A Novel Approach to e-Voting with Group Identity Based Identification and Homomorphic Encryption

Apurva K Vangujar, Buvana Ganesh and Paolo Palmieri

School of Computer Science & IT,
University College Cork, Ireland
a.vangujar@cs.ucc.ie, b.ganesh@cs.ucc.ie, p.palmieri@cs.ucc.ie *

Abstract. Electronic voting (e-voting) aims to provide a sustainable and accessible environment for voters while preserving anonymity and trust. In this paper, we present a novel e-voting scheme that combines Group Identity-based Identification (GIBI) scheme with Homomorphic Encryption (HE) based on the Distributed ElGamal (DE) cryptosystem. Our scheme allows for efficient voter authentication through the use of a Discrete Logarithm (DL)-based identification protocol and enables encrypted vote counting without the need for decryption. Additionally, our scheme allows for individual and universal verifiability through the use of Zero-Knowledge (ZK) proofs. We also propose some future work to enhance the scheme for more secure or practical use.

Keywords: Identity-based Identification, Group IBI, Homomorphic Encryption, e-voting, ElGamal, Distributed ElGamal, Discrete Logarithmic

1 Introduction

Electronic voting, or e-voting, refers to the use of electronic systems to cast and count votes in an election. It is becoming increasingly popular as a way of to modernize the voting process and increasing accessibility for voters. However, the use of e-voting also raises concerns about the security and integrity of the voting process. The two main aspects of e-voting, that have to be rigorous in terms of security, are voter anonymity and trust on the process. Electronic voting systems, compare to traditional paper-based elections, promise that election results will be calculated quickly with less chance of human error and also will reduce costs in a long-term period. Ensuring the accuracy and security of e-voting systems is critical to maintaining trust in the electoral process of the e-voting scheme so that it can be implemented in practice. Therefore, there is a need for research on secure cryptographic electronic election schemes.

As of 2021, around 36 countries around the world have experimented with electronic elections of some sort, according to a report by the International Institute for Democracy and Electoral Assistance [1]. Most countries use direct recording electronics, optical mark recognition, electronic ballot printers or internet voting systems. The adoption

* This publication has emanated from research supported in part by a Grant from Science Foundation Ireland under Grant number 18/CRT/6222

of e-voting systems has been more widespread in some regions, such as Europe and Latin America, than in others. However, the use of e-voting is growing globally and is expected to continue to do so in the future.

To handle the security aspect, the identity and the ballot of the voter should be protected. In the literature, a combination of signature schemes [23] and encryption algorithms [28] are used to protect the process. But this is not enough in the early registration phases, as can be seen in attacks proposed on e-voting schemes used in practice like the sVote system in Switzerland used by SwissPost [20]. To tackle this, we propose a solution using Identity-based Identification (IBI) scheme and Homomorphic Encryption (HE).

First proposed by Adi Shamir in 1984, an Standard Identification (SI) scheme [26] is a method for identifying individuals using their unique identity, such as a name, date of birth, or other personal information. SI scheme enables a prover P to verify their identity to a verifier V without providing any personal information. Boneh and Franklin [6] pioneered the identity-based encryption scheme that led to the flourishing of identity-based cryptography. In later years, Bellare et al. [3] constructed a more secure IBI scheme based on the Zero-Knowledge (ZK) proof that results in higher efficiency. In IBI scheme, a Trusted Authority (TA), known as the Private Key Generator (PKG), generates a unique private key for each individual based on their identities. IBI schemes can be used in e-voting systems to authenticate the identity of voters and ensure that they are eligible to vote.

The ElGamal cryptosystem [18] is a public-key encryption scheme which possesses the property of homomorphism, that allows computation on encrypted data. Therefore, it can be used to encrypt ballots to not disclose the vote to any party. There have been many works with other cryptosystems or variants of ElGamal for e-voting schemes, which we discuss below in the Section 1.2. In conjunction with ZK protocols, the ElGamal cryptosystem is used to provide privacy and confidentiality while ensuring the integrity of the election results.

1.1 Motivation

Apart from ElGamal encryption algorithm, several other cryptographic algorithms such as Digital signatures (DS), hash functions, asymmetric cryptography, ZK proofs have been used in e-voting systems, and the algorithms selected normally depends on the requirements of the specific e-voting system.

In the relevant literature, we see the use of Group Signature (GS) schemes in e-voting systems to provide anonymity for voters. In such systems, each voter is issued a GS key that they can use to sign their ballot. There are several different types of GS schemes, and the specific scheme used in an e-voting system may depend on the specific requirements and constraints of the system. The initial GS scheme was proposed by Chaum et al. [12] and improved overtime in the universal design proposed by Boneh et al. [7] and Cocks [15]. The ability of a designated group manager or other authority to reveal the specific identity of a voter in a GS scheme can be a significant disadvantage, particularly in an e-voting system where there may be a large number of voters and ballots [24]. This could potentially undermine the integrity of the voting process and violate the privacy of the voter.

On the other hand, Group Identity-based Identification (GIBI) scheme [13] do not have this vulnerability, as they do not allow the specific identity of a voter to be revealed by any authority. This can provide a higher level of anonymity and privacy for voters, which may be important in certain e-voting systems. Additionally, GIBI schemes may be more efficient than GS schemes in terms of the amount of computation required to generate and verify the identity of a voter. Voters can validate their right to vote without revealing their identity or other personal information, which is one of the primary advantages of using a ZK protocol in GIBI scheme in e-voting. Using a ZK protocol can aid in the prevention of voter fraud by ensuring that only eligible voters can cast ballots. Therefore, we adopt GIBI and ZK-proofs for our proposed scheme.

The confidentiality aspect of e-voting schemes is handled by variants of the Distributed ElGamal (DE) scheme in literature. This scales to multiple parties, providing additional security for the key management process. Both ElGamal and DE are secure encryption schemes that use a pair of keys (a public key and a private key) to encrypt and decrypt messages and both schemes are homomorphic. However, DE can provide an additional layer of security by distributing the encryption process among multiple users under the same public key. This can make it more difficult for an attacker to compromise the system and can help to protect the user's identity and the vote's anonymity in the voting process. DE can be easier for voters and election officials to use, as the encryption process is handled automatically by the servers [27]. This can help to simplify the voting process and reduce the burden on voters and election officials.

There are several reasons why it might be beneficial to develop a novel e-voting scheme which is a combination of GIBI and HE:

- **Improved security**: IBI and HE can offer strong security guarantees, making them suitable for use in e-voting systems. Therefore an e-voting scheme with such algorithms could offer even stronger security compared to existing schemes that use signatures to verify the voter.
- **Ease of use**: IBI schemes can be easier for voters to use compared to traditional methods that require physical documents or tokens. HE allows for computations to be performed on encrypted data, which can simplify the voting process for voters and reduce the burden on election officials.
- **Confidentiality**: ZK protocols are used in e-voting scheme to ensure the privacy and confidentiality of voters' ballots. The use of ZK protocols can help to protect against vote buying, coercion, and other forms of voter interference, as well as to prevent the revelation of individual votes after the election.
- **Increased accessibility**: GIBI and HE can enable e-voting systems to be more accessible, particularly for voters who may have difficulty accessing traditional polling stations or who may have difficulty using traditional identification scheme. Any e-voting scheme that combines these algorithms could increase the accessibility of voting.

Compared to existing schemes, a combination of IBI and HE with the ZK protocol for e-voting can offer improved security, usability, and accessibility.

1.2 Related work

E-voting schemes, first developed by Chaum [9], have been the topic of an extensive amount of research. There are now three election models used for electronic voting: mixed nets with encryptions, signatures, and HE-based models. In the mix-net model proposed by Chaum [11], different linked servers refer to as mixes are used to randomize input messages and output a permutation of them so that the input and output messages cannot be linked. Several mix-net models have been proposed in the literature [21,8,2]. The signature-based model involves DS scheme to verify the identities of voters and the integrity of the voting procedure. The approaches of Cocks [15] and Boneh et al. [7] are examples of schemes that use DS for e-voting. Chaum [10] presented one of the earliest ideas for the use of Blind Signatures (BS) scheme in e-voting and BS allow voters to sign their ballots without exposing the content of their votes to the voting authority. The schemes proposed by Rivest et al. [25], Benaloh [5] are later approaches for adopting BS scheme in e-voting.

The HE-based model allow votes to be encrypted and counted without decryption in e-voting, Cramer et al. [17] uses ElGamal encryption combination with ZK proofs, to allow votes to be encrypted and counted without requiring the votes to be decrypted. Schemes based on the HE model have universal verifiability while protecting the privacy of voters. Yang et al. [28] proposed a verifiable e-voting scheme with several parties, such as the registry, voting, and tallying authorities, with the assumption that at least one of them is honest and the others are only partially honest. They use ZK proofs to demonstrate the verifiability of ballots, the final tally, etc. by utilizing partial knowledge or proof of knowledge, but do not provide the means to verify identity of the voter before they enter the process.

Identity-based Cryptosystems (IBC) [26] are mainly aimed to simplify the certificate management and public key revocation issues. IBC associates the user's identity with the public key, therefore the public key is obtained directly from the user's identity. The Identity-based Signature (IBS) of Choon-Cheon [14] has been introduced. Zhang et al. [29] presented the Identity-based Blind Signature (ID-BS) approach using the IBC concept. After reviewing all the existing ID-BS schemes for the e-voting system, which were not possible to implement. Chin et al. [13] give e-voting as an application of such schemes in their paper. Malina et al. [23] proposed a GS scheme based approach to e-voting with group managers and polling stations. Given that voters are already validated, numerous DS schemes can be employed to create e-voting systems. This is the disadvantage of e-voting systems that are voluntary and possess individual verifiability. By providing additional authentication, universal and individual verifiability, an IBI scheme streamlines and enhances the security of the voting process.

