Structure*. PhD thesis, Information Security Institute, Queensland University of Technology, Nov. 2005.

[5] M. Au, S. S. M. Chow, W. Susilo, and P. Tsang. Short linkable ring signatures revisited. In *EuroPKI 2006*, pages 101–115. Springer-Verlag, 2006. LNCS 4043.

[6] M. Au, J. K. Liu, W. Susilo, and T. H. Yuen. Constant-size ID-based linkable and revocable-iff-linked ring signature. In *INDOCRYPT 2006*, volume 4329 of *LNCS*, pages 364–378. Springer-Verlag, 2006.

[7] M. Au, J. K. Liu, W. Susilo, and T. H. Yuen. Certificate based (linkable) ring signature. In *ISPEC 2007*, volume 4464 of *LNCS*, pages 79–92. Springer-Verlag, 2007.

[8] O. Baudron, P. A. Fouque, D. Pointcheval, G. Poupard, and J. Stern. Practical multi-candidate election system. In *PODC 2001*, pages 274–283. ACM Press, 2001.

[9] J. Benaloh. *Verifiable secret-ballot elections*. PhD thesis, Department of Computer Science, New Haven, Yale University, Sept. 1987.

[10] J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections. In 26*th ACM Symp. on Theory of Computing (STOC)*, pages 544–553. ACM, 1994.

[11] A. Bender, J. Katz, and R. Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. In *TCC '06*, volume 3876 of *LNCS*, pages 60–79. Springer, 2006.

[12] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, Feb. 1981.

[13] D. Chaum. Elections with unconditionally secret ballots and disruption equivalent to breaking RSA. In *EUROCRYPT 88*, pages 177–182. Springer-Verlag, 1988. LNCS 330.

[14] B. Chevallier-Mames, P. Fouque, D. Pointcheval, J. Stern, and J. Traore. On Some Incompatible Properties of Voting Schemes. In *IAVoSS Workshop On Trustworthy Elections, WOTE '06*, 2006.

[15] S. S. M. Chow, J. K. Liu, V. K. Wei, and T. H. Yuen. Ring signatures without random oracles. In *ASIACCS '06*, pages 297–302. ACM Press, 2006.

[16] S. S. M. Chow, R. W. C. Lui, L. C. K. Hui, and S. M. Yiu. Identity based ring signature: Why, how and what next. In *EuroPKI 2005*, pages 144–161, 2005.

[17] S. S. M. Chow, W. Susilo, and T. Yuen. Escrowed linkability of ring signatures and its applications. In *VietCrypt 2006*, pages 175–192. Springer-Verlag, 2006. LNCS 4341.

[18] J. Cohen and M. Fischer. A robust and verifiable cryptographically secure election scheme. In 26^{th} *Annual Symposium on Foundations of Computer Science*, pages 372–382. IEEE Computer Science Press, 85.

[19] R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO 94*, pages 174–187. Springer-Verlag, 1994. LNCS 839.

[20] R. Cramer, M. Franklin, B. Schoenmakers, and M. Yung. Multi-authority secret ballot elections with linear work. In *EUROCRYPT 96*, pages 72–83. Springer-Verlag, 1996. LNCS 1070.

[21] R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election schemes. In *EUROCRYPT 97*, pages 103–118. Springer-Verlag, 1997. LNCS 1233.

[22] I. Dåmgard and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In *Public Key Cryptography 2001*, pages 119–136. Springer-Verlag, 2001. LNCS 1992.

[23] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous identification in ad-hoc groups. In *EUROCRYPT 2004*. Springer-Verlag, 2004. LNCS 3027.

[24] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IT-31(4):496–472, July 1985.

[25] A. Fiat and A. Shamir. How to prove yourself: Practical solution to identification and signature problems. In *CRYPTO 86*, pages 186–194. Springer-Verlag, 1987. LNCS 263.

[26] A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale election. In *AUSCRYPT 92*, pages 244–251. Springer-Verlag, 1992. LNCS 718.

[27] J. Furukawa and K. Sako. An efficient scheme for proving a shuffle. In *CRYPTO 2001*, pages 368–387. Springer-Verlag, 2001. LNCS 2139.

[28] P. Golle, S. Zhong, D. Boneh, and M. Jakobsson. Optimistic mixing for exit-polls. In *ASIACRYPT 2002*, pages 451–465. Springer-Verlag, 2002. LNCS 2501.