The DE assumption was introduced in [16] and used first in the work by Adida et al. [2] following the Helios framework for e-voting. Since then, DE has been utilized in many e-voting schemes in different forms including, the Swiss sVote system [20] where it helped in encrypting the votes as tuples and scalar multiplication rather than exponentiation. In addition, Haines et al. [19] use codes and ZK to demonstrate the validity of votes using partial codes and OR proofs. In their scheme, encryption and decryption keys are distributed to multiple parties, and votes are encrypted using the ElGamal en-

cryption algorithm. The encrypted votes are subsequently sent to a tallying authority, which can count them without decrypting them.

1.3 Contribution

In this paper, we present a new e-voting scheme that combines GIBI with HE based on the DL assumption. Our main contributions are:

- The design and construction of an e-voting scheme that uses GIBI and HE scheme based on the DE cryptosystem.
- The development of a DL-based identification protocol to authenticate valid voters and allow them to cast their votes homomorphically encrypted.
- The ability of our scheme to allow voters to verify their ballot submissions and the final tally, while still keeping the encrypted ballots and vote counts secure.
- Suggestions for future work in this area.

1.4 Organisation

The paper is structured as follows. Section 2 introduces the mathematical notations and primitives used in the rest of the paper. Section 3 discusses the definition and construction of the proposed novel e-voting scheme. Section 4 and Section 5, we discuss the security and the efficiency of this proposed scheme. Finally, Section 6 points out the future work in this area, followed by the conclusion.

2 Preliminaries

2.1 Notations

Let λ as the security parameter. For a prime q, consider the multiplicative group $\mathbb{Z}_q$ over $\{0, 1, ..., q\}$. Let the multiplicative cyclic group be $\mathbb{G}$ of order q, and g be its generator.

2.2 Discrete Logarithmic (DL) Assumption

The DL assumption is defined as in [4].

Definition 1. *Let $\mathbb{G}$ be the multiplicative cyclic group of order q and let g be the generator. When (g, g^x) is known, x can be determined. The definition of the DL assumption is $Y = g^x$, which returns output x. N is an algorithm that exists to solve (t_{DL}, ϵ_{DL}) DL assumption probabilistically when (t_{DL}, ϵ_{DL}) is given.*

$$\Pr[N(\mathbb{G}, q, g, Y) = x] \geqslant \epsilon_{DL} \tag{1}$$

2.3 Identity based Identification Scheme

Kurosawa Heng [22] provide methods to transform a DS scheme into SI and further into IBI. Such an IBI scheme has three probabilistic polynomial-time (PPT) algorithms, which are as follows:

1. **Key Setup:** It takes the security parameter 1^λ and generates the group $\mathbb{G}$ of prime order of q and generator $g \in \mathbb{G}$ and hashing function $H : \{0,1\}^* \rightarrow \mathbb{G}$. It chooses a random integer $x \in \mathbb{Z}_q$ and sets $y = g^x$. It outputs the public parameters $\mathsf{pp} = (\mathbb{G}, q, g, y, x, H)$ and calculates mpk and $\mathsf{msk} = x$.
2. **Extract:** It takes the identity ID and x as input and outputs the user secret key usk, $d = H(\mathrm{ID})^x$
3. **Identification Protocol:** The Prover P takes the input $(\mathsf{mpk}, \mathsf{usk}, \mathrm{ID})$ whereas verifier V takes input as $(\mathsf{mpk}, \mathrm{ID})$.
   - P begins by selecting a random integer $r \in \mathbb{Z}_q$, sets $X = g^r$ and sends X to V.
   - V selects a random challenge $c \in \mathbb{Z}_q$ and sends to P.
   - P calculate the response $Q = (r + c)H(\mathrm{ID})$ and sends to V.
   - V verifies if the tuple $(g, g^r, H(\mathrm{ID}), Q)$ is valid or not.

2.4 Group Identity-based Identification Scheme

Group Identity-Based Identification (GIBI) scheme by Chin et al. [13] is defined as transactions between the Trusted Authority TA, the Group Manager GM, different groups $(\mathsf{G_1}, \mathsf{G_2}, ..., \mathsf{G_n})$ and group members.

1. **Setup**: TA sets up public parameters and outputs the pair $(\mathsf{mpk}, \mathsf{msk})$.
2. **Extract**:
   Phase 1: Run by the TA, $(\mathsf{mpk}, \mathsf{msk})$ are taken as input and the group key pair $(\mathsf{gpk}, \mathsf{gsk})$ is generated and passed to respective GM.
   Phase 2: Run by the GM for each member of a group who possess an ID. To be part of that group, the ID is sent to be registered as a member to the GM. Taking ancestor input as $(\mathsf{gpk}, \mathsf{gsk})$, GM generates user keys for ID are $(\mathsf{upk}, \mathsf{usk})$.
   Phase 3: Run by the GM. GM stores $(\mathrm{ID}, \mathsf{upk})$ for that ID and do not store usk.
3. **Identification Protocol (IP)**:
   Phase 1: Assume a group member $\mathsf{G_i}$ wants to perform IP as a group. Taking $\mathsf{G_1}$ as an example, for input usk, $\mathsf{G_1}$ outputs a signature σ_1. $\mathsf{G_1}$ then asks $\mathsf{GM_1}$ for a request to verify, and sends $(\mathsf{upk}, \sigma_1, \mathrm{ID})$ to $\mathsf{GM_1}$.
   Phase 2: $\mathsf{GM_1}$ checks if σ_1 is valid and verifies if the associated ID is within the list of members. If $\mathsf{G_1}$ is a valid member in the list, $\mathsf{GM_1}$ issues a notice to all other members in the group to generate their signatures and attach their upk. As $\mathsf{GM_1}$ receives the values for each members in group, $\mathsf{GM_1}$ also checks if the provided values are valid or not.
   Phase 3: Once all values are valid GM then performs verification by verifier V, by attaching a signature generated from gsk, σ_g as a representation of the group verification. GM performs ZK with V.

2.5 Distributed ElGamal for Homomorphic Encryption

The ElGamal encryption scheme is a public-key encryption scheme that is based on the difficulty of computing DL assumption [18]. In order to make the ElGamal encryption scheme additively homomorphic (i.e. homomorphic with respect to addition), it is possible to encrypt the message g^m instead of just message m. This variant of the ElGamal encryption scheme is called *Exponential ElGamal*.

Definition 2. *Let the group* P *under* $+$ *be the plaintext space. Let* C *a group under* $\cdot$ *be the ciphertext space. An encryption* E *is homomorphic, if for given* $c_1 = \mathsf{E}(m_1)$ *and* $c_2 = \mathsf{E}(m_2)$*, then* $c_1 \cdot c_2 = \mathsf{E}(m_1 + m_2)$*, for any operation* $+$ *and* $\cdot$*.*

The distributed version of the exponential ElGamal cryptosystem for n voters consists of the following algorithms for each voter i where $1 \geq i \geq n$. Consider all operations under mod q

1. **KeyGen** $(\lambda, g, q) \rightarrow (\mathsf{pk}_i, \mathsf{sk}_i)$, where the secret key $\mathsf{sk}_i = x_i$ is sampled uniformly from $\mathbb{Z}_q$ and the public key is $\mathsf{pk}_i = y_i = g^{x_i}$. The DE requires a common public key used for encryption for all voters.

$$\mathsf{pk} = \prod_{i=1}^{n} y_i = g^{x_1 + \ldots + x_n}$$

2. **Encrypt** $(\mathsf{pk}, m) \rightarrow c$ where $c = (a_i, b_i)$. For user i, encrypting their message m, where for a random k_i:

$$a_i = g^{k_i}; \quad b_i = g^m \mathsf{pk}^{k_i} \tag{2}$$

3. **Decrypt** $(x_i, c) \rightarrow m = b_i / a_i^{x_i}$ for one message. For multiple votes, DE requires partial decryptions to be collected from all parties that are then homomorphically combined to calculate the final tally. During the tallying phase, the decryption procedure for a combined ciphertext (a, b) is as follows:
   - Every user computes a^{x_i} and broadcasts commitment of computed values $H(a^{x_i})$ so that anyone can check if each a^{x_i} matches with $H(a^{x_i})$;
   - Each user sends the partial share to the authority to decrypt the combined message $m = m_1 + ... + m_n$.

$$\frac{b}{\prod_{i=1}^{n} a^{x_i}} = \frac{b}{a^{x_1 + \ldots + x_n}} = g^m \tag{3}$$

   Finally, m can be revealed by computing DL assumption.

4. **Evaluation** $(c_1, ..., c_n)$**:** For ciphertexts c_i, the additive homomorphism for messages can be easily verified. For n messages m_1 to m_n encrypted under pk and the combined secret key $x = x_1 + ... + x_n$.

$$\mathsf{Dec}(c_1 ... c_n) = \frac{b_1 ... b_n}{a_1^x ... a_n^x} = g^m \tag{4}$$

3 Novel e-Voting Scheme

3.1 Requirements

Electronic elections should meet all the requirements as the paper-based ones, and our goal is to provide greater security than is possible with the conventional methods. Such requirements are listed in Table 1 for e-voting schemes and what we wish to achieve in the one we propose.

Requirement	Explanation
Eligibility	Only authorized individuals can vote.
Unreusability	Each eligible voter is limited to one vote. It is against the rules to vote by proxy.
Privacy	All votes remain confidential. Voters are anonymous.
Robustness	Nobody is allowed to disrupt the election. A vote cast cannot be altered. In the final tally, all legal votes are counted, while invalid votes are detected and deleted.
Fairness	During the voting process, no participant can obtain information about the partial tally, as such data could influence voters.
Uncoercibility	During the election, a coercer can only monitor all public information and all conversations between voters and authorities, but he is able to instruct the voter on how to conduct himself during the voting process and can even provide him with random bits.
Receipt-freeness	Prior to the election, an opponent may engage in vote-buying, i.e., bribe the voter in exchange for their vote. Receipt-free voting prevents vote-buying because there is no record of the vote cast.
Individual verifiability	Each eligible voter can confirm that their vote was cast as intended and included in the final total.
Universal verifiability	Any voter or spectator can verify that the election is fair and that the final tally is the precise sum of all legitimate ballots.

Table 1. Requirements for e-Voting schemes

3.2 Participants

- **Trusted Authority.** Trusted Authority TA is responsible for setting up the system, declaring the results.
- **Voters.** In an e-voting scheme, voters are individuals who are eligible to cast their votes electronically using a computer or other electronic device. The set of all voters is denoted by v. All voter arrangement in e-voting is given further.
- **Candidates.** There are set of candidates $\mathsf{C} = (\mathsf{C}_1, \mathsf{C}_2, ..., \mathsf{C}_l, ..., \mathsf{C}_k)$, nominated by a political party, an organization, or as an independent candidate. In an e-voting system, C are typically listed on an electronic ballot, which is presented to the voter during the voting process. The v can then select their preferred C by clicking on their name or otherwise indicating their choice.

– **Registry.** The Registry $\mathsf{R} = (\mathsf{R}_1, \mathsf{R}_2, ..., \mathsf{R}_i, ..., \mathsf{R}_m)$ is in charge of managing the authorization phase. Each registry will have Registry Manager RM in a such way that $\mathsf{RM} = (\mathsf{RM}_1, \mathsf{RM}_2, ..., \mathsf{RM}_i, ..., \mathsf{RM}_m)$. One registry will include RM and set of v. For example: For $\mathsf{R}_1 = (\mathsf{RM}_1, \mathsf{v}_{1,1}, \mathsf{v}_{1,2}...\mathsf{v}_{1,j}, .., \mathsf{v}_{1,n})$. It personally verifies v eligibility and supervises the generation of private and public keys for each v.
– **Voting Authorities.** Voting Authorities $\mathsf{VA} = (\mathsf{VA}_1, \mathsf{VA}_2, ..., \mathsf{VA}_i, ..., \mathsf{VA}_m)$. Number of R is equal in number as VA in e-voting system. The VA manages ZK voter proofs. VA allow only eligible v to vote for the election. VA verifies all the v at the same time as it uses ZK proofs under DL assumption.
– **Tallying Authorities.** The tallying authority Ta, assigned with the responsibility of collecting and tallying votes cast, may be a government agency, an electoral commission, or another sort of organization. It also passes results to respective VA to declare results. The number of Ta is equals to VA.
– **Bulletin Board.** Bulletin board BB is readable by the public. VA displays BB with the final results. Everyone can view the statistics of election, but nobody can alter their content.

3.3 Overview

The proposed e-voting process includes five major phases described as follows and shown in Fig. 1: The setup, registration, and authorizing phase in Fig. 2; voting and tallying phase is shown in Fig. 3.

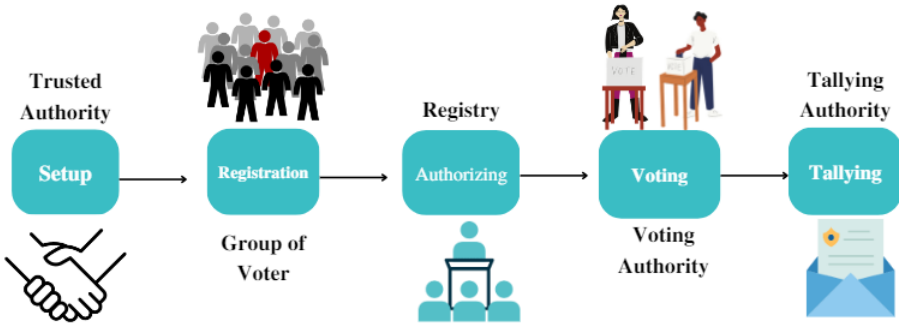

Fig. 1. Flow of Proposed e-Voting Scheme

1. *Setup*: In this phase, the trusted authority TA sets up the key for the group registry manager RM, followed by the registry setting up key for all voters v.
2. *Registration*: This may involve Registry R checking voter v eligibility, collecting and storing voter information, and issuing voter identification numbers that will be used to authenticate voters with Voting Authority VA.

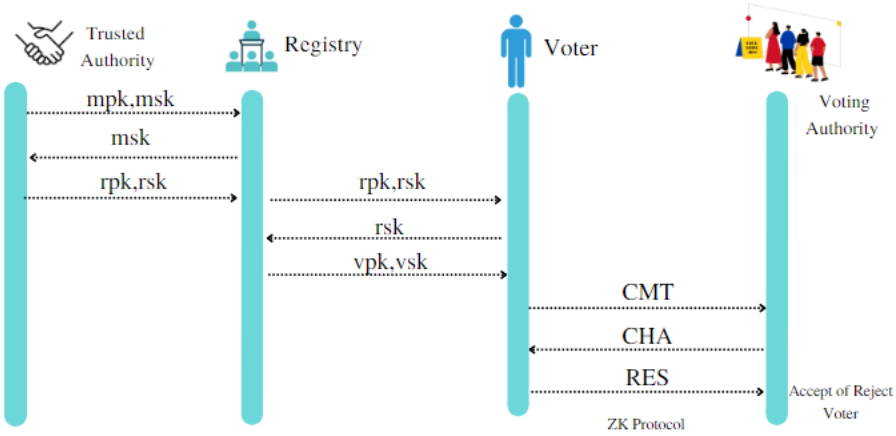

Fig. 2. Setup, Registration, and Authorizing Phase of Proposed e-voting Scheme

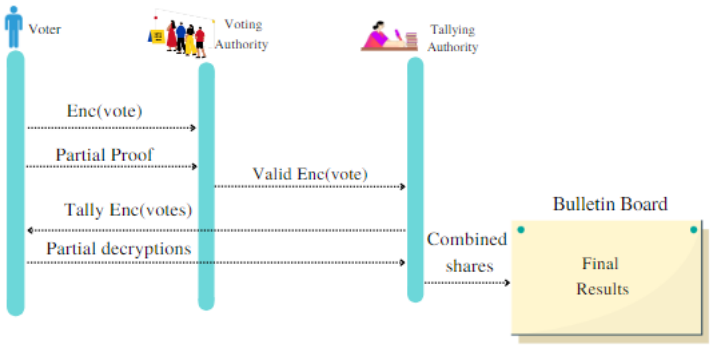

Fig. 3. Voting and Tallying Phase of Proposed e-Voting Scheme

3. *Authorization*: This phase covers the verification of voter v eligibility and the granting of access to the voting system. This may involve voter v authentication and authorization processes.
4. *Voting*: This phase comprises of the actual casting of votes. This may involve selecting candidates C, submitting votes, and validating the vote's legitimacy. All votes are to be encrypted in this phase.
5. *Tallying*: This phase involves the tallying authority Ta counting and aggregation all the votes to determine the election's outcome. This involves decrypting the votes, confirming the accuracy of the vote count, and declaring the decrypted results by VA and listing on the Bulletin Board BB.

3.4 Definition

Proposed Group Identity-based Identification and Homomorphic Encryption (GIBI-HE) scheme protocol consists of five PPT algorithms, namely, Key Setup, Extract, Iden-

tification Protocol, Encrypt and Validate, Tally, and Decrypt, run among five entities, namely, Trusted Authority, Voter, Registry, Voting Authority, and Tallying Authority.

- **Key Setup**: Using the random number generator and secret key chosen by the TA, pair of a master public key mpk and master secret key msk is generated as output.
- **Extract**: Considering the generic scenario for e-voting scheme.
  1. *Phase 1*: For RM_i, it calculates registry public and secret key $(\mathsf{rpk}_i, \mathsf{rsk}_i)$ using ancestor msk.
  2. *Phase 2*: For any $\mathsf{v}_{i,j}$, it calculates voter public and secret key $(\mathsf{vpk}_{i,j}, \mathsf{vsk}_{i,j})$ using ancestor rsk_i.
  3. *Phase 3*: Using the key of the voter for encryption, the common public key cpk is generated by multiplying all the voter's public keys. This is then sent back to all voters for encryption.
- **Identification Protocol**: Following are the three stages of communication between the voter v (acting as prover) and the VA (acting as verifier):
  1. *CMT*: $\mathsf{v}_{i,j}$ chooses random integer to calculate value and send to VA_i.
  2. *CHA*: VA_i generates the random challenge and forwards to $\mathsf{v}_{i,j}$.
  3. *RES*: $\mathsf{v}_{i,j}$ accepts the challenge and generates response based on the challenge. VA_i accepts $\mathsf{v}_{i,j}$ for voting process if and only if, it verifies the final equation using DL-tuple.
- **Encrypt**: After the authorization using ZK, the voter casts the vote and encrypts it using cpk. For C_k candidates and voter $\mathsf{v}_{i,j}$, the vote has the form $ev_{i,j} = (Enc(b_1),$ ..., $Enc(b_k))$, where $b_l = 0$ and 1 depending on which candidate the ballot was cast for.
- **Validate**: After encryption, the VA_i will determine if the vote is genuine or invalid by adding all components of $ev_{i,j}$ to determine if it equal 1 using ZK. For an instance with 3 candidates, (1,0,0) is a legal vote, however (1,0,1) is invalid since the $\mathsf{v}_{i,j}$ has cast two votes rather than simply one.
- **Tally**: The VA_i and Ta_i collect the encrypted votes from the voters and tally them using homomorphic additive property.
- **Decrypt**: To decrypt the vote, VA_i collects partial shares from all voters to compute the total combined tally.