[29] M. Green and S. Hohenberger. Blind Identity-Based Encryption and Simulatable Oblivious Transfer. In *ASIACRYPT 2007*, pages 265–282, 2007.

[30] J. Groth. A verifiable secret shuffle of homomorphic encryptions. In *PKC 2003*, pages 145–160. Springer-Verlag, 2003. LNCS 2567.

[31] J. Groth. Efficient maximal privacy in boardroom voting and anonymous broadcast. In *Financial Cryptography 2004*, pages 90–104. Springer-Verlag, 2004. LNCS 3110.

[32] M. Hirt. *Multi-party computation: Efficient protocols, general adversaries, and voting*. PhD thesis, ETH Zurich, 2001.

[33] M. Hirt and K. Sako. Efficient receipt-free voting based on homomorphic encryption. In *EUROCRYPT 2000*, pages 539–556. Springer-Verlag, 2000. LNCS 1807.

[34] M. Jakobsson. Flash mixing. In *PODC 98*, pages 83–89. ACM, 1998.

[35] M. Jakobsson. A practical mix. In *EUROCRYPT 98*, pages 449–461. Springer-Verlag, 1998. LNCS 1403.

[36] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated-verifier proofs and their applications. In *EUROCRYPT 96*, pages 143–154. Springer-Verlag, 1996. LNCS 1070.

[37] A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant electronic elections. In *WPES 2005*, pages 61–70. ACM Press, 2005.

[38] A. Kiayias and M. Yung. Self-tallying elections and perfect ballot secrecy. In *PKC 02*, pages 141–158. Springer-Verlag, 2002. LNCS 2274.

[39] A. Kiayias and M. Yung. The vector-ballot e-voting approach. In *Financial Cryptography 04*, pages 72–89. Springer-Verlag, 2004. LNCS 3110.

[40] B. Lee, C. Boyd, E. Dawson, and K. Kim. Providing receipt-freeness in mixnet-based voting protocols. In *Proc. ICISC 2003*, pages 245–258. Springer-Verlag, 2004. LNCS 2971.

[41] B. Lee and K. Kim. Receipt-free electronic voting scheme with a tamper-resistant randomizer. In *Proc. ICISC 2002*, pages 389–406. Springer-Verlag, 2003. LNCS 2587.

[42] P. Lee and C. Kim. Receipt-free electronic voting through collaboration of voter and honest verifier. In *JWISC 2000*, pages 101–108, 2000.

[43] H. Lipmaa, G. Wang, and F. Bao. Designated verifier signature schemes: Attack, new security notions and a new construction. In *ICALP 2005*, pages 459–471. Springer-Verlag, 2005. LNCS 3580.

[44] J. Liu, V. Wei, and D. Wong. Linkable spontaneous anonymous group signature for ad hoc groups. In *ACISP 2004*, pages 325–335. Springer-Verlag, 2004. Lecture Notes in Computer Science No. 3108.

[45] J. Liu and D. Wong. A restricted multi-show credential system and its application on e-voting. In *ISPEC 2005*, pages 268–279. Springer-Verlag, 2005. LNCS 3439.

[46] S. Micali, K. Ohta, and L. Reyzin. Accountable-subgroup multisignatures: extended abstract. In *CCS 2001*, pages 245–254, New York, NY, USA, 2001. ACM Press.

[47] A. Neff. A verifiable secret shuffle and its application to e-voting. In *CCS 2001*, pages 116–125. ACM, 2001.

[48] L. Nguyen. Accumulators from bilinear pairings and applications. In *CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 275–292, 2005.

[49] M. Ohkubo, F. Miura, M. Abe, A. Fujioka, and T. Okamoto. An improvement on a practical secret voting scheme. In *Information Security 99*, pages 225–234. Springer-Verlag, 1999. LNCS 1729.

[50] T. Okamoto. Receipt-free electronic voting schemes for large scale elections. In *Workshop on Security Protocols 97*, pages 25–35. Springer-Verlag, 1997. LNCS 1361.

[51] C. Park, K. Itoh, and K. Kurosawa. Efficient anonymous channel and all/nothing election scheme. In *EUROCRYPT 93*, pages 248–259. Springer-Verlag, 1994. LNCS 765.

[52] T. Pedersen. A threshold cryptosystem without a trusted party. In *EUROCRYPT 91*, pages 522–526. Springer-Verlag, 1991. LNCS 547.