3.5 Construction of GIBI-HE e-voting Scheme

The GIBI-HE scheme is a new e-voting scheme that combines the use of GIBI and DE to provide a secure and efficient voting system. This approach uses VA to conduct a collective voter verification process of eligible voters. To participate in the voting process, v must demonstrate their eligibility by verifying their true identity with the VA and obtaining the necessary rights to cast their votes. The GIBI-HE the scheme utilize the same underlying assumptions (such as the difficulty of the DL assumption) and key generation techniques, with the addition of a *phase 3* in the extract algorithm. An novel e-voting scheme is based on the six PPT algorithms:

$$\text{GIBI-HE} = (\mathcal{S}, \mathcal{E}_1, \mathcal{IP}, \mathcal{E}_2, \mathcal{T}, \mathcal{D})$$

1. **Setup ($\mathcal{S}$)**: On a security parameter 1^λ, TA takes cyclic group $\mathbb{G}$ prime number q, multiplicative group $\mathbb{Z}_q^*$ of prime order q. TA also selects random integer $x \in \mathbb{Z}_q^*$ to compute $y_1 = g_1^{-x}$ and $y_2 = g_2^{-x}$ where generator $g_1, g_2 \in \mathbb{G}$. Compute a hash function $H : \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \in \mathbb{Z}_q^*$. The master public key mpk is $(\mathbb{G}, q, g_1, g_2, y_1, y_2, H)$ while the master secret key msk is (x). The pair $(\mathsf{mpk}, \mathsf{msk})$ is passed to registry R_i to it respective RM_i.
2. **Extract ($\mathcal{E}_1$)**

   Phase 1: It is run by TA. There are multiple Registry $(\mathsf{R}_1, \mathsf{R}_2, ..., \mathsf{R}_i, ..., \mathsf{R}_m)$ which has respective Registry Manager $(\mathsf{RM}_1, \mathsf{RM}_2, ..., \mathsf{RM}_i, ..., \mathsf{RM}_m)$ and group of voters v. Take sample as $\mathsf{R}_1 = (\mathsf{RM}_1, \mathsf{v}_{1,1}, \mathsf{v}_{1,2}, ..., \mathsf{v}_{1,j}, ..., \mathsf{v}_{1,n})$, More generic way could be $\mathsf{R}_i = (\mathsf{RM}_i, \mathsf{v}_{i,1}, \mathsf{v}_{i,2}, ..., \mathsf{v}_{i,j}, ..., \mathsf{v}_{i,n})$, it takes as input $(\mathsf{mpk}, \mathsf{msk}, \mathsf{RM}_i)$, selects a random integer $t_i \in \mathbb{Z}_q^*$. Then, TA computes $A_i = g_1^{t_i}$, $B_i = g_2^{t_i}$ and $s_i = t_i + x\alpha_i$ where $\alpha_i = H(\mathsf{RM}_i, A_i, B_i, y_1, y_2)$. TA passes the registry public keys $\mathsf{rpk}_i = (\mathsf{RM}_i, g_1, g_2, y_1, y_2)$ and registry secret keys $\mathsf{rsk}_i = (\alpha_i, s_i)$ to registry manager RM_i.

   Phase 2: This phase is run by RM_i. Consider an eligible voter $\mathsf{v}_{i,j}$ who wants to register as a member of the voting list of the registry R_i. The voter sends their identity $\mathsf{v}_{i,j}$ to RM_i, which takes as input $(\mathsf{rpk}_i, \mathsf{rsk}_i, \mathsf{v}_{i,j})$. RM_i then selects a random integer $\hat{a}_i \in \mathbb{Z}_q^*$ and computes $\hat{A}_i = g_1^{\hat{a}_i}$, $\hat{B}_i = g_2^{\hat{a}_i}$ and $\hat{s}_i = \hat{a}_i + (\alpha_i, s_i)\beta_i$, where $\beta_i = H(\mathsf{v}_{i,j}, \mathsf{RM}_i, \hat{A}_i, \hat{B}_i)$. The RM_i outputs the voter's public key $\mathsf{vpk}_{i,j}$ where it consists of $(\mathsf{v}_{i,j}, \mathsf{RM}_i, g_1, g_2, \hat{A}_i, H, \hat{B}_i)$ and the voter's secret key $\mathsf{vsk}_{i,j} = (\beta_i, \hat{s}_i)$. The registry stores all $\mathsf{vpk}_{i,j}$ along with $\mathsf{v}_{i,j}$, and it passes $\mathsf{vsk}_{i,j}$ to all voters, respectively.

   Phase 3: For every voter $\mathsf{v}_{i,j}$, RM_i uses same random $\hat{a}_i \in \mathbb{Z}_q^*$ and $\hat{A}_i = g_1^{\hat{a}_i}$, $\hat{B}_i = g_2^{\hat{a}_i}$ from *Phase 2* as the secret and public key for encryption $\mathsf{esk}_i = \hat{a}_i$ and $\mathsf{epk}_i = g_1^{\hat{a}_i} g_2^{\hat{a}_i}$, we generate common public key $\mathsf{cpk} = \prod_{i=1}^{n} \mathsf{epk}_i$. Distribute cpk to all v. It will be useful for validating encrypted votes.
3. **Identification Protocol ($\mathcal{IP}$)**: Each voter has to prove their identity to VA. $\mathsf{v}_{i,j}$ as prover performs the transaction with a verifier VA_i. The ZK is carried out as follows:

   (a) CMT : $\mathsf{v}_{i,j}$ computes $E_i = g_1^{\hat{s}_i} \hat{A}_i^{\beta_i}$. Then, $\mathsf{v}_{i,j}$ generates a random integers $r_{i,1}, r_{i,2} \in \mathbb{Z}_q^*$, computes $X_i = g_1^{r_{i,1}} g_2^{r_{i,2}}$ and then sends (E_i, X_i) to VA_i.

   (b) CHA : VA_i picks a random challenge $c_i \in \mathbb{Z}_q^*$ and sends c_i to $\mathsf{v}_{i,j}$.

   (c) RSP : $\mathsf{v}_{i,j}$ computes response $Y_{i,1} = (r_{i,1} + c_i s_i)$, $Y_{i,2} = (r_{i,2} + c_i \hat{s}_i)$ and then sends the value of $(Y_{i,1}, Y_{i,2})$ to VA_i.

   VA_i calculates and accepts if the following equation holds for each i:

$$g_1^{Y_{i,1}} g_2^{Y_{i,2}} = X_i \cdot \left(\frac{E_i}{\hat{A}_i^{\beta_i}} \right)^{c_i}$$

   where $\beta_i = H(\mathsf{v}_{i,j}, \mathsf{RM}_i, g_1, g_2, , \hat{A}_i, \hat{B}_i)$ verified by itself since it is satisfied by DL-assumption.

Correctness Proof: The correctness of the ZK can be proven as such:

$$\begin{aligned} X_i \left(\frac{E_i}{\hat{A}^{\beta_i}}\right)^{c_i} &= g_1^{r_{i,1}} g_2^{r_{i,2}} \left(\frac{g_1^{s_i} g_2^{\hat{s}_i} \hat{A}_i^{\beta_i}}{\hat{A}_i^{\beta_i}}\right)^{c_i} \\ &= g_1^{r_{i,1}} g_2^{r_{i,2}} \cdot (g_1^{s_i} g_2^{\hat{s}_i})^{c_i} \\ &= g_1^{r_{i,1}+c_i s_i} g_2^{r_{i,2}+c_i \hat{s}_i} \\ &= g_1^{Y_{i,1}} g_2^{Y_{i,2}} \end{aligned}$$

4. **Encryption and Validation** ($\mathcal{E}_2$): Using cpk, each voter $\mathsf{v}_{i,j}$ casts a ballot $v_{i,j}$ and encrypts it as $ev_{i,j} = (a_{i,j}, b_{i,j})$. After the ballot is cast and sent to the VA_i, it are checked for validity. Using ZK proofs, VA_i verify that the vote is valid, i.e., that the voter has voted only for one candidate C_k. The voter, who is the prover, does the following:
   (a) *CMT*: For random integer $\hat{r}_{i,j} \in \mathbb{Z}_q^*$ and compute $ev_{i,j} = (a_{i,j}, b_{i,j}) = (g_1^{\hat{r}_{i,j}}, g_2^{v_{i,j}} \cdot \hat{A}_i^{\hat{r}_{i,j}})$. Also calculate the collapsed vote for the partial proof $\mathsf{pp}_{i,j} = \prod_{l=1}^{k} ev_{i,j}[l] = (\mathsf{pp}_{i,j}[0], \mathsf{pp}_{i,j}[1])$.
   (b) *CHA*: $\mathsf{v}_{i,j}$ generates random self-challenge $\hat{c}_{i,j} \in \mathbb{Z}_q$.
   (c) *RES*: Using the challenge $\hat{c}_{i,j}$, the voter $\mathsf{v}_{i,j}$ does the following:
   - $T_{i,j} = g_1^{\hat{c}_{i,j}}, \hat{T}_{i,j} = g_2^{\hat{v}_{i,j}}$,
   - $S_{i,j} = \mathsf{pk}^{\hat{c}_{i,j}}$,
   - $\mathtt{token} = H\big((\mathsf{pp}_{i,j}[0], \mathsf{pp}_{i,j}[1]) | T_{i,j} | \hat{T}_{i,j} | S_{i,j}\big)$
   - $s_{i,j} = \hat{r}_{i,j} \cdot \mathtt{token} + \hat{c}_{i,j}$
   - $\mathsf{v}_{i,j}$ sends $ev_{i,j}$ and partial proof to the respective VA_i,
   - *RES* = $(pp_{i,j}, T_{i,j}, \hat{T}_{i,j}, S_{i,j}, \mathtt{token}, s_{i,j})$.

   To verify the validity of a vote in an e-voting scheme, i.e, to check if the addition of the component of the vote equals to 1, the VA_i checks that the exponents from two equations from Eq. 5 are equal. If they are, the vote is valid and included in the final tally. If not, the vote is not valid and not included in the final tally. This helps ensure the security and integrity of the e-voting scheme.

$$g_1^{\mathtt{s}_{i,j}} = a_{i,j}^{\mathtt{token}} \cdot \left(\frac{T_{i,j}}{\hat{T}_{i,j}^{\mathtt{token}}}\right); \quad \mathsf{pk}^{\mathtt{s}_{i,j}} = \left(\frac{b_{i,j}}{g_1 g_2^{v_{i,j}}}\right)^{\mathtt{token}} \cdot S_{i,j} \tag{5}$$

5. **Tally** ($\mathcal{T}$): This phase is where valid votes are considered and passed to VA_i. The encrypted votes are collected from the voters and tallied by the VA_i and Ta_i. This algorithm just requires the collected ciphertexts $\{(a_{i,j}, b_{i,j})\}_{j=1}^{n}$ of valid votes. It combines all the encrypted vote without decrypting it. The Ta_i runs this algorithm, but does not decrypt the final tally and it is not authorized to display final result. Assume we are considering only votes from R_1. Thus, The combined encrypted votes are carried forward to VA_1. We include summation over all users though only valid votes are considered for ease of reading.

Here is a correctness proof to get summation of all the cast ballot valid votes by voter under R_1.

$$\begin{aligned} E(\prod_{j=1}^{n} v_{1,n}) &= (g_1^{\hat{r}_{1,1}}, g_2^{v_{1,1}} \cdot \hat{A}_1^{\hat{r}_{1,1}}) \cdot (g_1^{\hat{r}_{1,2}}, g_2^{v_{1,2}} \cdot \hat{A}_1^{\hat{r}_{1,2}}) \cdots (g_1^{\hat{r}_{1,n}}, g_2^{v_{1,n}} \cdot \hat{A}_1^{\hat{r}_{1,n}}) \\ &= (g_1^{\hat{r}_{1,1}+\hat{r}_{1,2}+\cdots+\hat{r}_{1,n}}, g_2^{v_{1,1}+v_{1,2}+\cdots+v_{1,n}} \cdot \hat{A}_1^{\hat{r}_{1,1}+\hat{r}_{1,2}+\cdots+\hat{r}_{1,n}}) \\ &= (g_1^{\sum_{j=1}^{n} \hat{r}_{1,j}}, g_2^{\sum_{j=1}^{n} v_{1,j}} \cdot \hat{A}_i^{\sum_{j=1}^{n} \hat{r}_{1,j}}) \\ &= (a_1, b_1) \end{aligned}$$

For any registry R_i where (a_i, b_i) represents the encrypted sum of all valid votes cast by voters under registry R_i, we generally define tally as defined below.

$$\begin{aligned} E\Big(\prod_{j=1}^{n} v_{i,j}\Big) &= (g_1^{\sum_{j=1}^{n} \hat{r}_{i,j}}, g_2^{\sum_{j=1}^{n} v_{i,j}} \cdot \hat{A}_i^{\sum_{j=1}^{n} \hat{r}_{i,j}}) \\ &= (a_i, b_i) \end{aligned}$$

6. **Decrypt** ($\mathcal{D}$): As there is no central secret/decryption key, Decrypt involves two rounds of communication.
   (a) The combined ciphertext (a, b) from the tallying stage is forwarded to all voters. All voters compute their partial shares $a^{\hat{a_i}}$ for the secret key $\hat{a}_i$. These are sent back to the VA_i.
   (b) Then by Equation 3, the partial shares are combined together and DL assumption is performed to retrieve the final tally. Then VA_i to decrypt the combined vote v.

$$\frac{b}{\prod_{i=1}^{n} a^{\hat{a_i}}} = \frac{b}{a^{\hat{a_1}+\ldots+\hat{a_n}}} = g^v \tag{6}$$

   Finally, the total number of votes v can be revealed by computing DL assumption.

4 Security Model and Security Proof

In this section, we cover two areas, namely the security model of the our scheme, and also the security proofs based on the security models. The following security model outlines the different potential threats and how our scheme addresses them.

4.1 Malicious Third Party TP

A malicious TP is only able to eavesdrop on encrypted votes and may attempt to impersonate other voters using information obtained from eavesdropping to engage in malicious activities against the registry or list of valid voters.

TP Acts as Registry Manager

Definition 3. *A malicious* TP *may try to impersonate* RM_i *to allow registration right without checking eligibility of voters. However, it is not possible if* TP *does not have* $(\mathsf{rpk}_i, \mathsf{rsk}_i)$ *which is tied to the* $(\mathsf{mpk}, \mathsf{msk})$ *from respective* R_i*. It is difficult to learn* msk *from* TA *and Additionally, the* TP *does not have access to the list of voters within the registry* R_i*.*

TP Act as Voter

Definition 4. *A malicious* TP *may try to impersonate a voter* $\mathsf{v}_{i,j}$ *by forming own* $(\mathsf{vpk}_{i,j}, \mathsf{vsk}_{i,j})$*. However,* $\mathsf{v}_{i,j}$ *is required to register through the* RM_i *under registry* R_i*. This makes it impossible for the* TP *to impersonate a legitimate voter in the registry.*

TP Act as Voting Authority

Definition 5. *A malicious* TP *may try to impersonate a* VA_i *to allow anyone to give the rights to cast a ballot without checking eligibility. But,* VA_i *runs the* ZK *for all voters* v*. The* RES *generated by any* $\mathsf{v}_{i,j}$ *includes of respective* $\mathsf{vsk}_{i,j}$*. It is impossible for* VA_i *to eliminate* ZK *in this process and proceed for voting.*

TP Act as Tallying Authority

Definition 6. *A malicious* TP *may try to impersonate a* Ta_i *to tally all encrypted votes. although, it is not possible if he does not have able to decrypt it or he can forge the original votes. He can not perform any operations on encrypted voted without voters respective partial share.*

4.2 Malicious Voter

A malicious $\mathsf{v}_{i,j}$ can not impersonate as any authority but it can act as RM_i and other voter from different R_i.

Voter Acts as Registry Manager

Definition 7. *Malicious* $\mathsf{v}_{i,j}$ *may try to replicate the role of the* RM_i *by generating their own* $(\mathsf{rpk}_i, \mathsf{rsk}_i)$ *to target certain voters within the registry and trick them into giving him their consent to be able to perform registry verification in next step.*

Voter Acts as Another Registry Voter

Definition 8. *Malicious voter may try to impersonate another registry voter by using* $(\mathsf{v}_{i,j}, \mathsf{vsk}_{i,j})$ *and it is hard to obtain* $\mathsf{vsk}_{i,j}$ *because of randomness introduced in key setup.*

4.3 Malicious Registry Manager RM_i

RM_i **Act as Another Voter within same registry**:

Definition 9. *The* RM_i*'s task is to generate voter's keys* $(\mathsf{vpk}_{i,j}, \mathsf{vsk}_{i,j})$ *for all the voters using the registry keys* $(\mathsf{rpk}_i, \mathsf{rsk}_i)$*, while keeping track of the voter in a list. Considering the authority and role that a* RM_i *has over their own group, a malicious* RM_i*'s goal is to be able to impersonate voter to obtain* cpk *for next ZK protocol.*

RM_i **Acts as Another Voter from Another Registry**:

Definition 10. *Considering the authority and role that a* RM_i *has over their own group, a malicious* RM_i*'s goal is to be able to impersonate another registry's* RM *to obtain the list of different set of voters.*

4.4 Malicious Voting Authority VA_i

VA_i **Acts as Registry Manager** RM_i

Definition 11. *A malicious* VA_i *may attempt to replicate the role of* RM_i *by generating their own registry keys* $(\mathsf{rpk}_i, \mathsf{rsk}_i)$ *and granting all voters eligibility. However, it is not possible for the* VA_i *to have all ZK proof from voters and they will not be able to carry forward encrypted votes to the* Ta_i *without knowledge of the total votes. In this case, the votes will be discarded.*

VA_i **Acts as Tallying Manager** Ta_i :

Definition 12. VA_i *acts as* Ta_i *by generating tallying of encrypted votes to voter and run partial decryption phase. Impersonator* VA_i *playing role of* Ta_i *should be collected cipher text and satisfy correctness proof before displaying result on* BB.

4.5 Malicious Tallying Authority Ta_i

A malicious Ta_i may attempt to alter tally the encrypted ballots during the vote tallying process, but cannot impersonate voters or RM_i as they are not involved in this scenario. The security of the e-voting system relies on the ability of Ta_i to accurately and securely tally the encrypted votes without access to their content.
Ta_i **Acts as** VA_i

Definition 13. *A malevolent* Ta_i *may try to replicate the role of* VA_i*, to allow the verifying all voters with* ZK *and alter encrypted votes. However, it is not possible if* Ta_i *can not generate the* ZK *proof and alter casted ballot from respective* $\mathsf{v}_{i,j}$.

4.6 Security Proof

The proposed e-voting system satisfies the security requirements of *eligibility, privacy, unreusability, fairness, receipt-freeness, individual* and *universal verifiability, uncoercibility*, and protection against attack under random oracle (RO), provided that at least one of the authorities is trustable.

Security Against Impersonation as Malicious Third Party and Malicious Voter.