[53] K. Peng, R. Aditya, C. Boyd, E. Dawson, and B. Lee. Multiplicative homomorphic e-voting. In *IndoCrypt 2004*, pages 61–72. Springer-Verlag, 2004. LNCS 3348.

[54] K. Peng, C. Boyd, and E. Dawson. Simple and Efficient Shuffling with Provable Correctness and ZK Privacy. In *CRYPTO 2005*, pages 188–204, 2005.

[55] R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT 2001*, pages 552–565. Springer-Verlag, 2001. LNCS 2248.

[56] K. Sako and J. Kilian. Secure voting using partial compatible homomorphisms. In *CRYPTO 94*, pages 411–424. Springer-Verlag, 1994. LNCS 839.

[57] K. Sako and J. Kilian. Receipt-free mix-type voting scheme - a practical solution to the implementation of a voting booth. In *EUROCRYPT 95*, pages 393–403. Springer-Verlag, 1995. LNCS 921.

[58] A. Shamir. How to share a secret. In *Communications of the ACM*, volume 22(2), pages 612–613. ACM Press, 1979.

[59] V. Shoup. *A Computational Introduction to Number Theory and Algebra*. Cambridge University Press, 2005.

[60] P. Tsang, V. Wei, T. Chan, M. Au, J. Liu, and D. Wong. Separable linkable threshold ring signatures. In *IndoCrypt 2004*, pages 384–398. Springer-Verlag, 2004. LNCS 3384.

A. Proof-of-Knowledge Protocols

A.1. (Witness-Indistinguishable) Proof

Let $\mathbb{G}$ be a cyclic group of prime order p, g_1 and g_2 be two generators of $\mathbb{G}$. Below gives a protocol for proving the equality of the discrete logarithm of two pairs of number, which refers to the relation R such that $(x, (g_1, g_2, u_1, u_2)) \in R \iff x = log_{g_1} u_1 = log_{g_2} u_2$.

1. $\mathcal{P}$ chooses $r \in \mathbb{Z}_p$ at random and sends $a = (t_1 = g_1^{r_1}, t_2 = g_2^{r_2})$ to $\mathcal{V}$.

2. $\mathcal{V}$ selects $c \in \mathbb{Z}_p$ and returns it to $\mathcal{P}$ as a random challenge.

3. $\mathcal{P}$ responses to $\mathcal{V}$ with $z = r + cx$ with private input $x = log_{g_1} u_1 = log_{g_2} u_2$.

4. $\mathcal{V}$ accepts the proof iff $g_1^z = t_1 u_1^c$ and $g_2^z = t_2 u_2^c$.

Now $\mathcal{P}$ wants to prove that he knows the witness of two purported relations, but does not want to reveal which. Suppose Σ_1 is the protocol for proving the first relation and Σ_2 is the second one, and let (a_i, c_i, z_i) be the transcript of Σ_i for $i \in \{1, 2\}$. Without loss of generality, assume $\mathcal{P}$ knows the witness for Σ_1. The protocol proceeds as follows.

1. $\mathcal{P}$ uses the simulator S for Σ_2 to generate a transcript (a_2, c_2, s_2).
2. $\mathcal{P}$ sends $(a_1 = g_1^{r_1}, a_2)$ to $\mathcal{V}$.
3. $\mathcal{V}$ selects $c \in \mathbb{Z}_p$ and returns it to $\mathcal{P}$ as a random challenge.
4. $\mathcal{P}$ computes $c_1 = c \oplus c_2$.
5. $\mathcal{P}$ generates (c_1, z_1) with private input x, and sends (c_1, c_2, z_1, z_2) to $\mathcal{V}$.
6. $\mathcal{V}$ accepts the proof iff both $(a_1, c \oplus c_2, z_1)$ and (a_2, c_2, z_2) are valid transcripts.

By using 1-out-of-n secret sharing (e.g. [58]) instead of XOR operator, the above protocol can be generalized to an 1-out-of-n witness indistinguishable proof [19].

A.2. (Non-Interactive) Designated-Verifier Proof

We use the modification from Lipmaa *et al*. [43], which fixed the original scheme by Jakobsson *et al*. [36].

Suppose the prover wants to prove the above relation R to only the verifier but no one else, who has a private key $x_B \in \mathbb{Z}_p$ and public key $y_B = g_1^{x_B}$. Let $H : \{0,1\} \to \mathbb{Z}_p$ be a hash function. A non-interactive designed verifier proof is as follow:

Proving: Randomly selects $w, r, t \in \mathbb{Z}_p$, calculates

$$\begin{aligned} c &= g_1^w y_B^r, \quad G = g_1^t, \quad M = g_2^t \\ h &= H(c, G, M, u_1, u_2, y_B) \\ d &= t + x_A(h + w) \bmod p \end{aligned}$$

and sends (w, r, G, M, d) to the verifier.

Verification: A designed verifier can verify a proof by calculating $c = g_1^w y_B^r$, $h = H(c, G, M, u_1, u_2, y_B)$ and verifying that $G u_1^{h+w} = g_1^d$ and $M u_2^{h+w} = g_2^d$.

Transcript simulation: A designated verifier simulates by randomly picking $d, \alpha, \beta \in \mathbb{Z}_p$ and computing

$$\begin{aligned} c &= g_1^\alpha, \quad G = g_1^d u_1^{-\beta}, \quad M = g_2^d u_2^\beta \\ h &= H(c, G, M, u_1, u_2, y_B) \\ w &= \beta - h \bmod p, \quad r = (\alpha - w) x_B^{-1} \bmod p \end{aligned}$$

B. Homomorphic Encryption

B.1. ElGamal Encryption [24]

Let $\mathbb{G}$ be a multiplicative group of prime order q with generator g. The private key x_T is chosen at random from $\mathbb{Z}_q$ and the corresponding public key is $y_T = g^{x_T}$. Given a message $m \in G$, the encryption of m is given by $(a, b) = (g^r, m \cdot {y_T}^r)$ for a randomly chosen $r \in \mathbb{Z}_q$. To decrypt the ciphertext (a, b), compute the plaintext $m = b \cdot a^{-x_T}$ using private key x_T.

Homomorphic Properties: For ciphertexts $(a_1, b_1) = (g^{r_1}, {y_T}^{r_1} m_1)$ and $(a_2, b_2) = (g^{r_2}, {y_T}^{r_2} m_2)$, anyone can easily obtain an encryption of $m_1 \cdot m_2$ by $(a_1 \cdot a_2 = g^{r_1 + r_2}, b_1 \cdot b_2 = {y_T}^{r_1 + r_2}(m_1 \cdot m_2))$.

Threshold Version: Suppose the key generation protocol of Pedersen [52] is used to share x_T among those N_T servers. After the protocol is carried out successfully, each tallying authority T_i ($1 \le i \le N_T$) will get a share $x_i \in \mathbb{Z}_q$ of the secret x_T, and has a commitment of its share computed as $h_i = g^{x_i}$ which is broadcast to other servers. The (group) public key is $y_T = g^{x_T}$. The secret x_T can be computed from any set Λ of size t as below:

$$x_T = \sum_{j \in \Lambda} x_j \lambda_j, \qquad \lambda_j = \prod_{l \in \Lambda \setminus \{j\}} \frac{l}{l - j} \tag{1}$$

This is a Shamir (t, N_T)-threshold secret sharing [58], any set of less than t servers cannot recover the secret x_T. To decrypt a ciphertext $(a, b) = (g^r, {y_T}^r m)$, they do not need to remove the private key x_T first. Instead, the following steps are carried out by any t servers:

1. Each server T_i broadcasts $w_i = a^{x_i}$ to other $t - 1$ servers and proves its correctness through a proof of knowledge of equality of discrete logs: $(\log_g h_i = \log_a w_i)$ [21].
2. From equation (1), the message m can be recovered by $m = b / \prod_{j \in \Lambda} w_j^{\lambda_j}$.

B.2. Baby/Giant Step Algorithm [59, page 271]

Let v and γ be two elements of $\mathbb{G}$. We want to compute the discrete logarithm of v to the base of γ in $\mathbb{G}$, i.e. the smallest positive integer $\mathcal{R}$ such that $\gamma^{\mathcal{R}} = v$. This can be done by Shank's baby step/giant step algorithm. Let $u = \lceil \sqrt{\ell^n} \rceil$. Compute $1, \gamma, \ldots, \gamma^{u-1}$, and set $\gamma_1 = \gamma^{-u}$. Write $\mathcal{R}$ in the form $\mathcal{R} = au + r$ with $0 \le r < u$, by the choice of u we must also have $a \le u$. For $a = 1, \ldots, u$, we compute $v \cdot \gamma_1^a$ and check whether it is in the list of $(1, \gamma, \ldots, \gamma^{u-1})$. If it is, we have $\gamma^{au+r} = v$ and $\mathcal{R} = au + r$. This takes $O(\sqrt{\ell^n})$ group operations and is better than brute-force attacks.