Theorem 1. *The GIBI-HE scheme above is* (t, q, ε)*-secure against impersonation in the random oracle model if the DL assumption of the vote holds such that:*

$$\varepsilon_{\text{GIBI-HE}} \leq \sqrt[l]{\varepsilon^{\text{DL}}_{\mathbb{G},\mathbb{C}}(k) + (\frac{1}{2^k} + \frac{1}{2^k} + \frac{1}{2^k})}$$

Proof. In this security game, Impersonator $\mathbb{I}$ who (t, q, ε) breaks the security of GIBI-HE scheme. Challenger $\mathbb{C}$ acts as simulator which helps to find the value of DL assumption. $\mathbb{C}$ will simulate $\mathbb{I}$ as following:

- **Setup.** $\mathbb{C}$ obtains master public key, $\mathsf{mpk} = (\mathbb{G}, q, g_1, g_2, y_1, y_2, H)$ by giving input 1^λ and passes mpk to $\mathbb{I}$.
- **Phase 1.** $\mathbb{I}$ can issue multiple set of queries $(\mathsf{q}_0, ..., \mathsf{q}_i, ..., \mathsf{q}_m)$ where q_i for $\mathsf{v}_{i,j}$ and there are m queries in total. In training phase, $\mathbb{I}$ attempts to learn from the $\mathbb{C}$ and will try to forge vsk; using vsk will run the further transcript of scheme. GIBI-HE is considered to be advance version of GIBI in combination with HE without the use of pairing in it for set of voter where there voters belongs to registry and multi-computation has different authority on one level to define group like structure.
  - **Case 1:** $\mathsf{v}_{i,j} \neq \mathsf{v}^*_{i,j}$ where $\mathsf{v}^*_{i,j}$ is targeted voter.
    - **Extract Query.** For $\mathsf{v}_{i,j} \neq \mathsf{v}^*_{i,j}$, $\mathbb{I}$ can continue to query the $\mathsf{vpk}_{i,j}$ of $\mathsf{v}_{i,j}$ as long as $\mathsf{v}_{i,j}$ is not an ancestor voter of $\mathsf{v}^*_{i,j}$. $\mathbb{C}$ takes cpk and RM_i. When $\mathbb{I}$ being queried with $\mathsf{vpk}_{i,j}$ and it returns $\mathsf{vsk}_{i,j} = (\beta, \hat{s}_i)$ to $\mathbb{I}$ and $\mathbb{C}$ generates epk_i and generate cpk, finally pass to $\mathbb{I}$.
    - **Identification Query.** $\mathbb{C}$ response with IP. In simulation, prover takes input $(\mathsf{v}_{i,j}, \mathsf{vsk}_{i,j}, \mathsf{rpk}_i)$ whereas verifier takes input $(\mathsf{rpk}_i, \mathsf{v}_{i,j})$. Prover generates (E_i, X_i) and $\mathbb{C}$ throws random challenge $c_i \in \mathbb{Z}^*_q$. Based on challenge prover calculate the response $Y_{i,1}, Y_{i,2}$ and send to verifier. Lately verifier verifies $g_1^{Y_{i,1}} g_2^{Y_{i,2}} = X_i \cdot \left(\frac{E_i}{\hat{A}_i^{\beta_i}}\right)^{c_i}$. This ZK proof help to check *eligibility* and avoid re-usability of voter to have ambiguous vote; it satisfy property for our GIBI-HE scheme.
    - **Encrypt Query.** $\mathbb{I}$ casts vote $v_{i,j}$ and encrypt $ev_{i,j}$ pass to $\mathbb{C}$. It protects the *integrity* of the casted ballot using DE encryption method.
    - **Validation Query.** Using ZK proof, in a simulator prover as $\mathsf{v}_{i,j}$ and verifier as VA communicates and based on random self challenge by $\mathsf{v}_{i,j}$, it generate response $RES = (pp_{i,j}, T_{i,j}, \hat{T}_{i,j}, S_{i,j}, \texttt{token}, s_{i,j})$, sends to $\mathbb{I}$. Another ZK proof avoids coercer and maintain *uncoercibility* which helps to eliminate vote buying. $\mathsf{v}_{i,j}$ will carry `token` and it ensures *receipt-freeness* for every valid voter.
    - **Decrypt Query.** $\mathbb{I}$ will provide (a, b) from tallying and forward to $\mathsf{v}_{i,j}$ to calculate partial share $a^{\hat{\alpha}_i}$ and send to $\mathbb{C}$. Only valid voters can decrypt the casted ballet using $a^{\hat{\alpha}_i}$ and it satisfies the property of *fairness* and *privacy* is maintained throughout the voting process.
  - **Case 2:** $\mathsf{v}_{i,j} = \mathsf{v}^*_{i,j}$
    - **Extract Query.** For $\mathsf{v}_{i,j} = \mathsf{v}^*_{i,j}$, the ancestor of vsk_{i,j^*} is unknown. But, the registry secret key rpk_i is known. Therefore, the algorithm aborts. There is Registry $\mathsf{R_i}$ where where all $\mathsf{RM_i}$ are defined as parent and voters in $\mathsf{RM_i}$ child node according to hierarchy under that respective $\mathsf{R_i}$.vsk helps to generate vsk of child eligible voter. Child node's vsk is generated only in case it has vsk's vrsk defined. $\mathbb{C}$ takes mpk and $\mathsf{v}^*_{i,j}$ as the input. Upon being queried with the public key of $\mathsf{v}^*_{i,j}$ and returns $\mathsf{vsk}_{i,j^*} = (s^*_i, \hat{\beta}^*_i)$ to $\mathbb{I}$.

* **Identification Query.** When transcript will create even if not yet queried before as an extract query. $\mathsf{v}^*_{i,j}$ as prover participate in transcript and add in the set. VA^*_i will not able to issue transcript for the already corrupted voter. $\mathsf{v}^*_{i,j}$ is targeted identity of voter and VA^*_i needs to verify it. $\mathsf{v}_{i,j} = \mathsf{v}^*_{i,j}$, $\mathbb{I}$ act as the cheater VA^*_i and $\mathbb{C}$ does not have user secret key of $\mathsf{v}^*_{i,j}$, however it needs to create it again to run an identification protocol. When $\mathbb{I}$ tries to forge $\mathsf{v}^*_{i,j}$ then he should know the previous RM_i. We can perform transcript as many times as number of queries does not exceed. Prover takes input $(\mathsf{rpk}^*_i, \mathsf{v}^*_{i,j}, \mathsf{vsk}^*_{i,j})$ where the verifier takes input $(\mathsf{rpk}^*_i, \mathsf{v}^*_{i,j})$. Prover generates (E^*_i, X^*_i). $\mathbb{C}$ generates random challenge $c^*_i \in \mathbb{Z}^*_q$ where corresponds to $\mathsf{v}^*_{i,j}$. On the basis of challenge prover calculates $Y^*_{i,1}, Y^*_{i,2}$ to VA^*_i as its response. Lastly VA^*_i verifies $\mathsf{v}^*_{i,j}$ of $g_1^{Y^*_{i,1}} g_2^{Y^*_{i,2}} = X^*_i \cdot \left(\frac{E^*_i}{\hat{A}_i^{\beta_i}}\right)^{c^*_i}$. This ZK proof provide *integrity* for valid voters.
* **Encrypt Query.** When $\mathbb{I}$ casts vote $v_{i,j}$ * and encrypt $ev^*_{i,j}$ pass to $\mathbb{C}$.
* **Validation Query.** Running validation query using voters self generated $\hat{c}^*_{i,j}$ abort the condition and fails to receive RES. This stage where $\mathsf{v}^*_{i,j}$ to verify the correctness of their vote without revealing the contents of the vote to anyone else which justify property of *individual verifiability*.
* **Decrypt Query.** $\mathsf{v}^*_{i,j}$ fails to run decrypt query since $\mathsf{v}^*_{i,j}$ does not hold any combined ciphertext and will fail to compute partial share for $\hat{a}^*_i$. Anyone can independently verify that the votes were accurately cast, collected, tallied, and the results were correctly computed in this stage that means it satisfy the property of *universal verifiability* and it allows for public scrutiny and helps to detect any potential fraud or errors.

- **Challenge** ($\mathbb{C}$). $\mathbb{I}$ outputs an $\mathsf{v}_{i,j} \neq \mathsf{v}^*_{i,j}$ that it wishes to impersonate.

– **Phase 2**. Impersonation phase where $\mathbb{I}$ acts as a cheating verifier and try to convince $\mathbb{C}$ based on information gathered in the phase 1. $\mathbb{I}$ wins the game if it is successful in convincing the verifier to accept with non-negligible probability. Breaking phase calculates as follows:

$[r_{i,1}, c_1, X_i, Y_{i,1}]$ and $[r_{i,2}, c_2, X_i, Y_{i,2}]$ from $\mathbb{I}$ where $c_1 \neq c_2$. From here, $\mathbb{C}$ extracts $\widetilde{\hat{s}_i} = (Y_{i,1} - Y_{i,2})/(c_2 - c_1)$ and $\widetilde{\beta_i} = (Y_{i,1} - Y_{i,2})/(c_2 - c_1)$.