C. Escrowed Linkable Ring Signature

Now we review the construction of an ID-based linkable ring signature scheme by Chow *et al*. [17]. Their scheme uses the pairing accumulator in [48] to accumulate the public keys into the ring and produces a witness proving that the signer's public key is included in the accumulator. Signatures are linked based on "event identity", e.g. "Best Singer on 09/12/2007"; so that the signatures by the same signer for voting in a different day or other events held in the same day cannot be linked.

Setup: This algorithm is executed by the trusted key generation center (KGC). Let $(\mathbb{G}_1, \mathbb{G}_2)$ be bilinear groups of the same prime order p. Select a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. Let g_1 be a generator of $\mathbb{G}_1$, g_2 be a generator of $\mathbb{G}_2$ and $\psi(g_2) = g_1$. Randomly pick $s, u \in \mathbb{Z}_p^*$ and compute $g_2^s, g_2^{s^2}, \ldots, g_2^{s^q}$, where q is the maximum number of members in a ring signature. The auxiliary information s can be safely deleted. Randomly pick $g_3, g_4 \in \mathbb{G}_1$. Set hash function $H : \mathbb{G}_1^3 \times \mathbb{G}_2 \times \mathbb{G}_T \times \mathbb{G}_1^2 \times \mathbb{G}_T^3 \times \{0,1\}^* \rightarrow \mathbb{Z}_p$ and $H_0 : (\{0,1\}^*)^2 \rightarrow \mathbb{G}_2$. Identities of users are in the group $\mathbb{Z}_p$. The public parameters param are $(\hat{e}, \psi, g_1, g_2, g_2^s, \ldots, g_2^{s^q}, g_3, g_4, u, H, H_0)$. For efficiency reason, $\rho = \hat{e}(g_1, g_2)$ and $\rho' = \hat{e}(g_1, g_2^s)$ can be included in the public parameters.

KGCGen: The KGC randomly picks $x, x' \in \mathbb{Z}_p^*$ as the master key, the corresponding public key is $(y = g_2^x, y' = g_2^{x'})$. For efficiency reason, $\omega = \hat{e}(g_1, y)$ can be pre-computed.

UGen: On input an identity id, the KGC computes the private key $S_{\mathsf{id}} = g_1^{1/(\mathsf{id}+x)}$. The user can verify the private key by checking if $\hat{e}(S_{\mathsf{id}}, g_2^{\mathsf{id}} y) = \hat{e}(g_1, g_2)$.

LAGen: On input of the identity of the linking authority (LA) ℓa, the KGC computes the private key $S_{\ell a} = H_1(\ell a)^{x'} \in \mathbb{G}_1$.

ELRingSign: The user with identity id_1 with private key S_{id_1} who wants to sign a ring signature for message M with users $\mathsf{id}_2, \ldots, \mathsf{id}_N$ firstly computes $W = g_1^{u(\mathsf{id}_2+s)\cdots(\mathsf{id}_N+s)}$, $V = g_1^{u(\mathsf{id}_1+s)(\mathsf{id}_2+s)\cdots(\mathsf{id}_N+s)}$ (W and V can be computed efficiently like the way described in the pairing accumulator [48]). Let $h = H_0(\mathsf{param}, eID)$ where eID is the event identity. s/he then computes the signature as a proof of knowledge of:

$$\begin{aligned}\{(ID_1, S_{ID_1}, W, d) : (\hat{e}(V, g_2) = \hat{e}(W, g_2^{ID_1+s}) \\ \wedge\ \hat{e}(S_{ID_1}, g_2^{ID_1} y) = \hat{e}(g_1, g_2) \\ \wedge\ \tilde{y} = \hat{e}(S_{ID_1}, h)\hat{e}(H_1(\ell a), y')^d \ \wedge\ U = g_2^d\}\end{aligned}$$

We now explain the above notation. (ID_1, S_{ID_1}, W, d) are the secret witness. The first equality $\hat{e}(V, g_2) = \hat{e}(W, g_2^{ID_1+s})$ refers that the signing key of ID_1 is accumulated in the list of identities V. The validity of the private key of ID_1 is ensured by $\hat{e}(S_{ID_1}, g_2^{ID_1} y) = \hat{e}(g_1, g_2)$; while the last two equalities ensure that the linkability tag $\hat{e}(S_{ID_1}, h)$ is encrypted such that the linking authority ℓa can decrypt and perform the linkability check later for the event identified by h. Detailed steps are as follows:

1. Randomly pick $t_1, t_2, t_3, d \in \mathbb{Z}_p^*$ and compute:

$$\begin{aligned} T_1 &= S_{\mathsf{id}_1} g_1^{t_1}, \quad T_2 = W g_1^{t_2}, \quad T_3 = g_3^{t_1} g_4^{t_2} g_1^{t_3}, \\ \tilde{y} &= \hat{e}(S_{\mathsf{id}_1}, h)\hat{e}(H_1(\ell a), y')^d, \quad U = g_2^d \end{aligned}$$

2. Randomly pick $r_1, r_2, \ldots, r_8 \in \mathbb{Z}_p^*$ and compute:

$$\begin{aligned} R_1 &= g_3^{r_2} g_4^{r_4} g_1^{r_6}, \quad R_2 = g_3^{r_3} g_4^{r_5} g_1^{r_7} T_3^{-r_1}, \\ R_3 &= \hat{e}(T_1, g_2)^{r_1} \hat{e}(g_1, g_2)^{-r_3} \omega^{-r_2}, \\ R_4 &= \hat{e}(T_2, g_2)^{r_1} \hat{e}(g_1, g_2)^{-r_5} \hat{e}(g_1, g_2^s)^{-r_4}, \\ R_5 &= \hat{e}(H_1(\ell a), y')^{r_8} \hat{e}(g_1, h)^{-r_2}, \quad R_6 = g_2^{r_8} \end{aligned}$$

3. Compute $c = H(T_1, T_2, T_3, h, \tilde{y}, U, R_1, \ldots, R_6, M)$

4. Compute $s_1 = r_1 + c\mathsf{id}_1, s_2 = r_2 + ct_1, s_3 = r_3 + ct_1\mathsf{id}_1, s_4 = r_4 + ct_2, s_5 = r_5 + ct_2\mathsf{id}_1, s_6 = r_6 + ct_3, s_7 = r_7 + ct_3\mathsf{id}_1, s_8 = r_8 + cd$.

5. Output $\sigma = (T_1, T_2, T_3, e, \tilde{y}, c, s_1, \ldots, s_8, U)$ as the signature, with the group public key V or the set of identity $\{\mathsf{id}_1, \mathsf{id}_2, \ldots, \mathsf{id}_N\}$.

ELRingVerify: Given a signature σ, the group public key V and a message M, parse σ as $(T_1, T_2, T_3, e, \tilde{y}, c, s_1, \ldots, s_8, U)$, the verification checks:

1. Compute $h = H_0(\mathsf{param}, e)$ and:

$$\begin{aligned} R_1 &= g_3^{s_2} g_4^{s_4} g_1^{s_6} T_3^{-c}, \quad R_2 = g_3^{s_3} g_4^{s_5} g_1^{s_7} T_3^{-s_1}, \\ R_3 &= \hat{e}(T_1, g_2)^{s_1} \rho^{-s_3} \omega^{-s_2} \big(\hat{e}(T_1, y)/\hat{e}(g_1, g_2)\big)^c \\ R_4 &= \hat{e}(T_2, g_2)^{s_1} \rho^{-s_5} \rho'^{-s_4} (\hat{e}(T_2, g_2^s)/\hat{e}(V, g_2))^c, \\ R_5 &= \hat{e}(H_1(\ell a), y')^{s_8} \hat{e}(g_1, h)^{-s_2} (\hat{e}(T_1, h)/\tilde{y})^c, \\ R_6 &= g_2^{s_8} U^{-c} \end{aligned}$$

2. Accept if $c = H(T_1, T_2, T_3, h, \tilde{y}, U, R_1, \ldots, R_6, M)$.

Link: On input signatures σ_b for $b \in \{0,1\}$, output $\perp$ if they do not pass ELRingVerify. Else compute $y_b = \tilde{y}/\hat{e}(S_{\ell a}, U_b)$. Output 1 if $y_0 = y_1$ and they correspond to the same event identifier, 0 otherwise. Note that the correctness of $\hat{e}(S_{\ell a}, U_b)$ can be easily proven, the linking authority can thus convince any other parties about the linkage between the signatures.