If $\beta_i = \widetilde{\beta_i}$ and $\hat{s}_i = \widetilde{\hat{s}_i}$ then $\mathbb{C}$ aborts.

$$
\begin{aligned}
g_1^{\beta_i} g_1^{\hat{s}_i} &= g_1^{\widetilde{\beta}_i} g_1^{\widetilde{\hat{s}}_i} \\
g_1^{\beta_i + a\hat{s}_i} &= g_1^{\widetilde{\beta}_i + a\widetilde{\hat{s}}_i} \\
g_1^{a\hat{s}_i} - g_1^{a\widetilde{\hat{s}}_i} &= g_1^{\widetilde{\beta}_i} - g_1^{\beta_i} \\
g_1^{a} &= g_1^{(\widetilde{\beta}_i - \beta_i)(\hat{s}_i - \widetilde{\hat{s}}_i)} \\
a &= -\frac{\widetilde{\beta}_i - \beta_i}{\hat{s}_i - \hat{s}_i}
\end{aligned}
$$

To calculate the probability of $\mathbb{C}$ winning the game to solve the DL assumption. we shall successfully extract $phase$ 2 valid conversations to derive $(\beta_i, \hat{s}_i)$ and encrypt $ev = (a_{i,j}, b_{i,j})$ and later calculating a with the probability $\varepsilon_{\text{GIBI-HE}} = (-\frac{1}{2^k} - \frac{1}{2^k})^l$. Assume $\mathbb{C}$ solves the DL assumption. $\mathbb{C}$ which computes correct value of a event is A where it accepts $ev_{i,j}$ and not aborting event is B. Winning probability can be given as following.

$$
\begin{aligned}
\mathbb{C} &= \Pr[A \wedge B] \\
\mathbb{C} &= \Pr[A|B]\Pr[B] \\
\varepsilon^{\text{DL}}_{\mathbb{G},\mathbb{C}}(k) &\geq (\varepsilon_{\text{GIBI-HE}} - \frac{1}{2^k} - \frac{1}{2^k})^l - \frac{1}{2^k}
\end{aligned}
$$

The probability of $\mathbb{C}$ aborting when event B is $\beta_i = \widetilde{\hat{s}}_i$ and $\hat{s}_i = \widetilde{\hat{s}}_i$ moreover $ev_{i,j} = (a_{i,j}, b_{i,j})$. Therefore probability of winning $\mathbb{C}$ is,

$$
\begin{aligned}
\varepsilon^{\text{DL}}_{\mathbb{G},\mathbb{C}}(k) &\geq (\varepsilon_{\text{GIBI-HE}} - \frac{1}{2^k} - -\frac{1}{2^k})^l - \frac{1}{2^k} \\
\varepsilon^{\text{DL}}_{\mathbb{G},\mathbb{C}}(k) + \frac{1}{2^k} &\geq (\varepsilon_{\text{GIBI-HE}} - \frac{1}{2^k} - \frac{1}{2^k})^l \\
\varepsilon_{\text{GIBI-HE}} &\leq \sqrt[l]{\varepsilon^{\text{DL}}_{\mathbb{G},\mathbb{C}}(k) + (\frac{1}{2^k} + \frac{1}{2^k} + \frac{1}{2^k})}
\end{aligned}
$$

Security Against Impersonation as Registry Manager. We define the security proof against impersonation as a RM_i, where a simulation game between a Challenger $\mathbb{C}$ and an Impersonator $\mathbb{I}$ is constructed. The goals of $\mathbb{C}$ and $\mathbb{I}$ are defined to solve the hard problem of the scheme and to impersonate as RM_i, respectively.

Theorem 2. *The* GIBI-HE *scheme above is* (t, q, ε)*-secure against impersonator as registry manager in the random oracle model if the DL hard problem for* GIBI-HE *holds.*

Proof. In this game, we construct a Challenger $\mathbb{C}$ making use of an Impersonator $\mathbb{I}$.

– **Setup.** Similar to the proof described as Setup in proof of Theorem 1.
– **Phase 1.** Impersonator $\mathbb{I}$ tries to fire queries to check fragile nature of simulator. Query set contains $(\mathsf{q}_0, \mathsf{q}_1, ..., \mathsf{q}_m)$ where m is the last query $\mathbb{I}$ can ask for.
  - **Case 1:** $\mathsf{RM}_i \neq \mathsf{RM}_i^*$ and $\mathsf{v}_{i,j} \neq \mathsf{v}_{i,j}^*$ Outside malicious identity acts as and RM_i and tries to give access and generate vsk for authorized voters.
    * **Extract Query**. When $\mathbb{I}$ as RM_i is an outside identity, then $\mathbb{C}$ cannot forge an identity string, or rsk for other RM and voter identities. Extract oracle aborts here in this case. $\mathbb{C}$ still can generate cpk_i and pass to the $\mathbb{I}$ which can not in use for RM_i since RM_i can not act as voter for further ZK protocol for authentication. RM_i does not hold its respective secret key for authentication.
    * **Identification Query**. In the simulator, RM_i act as the prover and VA_i act as verifier. For the voter authentication, simulator runs ZK with all voter under RM_i. $\mathsf{v}_{i,j}$ who holds valid $\mathsf{vsk}_{i,j}$ using parent-child rsk_i and msk will only get validated with $c_i \in \mathbb{Z}_q^*$ and $\mathsf{v}_{i,j}$ generate $(Y_{i,1}, Y_{i,2})$ and pass to VA_i. $\mathsf{v}_{i,j}$ under RM_i which does not hold set of $(\mathsf{rsk}_i, \mathsf{msk})$ will get eliminate or completely abort from voting process and satisfy property of *eligibility* of proposed e-voting.
    * **Encrypt Query.** $\mathbb{I}$ casts vote $v_{i,j}$ and encrypt only valid vote $ev_{i,j}$. This step provide *unreuability* of the votes. Validating valid ballots from valid voters can provide *recipet-freeness* by issuing `token` to valid $\mathsf{v}_{i,j}$.
    * **Validation Query.** Using ZK, simulator holds the communication script for $\mathsf{v}_{i,j}$ by generating self challenge to generate RES and passes to $\mathbb{I}$.
    * **Decrypt.** $\mathbb{I}$ will provide with (a, b) from the tally and forward to only authrozied voter using $a^{\hat{a}_i}$ and passes to $\mathbb{C}$. *uncoercibility* is maintained where $\mathsf{v}_{i,j}$ can not prove coercer how he has voted because of the dual randomness property in our scheme.
  - **Case 2:** $\mathsf{RM}_i = \mathsf{RM}_i^*$ and $\mathsf{v}_{i,j} = \mathsf{v}_{i,j}^*$
    * **Extract Query.** Extract oracle does not abort for example RM_i and voters v under it. $\mathbb{C}$ simulate for targeted RM_i^* same as Case 2 in proof of Theorem 1.
    * **Identification Query.** For *individual verifiability*, ZK provide authorization for eligible voter $\mathsf{v}_{i,j}^*$ right to cast ballot. ZK can be proven using random challenge $c_i^* \in \mathbb{Z}_q^*$ and prover calculates $Y_{i,1}^*, Y_{i,2}^*$ to VA_i^* as its response. DL-tuple is verified here for RM_i^* and all valid voters under it.
    * **Encrypt Query.** Similar to Case 2 encrypt oracle in proof of Theorem 1.
    * **Validate Query.** For $\mathsf{v}_{i,j}^*$ under RM_i^* validate by self generated challenge $\hat{c}_{i,j}^*$ and aborts for malicious votes or wrong ballots using $\mathsf{pp}_{i,j}^*$. All unwanted ballots are discarded from final tally and outcome and fails to generate RES^*. This satisfy the property of *individual verifiability* where $\mathsf{v}_{i,j}^*$ to verify that vote is recorded or not by later checking on BB.
    * **Decrypt Query.** $\mathsf{v}_{i,j}^*$ discard from decrypt oracle since it will not have (a, b) to satisfy DE assumption and eventually can not calculate its ballot $v_{i,j}$. In case of authorized voters, the case is opposite and so *universal verifiability* to be satisfied by any valid $\mathsf{v}_{i,j}^*$ under RM_i^* verify the accuracy of the election outcome.

– **Phase 2**. $\mathbb{I}$ pretends to be a valid voter using $v^*_{i,j}$, where $v^*_{i,j}$ was queried during the extract query. $\mathbb{I}$ generates a voter's $\mathsf{vsk}^*_{i,j}$ and then sends the response to $\mathbb{C}$. After $\mathbb{C}$ obtains the $\mathsf{vsk}^*_{i,j}$, $\mathbb{C}$ checks the validity of the votes [1]. If the $\mathsf{vsk}^*_{i,j}$ produced by $\mathbb{I}$ is not valid, $\mathbb{C}$ aborts and it fails in the security game. Else, $\mathbb{C}$ can use the forgery $E_{\mathsf{v}_\mathbb{I}}$ to solve the DL hard problem used in the scheme for GIBI and DE and wins in the security game. We now analyze the probability of aborts during the whole simulation process for malicious RM^*_i.

$$\Pr[\mathbb{C} \text{ wins}] = \Pr[\mathbb{C} \text{ accepts } ev^*_{i,j}] - \Pr[\mathbb{C} \text{ no abort}]$$

During the query phase of the GIBI scheme, the occurrence of aborts depends on the specific GIBI scheme used. However, the probability of aborts due to a hash collision is negligible. Thus, if an attacker $\mathbb{I}$ is able to come up with valid $\mathsf{vsk}^*_{i,j}$ during the query phase, it can be inferred that $\mathbb{I}$ has broken the DE encryption scheme used in the GIBI-HE scheme. This is because the attacker $\mathbb{I}$ would have had to produce valid $\mathsf{vsk}^*_{i,j}$ and ballot $v^*_{i,j}$ on the encrypted ballots $ev^*_{i,j}$, which is only possible if the attacker has access to the private key used for GIBI and DE scheme.

These proof will show that our scheme satisfies all of the required security properties, including *eligibility, unreusability, fairness, receipt-freeness, individual* and *universal verifiability, uncoercibility*, and protection against attack under the RO model. It will also demonstrate that at least one of the authorities in the system can be trusted to maintain the integrity of the voting process.

Theorem 1 proves the semantic security of the scheme for security model 4.1 and 4.2 , which means that an attacker who has access to the public parameters and the encryption oracle cannot distinguish between encryptions of two different votes; makes it $robust$ in nature. Theorem 2 proves the security of the scheme for security model 4.3 in the presence of malicious authorities using DL assumption, meaning that even if some authorities behave maliciously by modifying encrypted votes or sending incorrect votes, the scheme remains secure and maintain its *privacy* on all levels.

To prove the existence of simulators for malicious VA_i 4.4 and Ta_i 4.5, we can use the same simulator as in Theorem 2. This simulator constructs a "real world" transcript of interactions between the adversary and the authorities, and then constructs an "ideal world" transcript by simulating the authorities' behavior. By comparing the two transcripts, we can prove that the adversary cannot distinguish between them, which implies the security of the scheme.

Thus, with the combination of Theorem 1 and Theorem 2, we can provide a convincing proof that the scheme is secure against malicious authorities in proposed GIBI-HE e-voting scheme, and that simulators can be constructed for such attacks.

[1] It is noted that $\mathbb{I}$ has to produce the $v^*_{i,j}$ of a valid voter in the registry, else $v^*_{i,j}$ will fail when $\mathbb{C}$ does cross-checking on the validity of $v^*_{i,j}$ as a registry voter list.

5 Efficiency Analysis

In this section, we evaluate the efficiency of previous related e-voting scheme and then determine the efficiency cost of the GIBI-HE scheme. There are no IBI-based e-voting schemes, thus we explore the closest substitute: GS scheme for e-voting.

Yang et al. [28] use DS and ElGamal encryption in their e-voting scheme to achieve almost the same security guarantees as this paper but their system uses a total of P points split between candidates. They do not provide the configuration of their setup or the DS used. The total computation time for a voter can be presented as the total computation time of encryption, partial proofs, ZK, and the signature scheme is $2t \times k \times L_P \times 5t \times n_c \times L_P + 2t + t$, where t is time of one exponentiation, L_P is total available points, and k candidates.

Next we consider the e-voting GS scheme by Malina et al. [23], which use ElGamal encryption and GS for verification. Bilinear Pairings (BP) are used for for this scheme which are computationally expensive. This issue can be mitigated by using batch verification, which reduces the number of BP operations required by the system. The number of BP e operations can be decreased from $n \times k$ (where n is the number of signatures and k is the number of BP operations during individual message verification) to l (where l is the number of BP operations during batch verification). This can help to increase the scheme's efficiency and decrease its computation overhead.

In Table 2, we consider alternative schemes in order to calculate the identification cost. We consider the IBI scheme which is an efficient scheme in the presence of targeted identity.

Table 2. Comparison with other similar schemes

Work	**Assumption**	**Scheme**	**Encryption**	**Efficiency**	**Security**
Yang et al. [28]	DL	DS	DE	$O(2t \times k)$	Standard
Malina et al. [23]	BP	GS	ElGamal	$O(n \times k \times p)$	unknown
Our work	DL	GIBI	DE	$O(k\log n)$	RO

Legends: t is time of one exponentiation, k number of candidates, n is time of group operation, RO random oracle, and big-O is complexity.

In Table 3, we calculate the computational cost for our new GIBI-HE e-voting scheme. We consider k candidates and n voters, setup to validation phase run $n \times k$ times. Tallying is run n times for each VA. Decrypt is distributed and run once involving n voters.

According to operational cost analysis, the GIBI-HE scheme is more efficient and secure than the other group DS and IBI schemes.

6 Future Work

We have several ideas for enhancing and improving our e-voting scheme. The full implementation and the formal security proof are being worked on at the moment and

Table 3. Efficiency Analysis for the GIBI-HE Scheme.

Algorithm	$\mathbb{E}$	$\mathbb{A}$	$\mathbb{Z}_q$	$\mathbb{G}$	$\mathsf{R}_{\mathsf{comm}}$
Setup	1	0	0	1	0
Extract	1	2	1+n	0	1
Identification	3	1	1	1	1
Encrypt	3	0	1	1	0
Validation	2	0	k	1	1
Tally	0	0	n	0	1
Decrypt	1	0	n+1	0	1

For k candidates, n voters, $\mathbb{E}$ Exponentiation in $\mathbb{Z}_q^*$, $\mathbb{A}$ Addition in $\mathbb{Z}_q^*$, Multiplicative $\mathbb{Z}_q$, Randomness in $\mathbb{G}$, and $\mathsf{R}_{\mathsf{comm}}$ Round of communications

shall be added to the paper. One possibility is the introduction of one-time-use or time-based identification protocols, which can further enhance the security of the scheme. Another avenue of exploration is the use of different architectures of IBI schemes, such as those based on rings or trees, rather than the group-based approach we have used in our current scheme. Finally, we are also interested in exploring the use of post-quantum cryptosystems such as lattice-based cryptography for e-voting, in order to ensure the long-term security and viability of our system.

7 Conclusion

In conclusion, the proposed e-voting scheme using GIBI-HE provides a secure and efficient solution for conducting elections electronically. The GIBI scheme enables voters to register for elections and ensures the *eligibility* and *unreusability* of their votes through the use of a ZK protocol. The use of DE encryption in a group-like structure allows for secure multiparty communication and ensures *fairness* and *receipt-freeness* in the voting process.

The scheme also generates proof of votes for each voter and allows for *individual* and *universal verifiability* through the use of additional ZK proofs. The use of partial shares for decryption makes the system independent of any central authority for vote decryption. The proposed scheme is secure under various scenarios and *robust* in the RO model. However, it is important to note that *uncoercibility* must be maintained by authorities to prevent human error from compromising the security of the system. Therefore, the GIBI-HE e-voting scheme is a novel and secure method for conducting elections electronically.

References

1. International institute for democracy and electoral assistance. `https://www.idea.int/data-tools/question-view/742`
2. Adida, B.: Helios: Web-based open-audit voting. In: USENIX security symposium. vol. 17, pp. 335–348 (2008)
3. Bellare, M., Namprempre, C., Neven, G.: Security proofs for identity-based identification and signature schemes. Journal of Cryptology **22**(1), 1–61 (2009)

4. Bellare, M., Palacio, A.: Gq and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In: Annual International Cryptology Conference. pp. 162–177. Springer (2002)
5. Benaloh, J.D.C.: Verifiable secret-ballot elections. Yale University (1987)
6. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Annual international cryptology conference. pp. 213–229. Springer (2001)
7. Boneh, D., Franklin, M.K.: Predicate encryption and simple definitions of secrecy. International Conference on the Theory and Application of Cryptographic Techniques pp. 319–335 (1997)
8. Chang, D., Chauhan, A.K., Kang, J., et al.: Apollo: End-to-end verifiable voting protocol using mixnet and hidden tweaks. In: ICISC 2015. pp. 194–209. Springer (2015)
9. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM **24**(2), 84–88 (1981). https://doi.org/10.1145/358549.358563, http://doi.acm.org/10.1145/358549.358563
10. Chaum, D.: Blind signatures for untraceable payments. In: Advances in cryptology. pp. 199–203. Springer (1983)
11. Chaum, D.: Elections with unconditionally-secret ballots and disruption equivalent to breaking rsa. In: Workshop on the Theory and Application of of Cryptographic Techniques. pp. 177–182. Springer (1988)
12. Chaum, D., van Heyst, E., Pfitzmann, B.: Group signatures. International Conference on the Theory and Application of Cryptographic Techniques pp. 257–265 (1991)
13. Chin, J.J., Vangujar, A.K., Ng, T.S., Yip, S.C., Chia, J.: Group identity-based identification: Definitions, construction and implementation. Malaysian Journal of Mathematical Sciences **15**, 123–139 (2021)
14. Choon, J.C., Hee Cheon, J.: An identity-based signature from gap diffie-hellman groups. In: International workshop on public key cryptography. pp. 18–30. Springer (2003)
15. Cocks, C.C.: Covert security of key-agreement protocols. International Conference on the Theory and Application of Cryptographic Techniques pp. 260–275 (2001)
16. Cortier, V., Galindo, D., Glondu, S., Izabachene, M.: Distributed elgamal á la pedersen: application to helios. In: Proceedings of the 12th ACM Workshop on Workshop on Privacy in the Electronic Society. pp. 131–142 (2013)
17. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Annual International Cryptology Conference. pp. 174–187. Springer (1994)
18. Gamal, T.E.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inf. Theory **31**(4), 469–472 (1985). https://doi.org/10.1109/TIT.1985.1057074, https://doi.org/10.1109/TIT.1985.1057074
19. Haines, T., Goré, R., Sharma, B.: Did you mix me? formally verifying verifiable mix nets in electronic voting. In: 2021 IEEE Symposium on Security and Privacy (SP). pp. 1748–1765. IEEE (2021)
20. Haines, T., Lewis, S.J., Pereira, O., Teague, V.: How not to prove your election outcome. In: 2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020. pp. 644–660. IEEE (2020). https://doi.org/10.1109/SP40000.2020.00048, https://doi.org/10.1109/SP40000.2020.00048
21. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Proceedings of the 2005 ACM workshop on Privacy in the electronic society. pp. 61–70 (2005)
22. Kurosawa, K., Heng, S.H.: From digital signature to id-based identification/signature. In: International Workshop on Public Key Cryptography. pp. 248–261. Springer (2004)
23. Malina, L., Smrz, J., Hajny, J., Vrba, K.: Secure electronic voting based on group signatures. In: 38th International Conference on Telecommunications and

Signal Processing, TSP 2015, Prague, Czech Republic, July 9-11, 2015. pp. 6–10. IEEE (2015). https://doi.org/10.1109/TSP.2015.7296214, https://doi.org/10.1109/TSP.2015.7296214
24. Malina, L., Smrz, J., Hajny, J., Vrba, K.: Secure electronic voting based on group signatures. In: 2015 38th International Conference on Telecommunications and Signal Processing (TSP). pp. 6–10 (2015). https://doi.org/10.1109/TSP.2015.7296214
25. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: International conference on the theory and application of cryptology and information security. pp. 552–565. Springer (2001)
26. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Workshop on the theory and application of cryptographic techniques. pp. 47–53. Springer (1984)
27. Wang, Y., Chen, G.J.: Secure and efficient e-voting system based on distributed elgamal. International Conference on Computer Science and Software Engineering pp. 399–404 (2006)
28. Yang, X., Yi, X., Nepal, S., Kelarev, A., Han, F.: A secure verifiable ranked choice online voting system based on homomorphic encryption. IEEE Access **6**, 20506–20519 (2018). https://doi.org/10.1109/ACCESS.2018.2817518, https://doi.org/10.1109/ACCESS.2018.2817518
29. Zhang, F., Kim, K.: Id-based blind signature and ring signature from pairings. In: International conference on the theory and application of cryptology and information security. pp. 533–547. Springer (2002)